Residuals and Diagnostics for Binary and Ordinal Regression Models: An Introduction to the sure package

by Brandon M. Greenwell, Andrew J. McCarthy, Bradley C. Boehmke, and Dungang Liu

Abstract Residual diagnostics is an important topic in the classroom, but it is less often used in practice when the response is binary or ordinal. Part of the reason for this is that generalized models for discrete data, like cumulative link models and logistic regression, do not produce standard residuals that are easily interpreted as those in ordinary linear regression. In this paper, we introduce the R package **sure**, which implements a recently developed idea of **SU**rrogate **RE**siduals. We demonstrate the utility of the package in detection of cumulative link model misspecification with respect to mean structures, link functions, heteroscedasticity, proportionality, and interaction detection.

Introduction

Categorical outcomes are encountered frequently in practice across different fields. For example, in medical studies, the outcome of interest is often binary (e.g., presence or absence of a particular disease after applying a treatment). A binary outcome can be viewed as a special case of an ordinal outcome; that is, a categorical outcome having a natural ordering. For instance, in an opinion poll, the response may be satisfaction, with categories low, medium, and high. In this case, the response is ordered: low $<$ medium $<$ high.

Logistic and probit regression are popular choices for modelling a binary outcome. Although this paper focuses on models for ordinal responses, the surrogate approach to constructing residuals actually applies to a wide class of general models of the form

$$\mathcal{Y} \sim F_a\left(y; \boldsymbol{X}, \boldsymbol{\beta}\right)$$

where $F_a\left(\cdot\right)$ is a discrete cumulative distribution function. This includes binary regression as a special case. For example, the probit model has

$$\mathcal{Y} \sim Bernoulli\left[\Phi\left(\boldsymbol{X}^\top \boldsymbol{\beta}\right)\right],$$

where $\Phi\left(\cdot\right)$ is the cumulative distribution function for the standard normal distribution.

The *cumulative link model* is a natural choice for modelling a binary or ordinal outcome. Consider an ordinal categorical outcome $\mathcal{Y}$ with ordered categories $1 < 2 < \cdots < J$. In a cumulative link model, the cumulative probabilities are linked to the predictors according to

$$G^{-1}\left(\Pr\left\{\mathcal{Y} \le j\right\}\right) = \alpha_j + f\left(\boldsymbol{X}, \boldsymbol{\beta}\right), \tag{1}$$

where G is a continuous cumulative distribution function, α_j are the category-specific intercepts, $\boldsymbol{X}$ is a matrix of covariates, and $\boldsymbol{\beta}$ is a vector of fixed regression coefficients. The intercept parameters satisfy $-\infty = \alpha_0 < \alpha_1 < \cdots < \alpha_{J-1} < \alpha_J = \infty$. We should point out that some authors (and software) use the alternate formulation

$$G^{-1}\left(\Pr\left\{\mathcal{Y} \ge j\right\}\right) = \alpha_j^\star + f\left(\boldsymbol{X}, \boldsymbol{\beta}^\star\right), \tag{2}$$

This formulation provides coefficients that are consistent with the ordinary logistic regression model. The estimated coefficients from model (2) will have the opposite sign as those in model (1).

Another way to interpret the cumulative link model is through a *latent* continuous random variable $\mathcal{Z} = -f\left(\boldsymbol{X}, \boldsymbol{\beta}\right) + \epsilon$, where ϵ is a continuous random variable with location parameter 0, scale parameter 1, and cumulative distribution function $G\left(\cdot\right)$. We then construct an ordered factor according to the rule

$$y = j \quad if \quad \alpha_{j-1} < z \le \alpha_j.$$

For $\epsilon \sim N\left(0, 1\right)$, this leads to the usual probit model for ordinal responses,

$$\Pr\left\{\mathcal{Y} \le j\right\} = \Pr\left\{\mathcal{Z} \le \alpha_j\right\} = \Pr\left\{-f\left(\boldsymbol{X}, \boldsymbol{\beta}\right) + \epsilon \le \alpha_j\right\} = \Phi\left(\alpha_j + f\left(\boldsymbol{X}, \boldsymbol{\beta}\right)\right).$$

Common choices for the link function $G^{-1}\left(\cdot\right)$ and the implied (standard) distribution for ϵ are described in Table 1.

Link	Distribution of ϵ	$G(y)$	$G^{-1}(p)$
logit	logistic	$\exp(y) / [1+\exp(y)]$	$\log[p/(1-p)]$
probit	standard normal	$\Phi(y)$	$\Phi^{-1}(p)$
log-log	Gumbel (max)	$\exp[-\exp(-y)]$	$-\log[-\log(p)]$
complementary log-log	Gumbel (min)	$1-\exp[-\exp(y)]$	$\log[-\log(1-p)]$
cauchit	Cauchy	$\pi^{-1}\arctan(y)+1/2$	$\tan(\pi p-\pi/2)$

Table 1: Common link functions. **Note:** the logit is typically the default link function used by most statistical software.

There are a number of R packages that can be used to fit cumulative link models (1) and (2). The recommended package **MASS** (Venables and Ripley, 2002) contains the function `polr` (proportional odds logistic regression) which, despite the name, can be used with all of the link functions described in Table 1. The **VGAM** package (Yee, 2017) has the `vglm` function for fitting vector generalized linear models, which includes the broad class of cumulative link models. By default, `vglm` uses the same parameterization as in Equation (1), but provides the option of using the parameterization seen in Equation (2); this will result in the estimated coefficients having the opposite sign. Package **ordinal** (Christensen, 2015) has the `clm` function for fitting cumulative link models. The popular **rms** package (Harrell Jr, 2017) has two functions: `lrm` for fitting logistic regression and cumulative link models using the logit link, and `orm` for fitting ordinal regression models. Both of these functions use the parameterization seen in Equation (2).

For a continuous outcome $\mathcal{Y}$, the residual is traditionally defined as the difference between the observed and fitted values. For ordinal outcomes, the residuals are more difficult to define, and few definitions have been proposed in the literature. Liu et al. (2009) proposed using the cumulative sums of residuals derived from collapsing the ordered categories into multiple binary outcomes. Unfortunately, this method leads to multiple residuals for the ordinal outcome and therefore is difficult to interpret. Li and Shepherd (2012) showed that the sign-based statistic (SBS)

$$R_{SBS} