Visualization of Regression Models Using `visreg`

Patrick Breheny
University of Kentucky

Woodrow Burchett
University of Kentucky

May 24, 2013

Abstract

Regression models allow one to isolate the relationship between the outcome and an explanatory variable while the other variables are held constant. Here, we introduce an R package, `visreg`, for the convenient visualization of this relationship via short, simple function calls. In addition to estimates of this relationship, the package also provides pointwise confidence bands and partial residuals to allow assessment of variability as well as outliers and other deviations from modeling assumptions. The package provides several options for visualizing models with interactions, including lattice plots, contour plots, and both static and interactive perspective plots. The implementation of the package is designed to take full advantage of R's generic functions, allowing a consistent interface for visualizing not only linear models, but generalized linear models, proportional hazards models, generalized additive models, robust regression models, and more.

In simple linear regression, it is both straightforward and extremely useful to plot the regression line. The plot tells you everything you need to know about the model and what it predicts. It is common to superimpose this line over a scatter plot of the two variables. A further refinement is the addition of a confidence band. Thus, in one plot, the analyst can immediately assess the empirical relationship between x and y in addition to the relationship estimated by the model and the uncertainty in that estimate, and also assess how well the two agree and whether assumptions may be violated.

Multiple regression models address a more complicated question: what is the relationship between an explanatory variable and the outcome as the other explanatory variables are held constant? This relationship is just as important to visualize as the relationship in simple linear regression, but doing so is not nearly as common in statistical practice.

As models get more complicated, it becomes more difficult to construct these sorts of plots. With multiple variables, we cannot simply plot the observed data, as this does not hold the other variables constant. Interactions among variables, transformations, and non-linear relationships all add extra barriers, making it time-consuming for the analyst to construct these plots. This is unfortunate, however – as models grow more complex, there is an even greater need to represent them with simple illustrations.

In this paper, we aim to eliminate the hurdle of implementation through the development of a simple interface for visualizing regression models arising from a wide class of models: linear models, generalized linear models, robust regression models, additive models, proportional hazards models, and more. We implement this interface in R (R Development Core Team, 2011) and provide it as the package `visreg`, publicly available from the Comprehensive R Archive Network (`http://CRAN.R-project.org/package=visreg`). The purpose of the package is to automate the tedious work involved in plotting regression functions, so that after fitting one of the above types of models, the analyst can construct attractive and illustrative plots with simple, one-line function calls. In particular, `visreg` offers several tools for the visualization of models containing interactions, which are among the easiest to misinterpret and the hardest to explain to non-statisticians.

It is worth noting that there are two distinct goals involved in plotting regression models: illustrating the fitted model visually and diagnosing violations of model assumptions. Our emphasis here is primarily on the former: providing visual aids to assist with understanding the fitted model. The partial residual plots provided by the package are helpful for diagnostic purposes, although their limitations have been pointed out by several authors (Mallows, 1986; Cook, 1993). Various extensions and modifications of partial residuals have been proposed, and there is an extensive literature on regression diagnostics (Belsley et al., 1980; Cook and Weisberg, 1982). It would be difficult to provide a generic implementation for diagnostic plots that would apply to all types of regression models, and `visreg` makes no attempt to do so. Indeed, many diagnostics are specific to the type of model (e.g., Pregibon, 1981; Grambsch and Therneau, 1994).

All regression models, however, describe how the response varies as a function of the explanatory variables, and in R, this is implemented for an extensive catalog of models using the generic `predict` function. It is this abstraction upon which `visreg` is based: the use of generic functions to provide a single tool with a consistent interface for the convenient visualization of a wide array of models.

The visualization ideas here are not new. Indeed, this project was inspired by the work of Trevor Hastie, Robert Tibshirani, and Simon Wood, who have convincingly demonstrated the utility of these types of plots in the context of generalized additive models (Hastie and Tibshirani, 1990; Wood, 2006). Furthermore, an existing R package, `effects` (Fox, 2003), provides many of the same tools as `visreg` (albeit without the partial residuals) for linear and generalized linear models. What `visreg` provides is a single, easy to use tool that may be applied to any model with a `predict` method. This enables the user to quickly create effective and aesthetically pleasing plots without having to learn new syntax for each type of model.

The outline of the paper is as follows. In Section 1, we explicitly define what we are plotting in `visreg` and provide the relevant mathematical details. The remainder of the article is devoted to illustrating the interface and results produced by the software in three extensions of simple linear regression: Section 2 deals with multiple (additive) linear regression models; Section 3 deals with models that possess interactions, and Section 4 deals with other sorts of models, such as generalized linear models, proportional hazards models, etc.

1 Conditional and contrast plots

We consider models in which the relationship between the outcome and the explanatory variables is expressed in terms of a linear predictor η:

$$\boldsymbol{\eta} = \mathbf{X}\boldsymbol{\beta} = \sum_j \mathbf{x}_j \beta_j, \tag{1.1}$$

where $\mathbf{x}_j$ is the jth column of the design matrix $\mathbf{X}$. For the sake of clarity, we focus in this section on linear regression, in which the expected value of the outcome $\mathbb{E}(Y_i)$ equals η_i; extensions to other, nonlinear models are discussed in Section 4. In the absence of interactions (see Section 3), the relationship between X_j and Y is neatly summarized by β_j, which expresses the amount by which the expected value of Y changes given a one-unit change in X_j.

Partial residuals are a natural multiple regression analog to plotting the observed x and y in simple linear regression. Partial residuals were developed by Ezekiel (1924), rediscovered by Larsen and McCleary (1972), and have been discussed in numerous papers and textbooks ever since (Wood, 1973; Atkinson, 1982; Kutner et al., 2004). Letting $\mathbf{r}$ denote the vector of residuals for a given model fit, the partial residuals belonging to variable j are defined as

$$\mathbf{r}_j = \mathbf{y} - \mathbf{X}_{-j}\widehat{\boldsymbol{\beta}}_{-j} \tag{1.2}$$

$$= \mathbf{r} + \mathbf{x}_j\widehat{\boldsymbol{\beta}}_j, \tag{1.3}$$

where the $-j$ subscript refers to the portion of $\mathbf{X}$ or $\boldsymbol{\beta}$ that remains after the jth column/element is removed.

The reason partial residuals are a natural extension to the multiple regression setting is that the slope of the simple linear regression of $\mathbf{r}_j$ on $\mathbf{x}_j$ is equal to the value $\widehat{\boldsymbol{\beta}}_j$ that we obtain from the multiple regression model (Larsen and McCleary, 1972).

Thus, it would seem straightforward to visualize the relationship between X_j and Y by plotting a line with slope β_j through the partial residuals. Clearly, however, we may add any constant to the line and to $\mathbf{r}_j$ and the above result would still hold. Nor is it obvious how the confidence bands should be calculated.

We consider asking two subtly different questions about the relationship between X_j and Y:

(1) What is the relationship between $\mathbb{E}(Y)$ and X_j given $\mathbf{x}_{-j} = \mathbf{x}^*_{-j}$?

(2) How do changes in X_j relative to a reference value x_j^* affect $\mathbb{E}(Y)$?

The biggest difference between the two questions is that the first requires specification of some $\mathbf{x}^*_{-j}$, whereas the second does not. The reward for specifying $\mathbf{x}^*_{-j}$ is that specific values for the predicted $\mathbb{E}(Y)$ may be plotted on the scale of the original variable Y; the latter type of plot can address only relative changes. Here, we refer to the first type of plot as a *conditional plot*, and the second type as a *contrast plot*. As we will see, the two questions produce regression lines with identical slopes, but with different intercepts and confidence bands. It is worth noting that these are not the only possible questions; other possibilities, such as "What is the marginal relationship between X_j and Y, integrating over $\mathbf{X}_{-j}$?" exist, although we do not explore them here.

For a contrast plot, we consider the effect of changing X_j away from an arbitrary point x_j^*; the choice of x_j^* thereby determines the intercept, as the line by definition passes through $(x_j^*, 0)$. The equation of this line is $y = (x - x_j^*)\hat{\beta}_j$. For a continuous X_j, we set x_j^* equal to $\bar{x}_j$. The confidence interval at the point $x_j = x$ is based on

$$V(x) = \mathbb{V}\left\{\hat{\eta}(x) - \hat{\eta}(x_j^*)\right\} = (x - x_j^*)^2 \mathbb{V}(\hat{\beta}_j).$$

When X_j is categorical, we plot differences between each level of the factor and the reference category (see Figure 3 for an example); in this case, we are literally plotting contrasts in the classical ANOVA sense of the term (hence the name). Our usage of the term "contrast" for continuous variables is somewhat looser, but still logical in the sense that it estimates the contrast between a value of X_j and the reference value.

For a conditional plot, on the other hand, all explanatory variables are fully specified by x and $\mathbf{x}^*_{-j}$. Let $\lambda(x)^T$ denote the row of the design matrix that would be constructed from $x_j = x$ and $\mathbf{x}^*_{-j}$. Then the equation of the line is $y = \lambda(x)^T\widehat{\boldsymbol{\beta}}$ and and the confidence interval at x is based on

$$V(x) = \mathbb{V}\left\{\lambda(x)^T\widehat{\boldsymbol{\beta}}\right\} = \lambda(x)^T \mathbb{V}(\widehat{\boldsymbol{\beta}})\lambda(x).$$

In both conditional and contrast plots, the confidence interval at x is then formed around the estimate in the usual manner by adding and subtracting $t_{n-p,1-\alpha/2}\sqrt{V(x)}$, where $t_{n-p,1-\alpha/2}$ is $1-\alpha/2$ quantile of the t distribution with $n-p$ degrees of freedom. Examples of contrast plots and conditional plots are given in Figures 2 and 3. Both plots depict the same relationship between wind and ozone level as estimated by the same model (details given in Section 2). Note the difference, however, in the vertical scale and confidence bands. In particular, the confidence interval for the contrast plot has zero width at x_j^*; all other things remaining the same, if we do not change X_j, we can say with certainty that $\mathbb{E}(Y)$ will not change either. There is still uncertainty, however, regarding the actual value of $\mathbb{E}(Y)$, which is illustrated in the fact that the confidence interval of the conditional plot has positive width everywhere.

2 Additive linear models

We are now ready to describe the basic framework and usage of `visreg`. In this section, we will fit various models to a data set involving the relationship between air quality (in terms of ozone concentration) and various aspects of weather in the standard R data set `airquality`.

2.1 Basic framework

The basic interface to the package is the function `visreg`, which requires only one argument: the fitted model object. So, for example, the following code produces Figure 1:

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

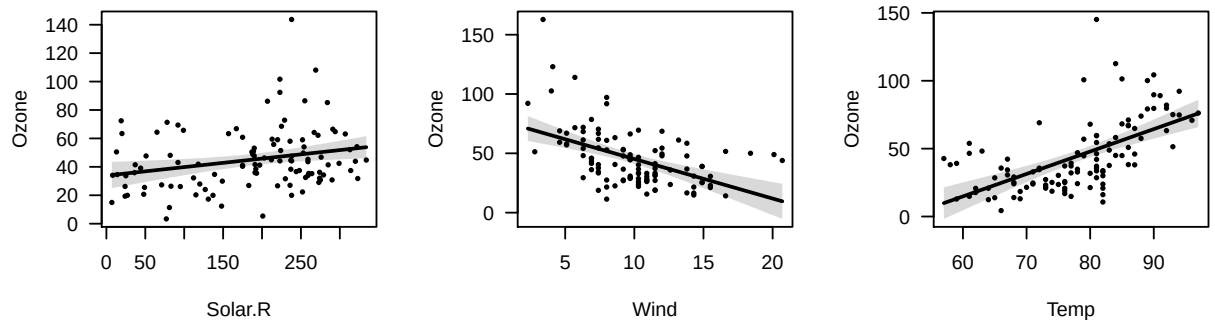

Figure 1: Basic output of `visreg` for an additive linear model: conditional plots for each explanatory variable.

By default, `visreg` provides conditional plots for each of the explanatory variables in the model. For the conditioning, the other variables in $\mathbf{x}^*_{-j}$ are set to their median for numeric variables and to the most common category for factors. All of these options can be modified by passing additional arguments to visreg. For example, contrast plots can be obtained with the `type` argument; the following code produces Figure 2.

```
> visreg(fit, "Wind", type = "contrast")
> visreg(fit, "Wind", type = "conditional")
```

The second argument specifies the explanatory variable to be visualized; note that the right plot in Figure 2 is the same as the middle plot in Figure 1.

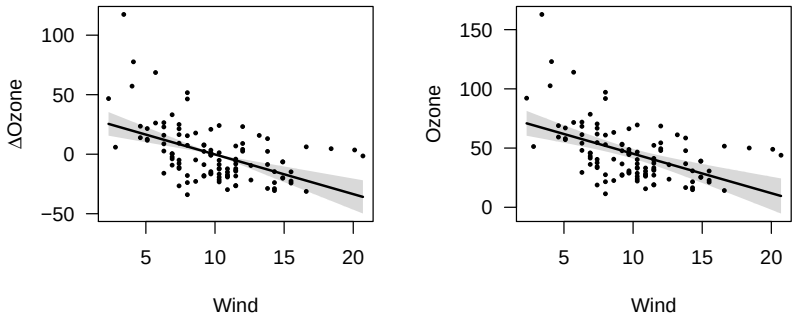

Figure 2: The estimated relationship between wind and ozone concentration in the same model, as illustrated by two different types of plots. Left: Contrast plot. Right: Conditional plot.

In addition to continuous explanatory variables, `visreg` also allows the easy visualization of differences between the levels of categorical variables (factors). The following block of code creates a factor called `Heat` by discretizing `Temp`, and then visualizes its relationship with `Ozone`, producing the plot in Figure 3.

```
> 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")
```

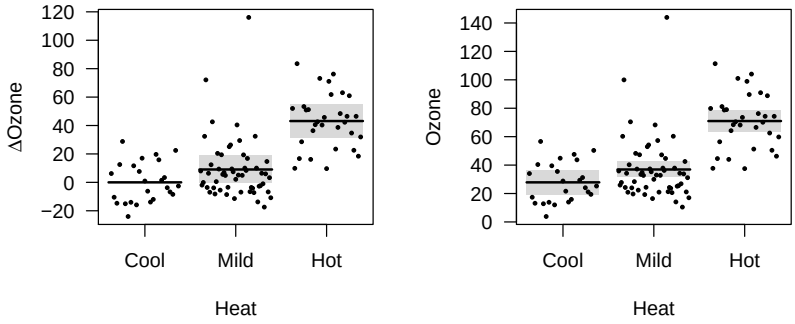

Figure 3: Visualization of a regression function involving a categorical explanatory variable. Left: Contrast plot. Right: Conditional plot.

Again, note that the confidence interval for the contrast plot has zero width for the reference category. There is no uncertainty about how the expected value of ozone will change if we remain at the same level of `Heat`; it is zero by definition. On the other hand, the width of the confidence interval for `Mild` heat is wider for the contrast plot than it is for the conditional plot. There is less uncertainty about the expected value of ozone on a mild day than there is about the difference in expected values between mild and cool days.

2.2 Transformations

Often in modeling, we introduce transformations of explanatory variables, transformations of the response variable, or both. The `visreg` package automatically handles these transformations when visualizing the regression model.

Linear models assume a linear relationship between the explanatory variables and the outcome. A common way of extending the linear model is to introduce transformations of the original explanatory variables. For example, to allow the effect of wind on ozone to be nonlinear, we may introduce a quadratic term for wind into the model:

```
> fit1 <- lm(Ozone ~ Solar.R + Wind + I(Wind^2) + Temp, data = airquality)
```

Transformations of the response are also common. For example, ozone levels must be positive. However, as Figure 1 illustrates, a standard linear model allows the estimated relationship and its confidence band to fall below 0. One way of remedying this is to model the log of ozone concentrations instead of the ozone concentrations directly:

```
> fit2 <- lm(log(Ozone) ~ Solar.R + Wind + Temp, data = airquality)
```

And of course, these elements may be combined:

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

Visualization is particularly important in these models, as it is difficult to determine the exact nature of the relationship between explanatory variable and response simply by looking at the regression coefficients

when that relationship is nonlinear. The `visreg` package provides a convenient way to view such relationships. Transformations involving explanatory variables are handled automatically, while transformations involving the response require the user to provide the inverse transformation. The following code produces Figure 4.

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

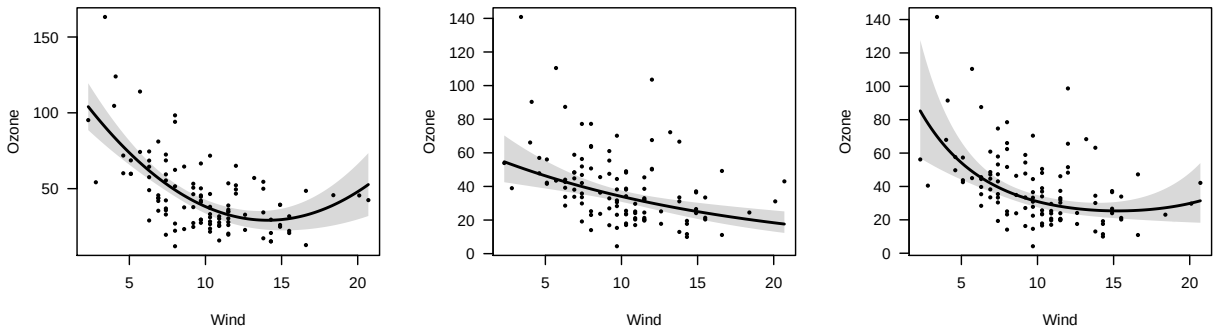

Figure 4: Plots of the modeled relationship between wind and ozone concentration, as estimated by different models. Left: The model contains a transformation of wind. Middle: The model contains a transformation of ozone concentration. Right: The model contains transformations of both wind and ozone.

2.3 Conditioning

As noted in Section 2.1, the default behavior of `visreg` when constructing a conditional plot is to fill in $\mathbf{x}^*_{-j}$ with the median for continuous variables and the most common category for categorical variables. This behavior can be modified using the `cond` argument. Note that this has no bearing on contrast plots in additive models, which do not require a full specification of $\mathbf{x}^*_{-j}$.

The `cond` argument must be provided as a named list. Each element of that list specifies the value for an element of $\mathbf{x}^*_{-j}$; any elements left unspecified are filled in with the median/most common category. We revisit our initial model from Section 2.1 with this code, which produces Figure 5.

```
> visreg(fit, "Wind", cond = list(Temp = 50))
> visreg(fit, "Wind")
> visreg(fit, "Wind", cond = list(Temp = 100))
```

We make several observations concerning Figure 5: i) The values on the vertical axis differ; as we condition on higher temperatures, the expected ozone concentration goes up since the regression coefficient for temperature is positive. ii) The slope of the line, the distance from the line to each residual, and the range of the residuals is the same in all three plots; conditioning on different values of temperature merely adds a constant to the regression line and the partial residuals. iii) The width of the confidence band *does* change, however: the data set has few observations at very high and very low temperatures, so the standard errors are much larger for the plots on the right and left than for the plot in the middle. iv) The shape of the confidence band also changes. In the middle plot, the confidence band is narrowest in the middle and wider at the ends. In the left plot (conditioning on low temperature), however, the confidence band is narrowest

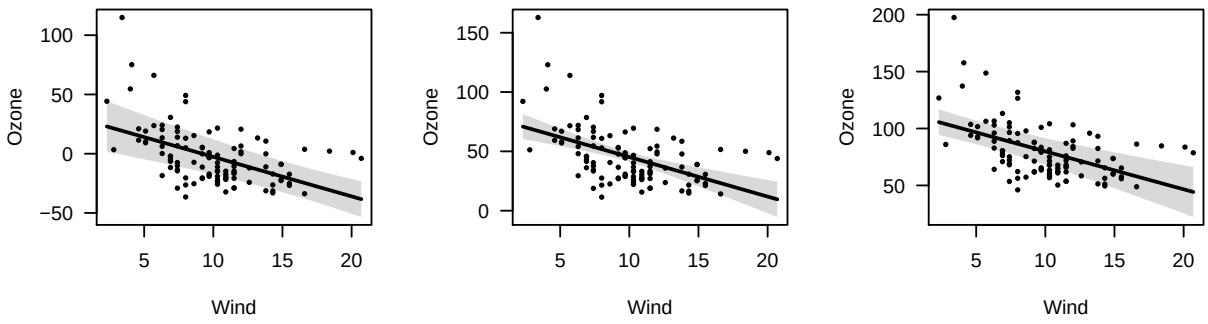

Figure 5: Estimated relationship between wind and ozone concentration, conditioning on different values of temperature. Left: Temperature=50 °F. Middle: The median temperature, 79 °F (default). Right: Temperature=100 °F.

for high wind levels. This arises because there is a negative correlation between wind and temperature ($\hat{\rho} = -0.46$), and thus, more cold windy days in the data set than cold calm days. The opposite phenomenon happens in the right plot, where the relative absence of hot windy days causes the confidence band to be wider for high winds than for low winds.

Recall that this model had three explanatory variables; in the above example, `visreg` calculated the conditional response by filling in solar radiation with its median value, as it was not specified otherwise in the `cond` argument.

3 Linear models with interactions

Visualization is also very important for models with interactions – as with polynomial terms, in these models the relationship between an explanatory variable and the response depends on multiple regression coefficients, and a model's fit is more readily understood with a visual representation than by looking at a table of regression coefficients.

For models with interactions, we must simultaneously visualize the effect of two explanatory variables. The `visreg` package offers two methods for doing this: cross-sectional plots, which plot one-dimensional relationships between the response and one predictor for several values of another predictor, and surface plots, which attempt to provide a picture of the regression surface over both dimensions simultaneously.

3.1 Cross-sectional plots

To begin, let's fit a model that involves an interaction between a continuous term and a categorical term, using our derived variable Heat from Section 2.1:

```
> fit <- lm(Ozone ~ Solar.R + Wind * Heat, data = airquality)
```

The `visreg` package creates cross-sectional plots using the `lattice` package (Sarkar, 2008). To request a cross-sectional plot, the user specifies a `by` variable, as in the following code which produces Figure 6.

```
> visreg(fit, "Wind", by = "Heat")
```

The cross-sectional plot in Figure 6 allows us to see that the relationship between wind and ozone concentration appears to become more pronounced depending on how hot the day is. On cool days, wind

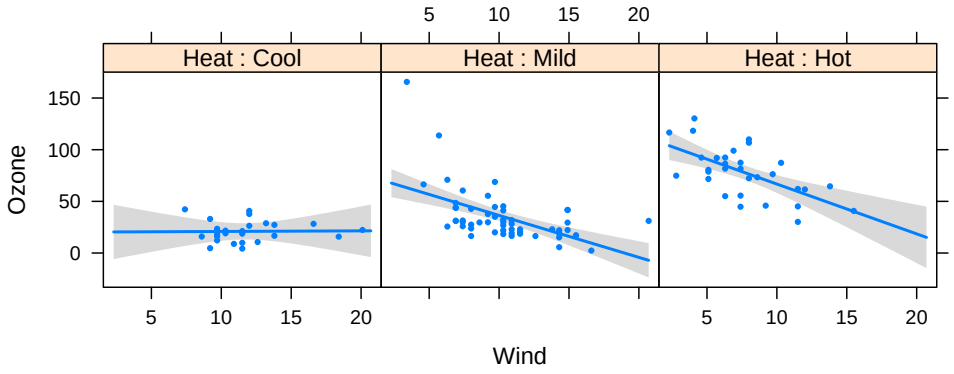

Figure 6: Cross-sectional plots depicting the fit of a model with an interaction between a continuous term (Wind) and a categorical term (Heat), with the continuous term on the horizontal axis.

has no effect on ozone concentration. Wind has a moderate effect on ozone concentrations on mild days, and an even larger effect on hot days.

Note that `visreg` handles the partial residuals properly – the partial residuals for observations collected on cool days appear only in the left panel, and so on. As with the earlier plots, this ensures that the least squares line drawn through the residuals on the plot will yield the same slope as that estimated by the full model fit. Furthermore, this allows us to see potentially influential observations like the one in the middle panel, which has very low wind and very high ozone concentration. Finally, the proper handling of partial residuals also allows us to observe the lack of hot windy days and cool days with no wind that we commented on in Section 2.3. Note that the confidence intervals in these regions are comparatively wide.

Alternatively, we may wish to overlay these cross-sections. This allows for a more direct comparison between the different regression lines, although it often becomes difficult to include partial residuals and confidence bands without crowding the figure. The visreg package allows an `overlay` option for creating these plots:

```
visreg(fit, "Wind", by="Heat", overlay=TRUE, partial=FALSE)
```

The above code produces Figure 7, where the plotting of partial residuals has been turned off for the sake of clarity (similarly, `band=FALSE` can be specified to turn off the confidence bands). If `partial=TRUE`, the partial residuals are colored according to the existing scheme.

The above examples featured a continuous variable along the horizontal axis and a categorical variable as the `by` variable. However, `visreg` allows each of these variables to be either continuous or categorical. For example, let us try plotting the same model, but reversing the roles of Heat and Wind (Figure 8).

```
> visreg(fit, "Heat", by = "Wind")
```

The model is the same, but the emphasis of the plot is now on heat instead of wind. Figure 8 illustrates that heat has a pronounced effect on ozone concentration when the day is not windy, but a relatively insignificant effect on ozone for windy days.

In contrast to Figure 6, where it was natural to construct a panel for each level of the categorical variable, Figure 7 requires arbitrary decisions concerning how many cross-sections to take, and where to place them. The default behavior of `visreg` is to take cross-sections at the 10th, 50th, and 90th percentiles of the by variable, although both the number of points and their location can be modified using the `breaks` option. Again, each residual appears only once, in the panel it is closest to. However, the least squares estimates are

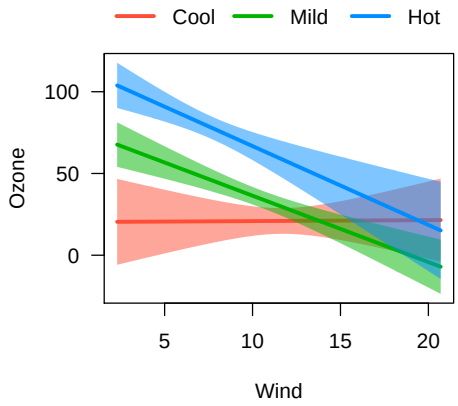

Figure 7: Cross-sectional plot depicting the fit of a model with an interaction between a continuous term (Wind) and a categorical term (Heat), where the regression lines for each category are overlaid.

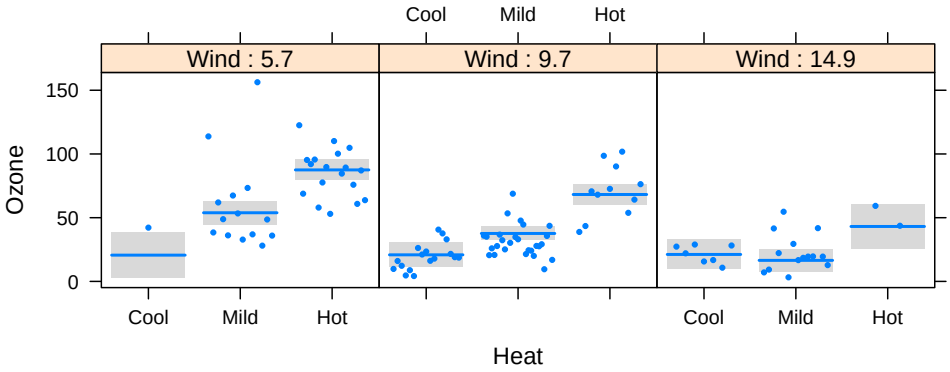

Figure 8: Cross-sectional plots depicting the fit of a model with an interaction between a continuous term (Wind) and a categorical term (Heat), with the categorical term on the horizontal axis.

no longer equivalent to drawing a line through the partial residuals due to the continuous manner in which information is pooled across the panels.

We have been focusing here on conditional plots, but contrast plots can be made in the same way. For example, the code below produces the appropriate contrast plot analogue to Figure 6.

```
> visreg(fit, "Wind", by = "Heat", type = "contrast")
```

It is worth noting that for a model containing an interaction, a basic call to `visreg` (*i.e.*, without a `by` argument) amounts to plotting a main effect in the presence of an interaction. Because this has the potential to be misleading, `visreg` by default prints a message warning the user of this and reminding him or her of the levels of the other variables at which the plot is constructed. For example, since `'Mild'` is the most common level of `Heat`, `visreg(fit, ''Wind'')` will produce the middle panel of Figure 6. The left and right panels, respectively, would be produced with

```
> visreg(fit, "Wind", cond = list(Heat = "Cool"))
> visreg(fit, "Wind", cond = list(Heat = "Hot"))
```

3.2 Surface plots

Another approach to visualizing models with interactions is plotting the regression surface using contour or perspective plots. Suppose we fit a complicated model involving a number of interactions between linear and quadratic terms involving wind and temperature:

```
> 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)
```

Whether or not this is a good model for analyzing these data is a good question, but we will not address it here. Our purpose is to show that it is difficult to grasp the fit of the model by looking at the regression coefficients directly, but easy to do so using `visreg`. In addition to the tools for creating cross-sectional plots described in the Section 3.1, the `visreg` package provides the function `visreg2d`, which can be used to produce two-dimensional contour and perspective plots. The following code produces Figure 9:

```
> visreg2d(fit, "Wind", "Temp", plot.type = "image")
> visreg2d(fit, "Wind", "Temp", plot.type = "persp")
```

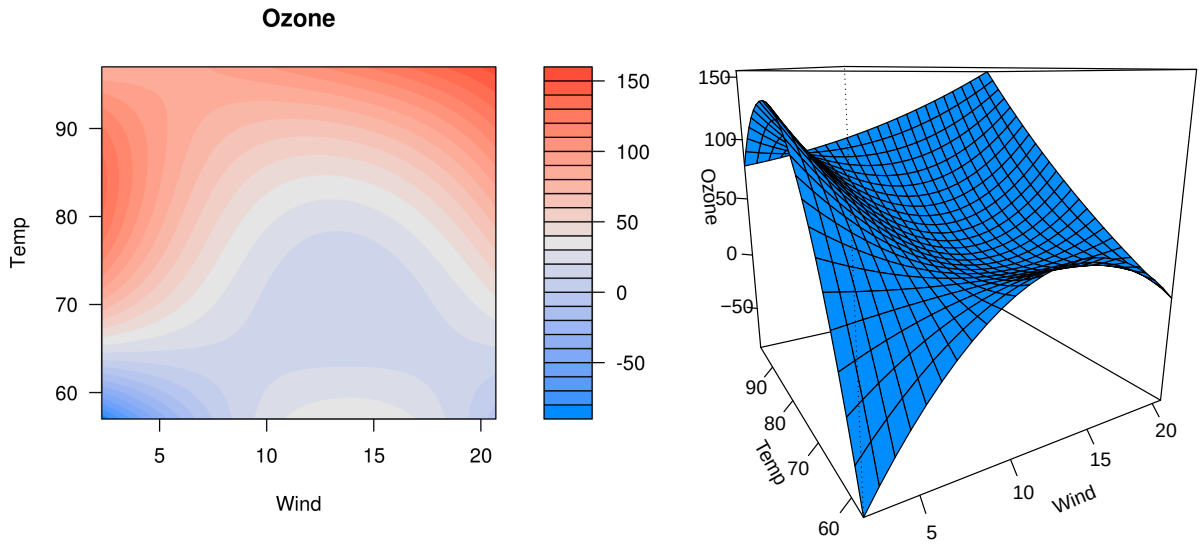

Figure 9: Representations of the regression surface as a function of wind and temperature. Left: Filled contour plot. Right: Perspective plot.

The advantage of these kinds of plots compared with those in Section 3.1 are that they allow us to visualize the effect of simultaneously varying two factors. The disadvantage is that there is no convenient way of superimposing either residuals or confidence intervals. These plots are most useful when both variables are continuous, as one is not forced to take cross-sections over a continuous variable. The `visreg2d` function still functions correctly when one or both of its arguments is a categorical variable, although in our opinion, the cross-section plots of Section 3.1 are more useful in these settings.

In addition to the static perspective plot presented above, `visreg2d` can also create interactive perspective plots using the `rgl` package (Adler and Murdoch, 2011), which allow the user to rotate, tilt, and spin the regression surface. This makes it considerably easier to comprehend its three-dimensional shape. Such plots can be constructed with the code:

```
> visreg2d(fit, x = "Wind", y = "Temp", plot.type = "rgl")
```

Visualization of higher-order interactions, such as three-way or four-way interactions, becomes increasingly difficult. To some extent, `visreg` facilitates visualization of such models through the use of the `cond` argument. For example, code such as the following could be used to visualize a three-way interaction:

```
> fit <- lm(Ozone ~ Solar.R * Wind * Temp, data = airquality)
> visreg2d(fit, "Wind", "Temp", cond = list(Solar.R = 100))
> visreg2d(fit, "Wind", "Temp", cond = list(Solar.R = 300))
```

4 Nonlinear models

As mentioned at the outset, the goal in creating the `visreg` package was to implement it in as generic a manner as possible, so that it works with a wide variety of model fits from different functions and packages. All that it requires is functioning `model.frame` and `predict` methods for the fitted model object (plotting of partial residuals requires a `residuals` method as well). Thus, the `visreg` package and all its options work not only with linear models as produced by `lm`, but with generalized linear models produced by `glm`, proportional hazards models produced by `coxph` (Therneau, 2012), robust linear models produced by `rlm` (Venables and Ripley, 2002), negative binomial models produces by `glm.nb` (Venables and Ripley, 2002), generalized additive models produced by `gam` (Wood, 2012), local regression models produced by `loess` and `locfit` (Loader, 2010), and many more. Indeed, the type of object does not even need to be part of an R package; user-defined model classes can also be visualized with `visreg`, provided that they are compatible with `model.frame` and `predict`. Two notable exceptions are the `nls` function, which does not work with `model.frame`, and `lmer` (Bates et al., 2012), which does not have a `predict` method. In this section, we briefly illustrate the use of `visreg` with some of the above types of models, and then comment on some aspects of these plots that change when applied to nonlinear models.

We begin with a logistic regression model applied to a study investigating risk factors associated with low birth weight (Hosmer and Lemeshow, 2000). The following code produces Figure 10.

```
> data("birthwt", package = "MASS")
> 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", rug = 2,
+         xlab = "Mother's weight", ylab = "P(low birthweight)")
```

On the left side of Figure 10, the model is plotted on the scale of the linear predictor (the default scale in `visreg`), where the model is indeed linear. The confidence intervals in the figure are Wald confidence intervals based on standard errors returned by `predict.glm`. The partial residuals are calculated based on Equation 1.2, with $\mathbf{r}$ the deviance residuals (the default residuals returned by `residuals.glm`). The plot on the right is simply a transformed version of the plot on the left, where an inverse logistic transformation has been applied to the regression line and confidence bands (this is handled automatically by the `scale = "response"` option).

Note that for the plot on the right, we have opted to plot a rug as opposed to the partial residuals. The `visreg` package provides two types of rug annotations. With `rug=TRUE` or `rug=1`, a standard rug along the bottom of the plot is provided. With `rug=2`, separate rugs are drawn on the top for observations with positive residuals and on the bottom for observations with negative residuals (for logistic regression, this corresponds to $Y = 1$ and $Y = 0$, respectively).

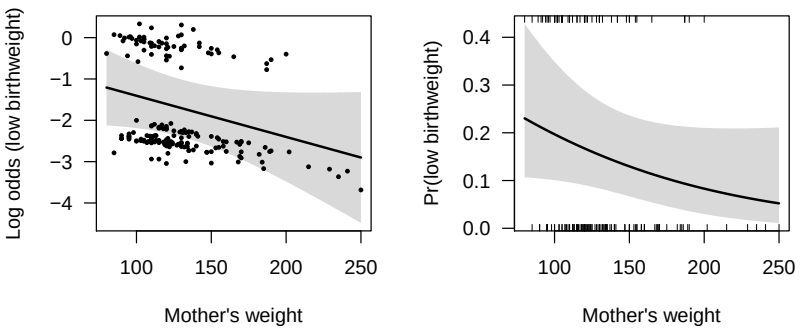

Figure 10: Visualization of a logistic regression model. Left: Log odds scale. Right: Probability scale.

In practice, we have found plots like those on the left useful for visualizing the model fit and observing potential departures from model assumptions such as outliers and influential points, and plots like those on the right very useful for communicating modeling results to non-statisticians.

We conclude with a brief demonstration applying `visreg` to some other types of models (note that these are models for which the `effects` package is incompatible): a proportional hazards model, a robust regression model, and a local regression model. The left side of Figure 11 presents a visualization of the following proportional hazards model:

```
> require("survival")
> fit <- coxph(Surv(futime, fustat) ~ age + rx, data = ovarian)
> visreg(fit, "age", ylab = "log(Hazard ratio)")
```

Note that in proportional hazards models, baseline hazard functions are not explicitly estimated, and therefore the meaning behind a conditional plot is questionable. For this reason, contrast plots are (arguably) more appropriate. A similar phenomenon occurs with logistic regression applied to case-control studies, in which an intercept is estimated, but is the estimate is heavily biased by the study design.

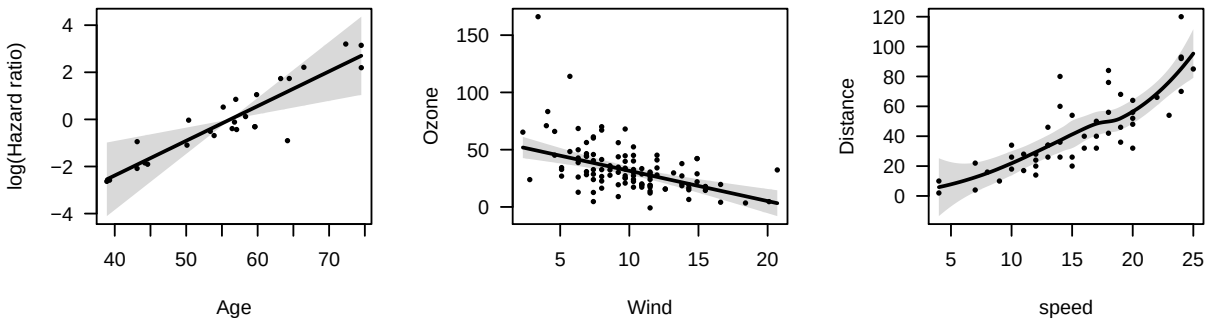

Figure 11: Visualizations of robust regression and proportional hazard models. Left: Proportional hazards. Right: Robust regression.

The middle of Figure 11 presents a visualization of the following robust regression model:

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

Note that the design matrix for the robust regression model is the same as that from Section 3.1, and that the plot in the middle of Figure 11 is analogous to the middle panel from Figure 6. Note, however, that the robust regression model produces a different fit, due in part to the reduced impact of the potential outlier mentioned in Section 3.1. Specifically, the fit produced by the robust regression model is flatter and does not predict negative ozone concentrations for high wind levels as the linear regression model does.

Finally, we apply `visreg` to a local regression model fit with `loess`, producing a visualization of the model far superior to that provided by the default `plot` method for `loess`. This plot appears on the right side of Figure 11.

```
fit <- loess(dist ~ speed, cars)
visreg(fit, ylab="Distance")
```

All of the features and options we mentioned earlier, in particular the cross-section and surface plots of Section 3, work in the same way for nonlinear models as they do for linear models. We note that no internal re-coding was necessary for `visreg` to work these models; the compatibility was automatically provided by the use of generic functions.

The extension of `visreg` to nonlinear models is straightforward in its implementation, but it is worth making some comments about partial residuals for nonlinear models. In particular, it is no longer the case that the regression line through the partial residuals produces a line with the same slope as that produced by the model. Viewing nonlinear models as reweighted least squares models, the observations have different weights and these weights are not reflected in the partial residuals plotted by `visreg`. This phenomenon has been commented on by many authors, with a variety of proposals for alternative types of reweighted partial residuals that may be better at detecting outliers and influential observations (Pregibon, 1981; Landwehr et al., 1984; O'Hara Hines and Carter, 1993).

Residuals and how useful they are in detecting influential observations and departures from model assumptions may of course depend on the model. Other types of plots, such as added variable plots (Atkinson, 1985), are also helpful for visualizing regression models and their fit. We feel that the approach we have provided here is reasonable and the best that can be expected from a generic tool with broad applicability to a wide variety of models, although we certainly concede that improvements may be possible for certain models and certain visualization goals.

To facilitate extensions and modifications of the plots provided by `visreg`, its functions also invisibly return the data frames, estimates, confidence intervals, and residuals used in the construction of its plots. We anticipate that the majority of users would be uninterested in these details, but any user wishing to modify `visreg`'s output (*e.g.*, changing the size of the partial residuals to reflect their weight) may do so by assigning its output to an object, as in the following code:

```
> v <- visreg(fit, "Wind", cond = list(Heat = "Mild"))
```

5 Conclusion

The `visreg` package provides a very useful set of tools for simultaneously visualizing the estimated relationship between an explanatory variables and the outcome, the variability of that estimate, and the observations from which the estimates derive. These tools have a simple interface and are readily applied to wide variety of models. We have found the development of this package to provide a convenient and versatile tool to assist with regression modeling, both for model exploration and for communicating modeling results.

References

Adler, D. and Murdoch, D. (2011). *rgl: 3D Visualization Device System (OpenGL).* R package version 0.92.798, URL `http://CRAN.R-project.org/package=rgl`.

Atkinson, A. C. (1982). Regression diagnostics, transformations and constructed variables. *Journal of the Royal Statistical Society. Series B (Methodological)*, **44** pp. 1–36.

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

Bates, D., Maechler, M. and Bolker, B. (2012). *lme4: Linear mixed-effects models using S4 classes.* R package version 0.999999-0, URL `http://CRAN.R-project.org/package=lme4`.

Belsley, D. A., Kuh, E. and Welsch, R. E. (1980). *Regression diagnostics.* Wiley.

Cook, R. (1993). Exploring partial residual plots. *Technometrics*, **35** 351–362.

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

Ezekiel, M. (1924). A method of handling curvilinear correlation for any number of variables. *Journal of the American Statistical Association*, **19** 431–453.

Fox, J. (2003). Effect displays in r for generalised linear models. *Journal of Statistical Software*, **8** 1–27.

Grambsch, P. M. and Therneau, T. M. (1994). Proportional hazards tests and diagnostics based on weighted residuals. *Biometrika*, **81** pp. 515–526.

Hastie, T. and Tibshirani, R. (1990). *Generalized Additive Models.* Chapman & Hall/CRC.

Hosmer, D. and Lemeshow, S. (2000). *Applied Logistic Regression.* Wiley.

Kutner, M., Nachtsheim, C., Neter, J. and Li, W. (2004). *Applied Linear Statistical Models.* McGraw-Hill.

Landwehr, J., Pregibon, D. and Shoemaker, A. (1984). Graphical methods for assessing logistic regression models. *Journal of the American Statistical Association*, **79** 61–71.

Larsen, W. and McCleary, S. (1972). The use of partial residual plots in regression analysis. *Technometrics*, **14** 781–790.

Loader, C. (2010). *locfit: Local Regression, Likelihood and Density Estimation.* R package version 1.5-6, URL `http://CRAN.R-project.org/package=locfit`.

Mallows, C. (1986). Augmented partial residuals. *Technometrics*, **28** 313–319.

O'Hara Hines, R. J. and Carter, E. M. (1993). Improved added variable and partial residual plots for the detection of influential observations in generalized linear models. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, **42** pp. 3–20.

Pregibon, D. (1981). Logistic regression diagnostics. *Annals of Statistics*, **9** 705–724.

R Development Core Team (2011). *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL `http://www.R-project.org/`.

Sarkar, D. (2008). *Lattice: Multivariate Data Visualization with R.* Springer.

Therneau, T. (2012). *A Package for Survival Analysis in S.* R package version 2.36-12, URL `http://CRAN.R-project.org/package=survival`.

Venables, W. and Ripley, B. (2002). *Modern Applied Statistics with S*. Springer.

Wood, F. (1973). The use of individual effects and residuals in fitting equations to data. *Technometrics*, **15** 677–695.

Wood, S. (2006). *Generalized Additive Models: An Introduction with R*. Chapman & Hall/CRC.

Wood, S. (2012). *mgcv: Mixed GAM Computation Vehicle with GCV/AIC/REML Smoothness Estimation*. R package version 1.7-17, URL `http://CRAN.R-project.org/package=mgcv`.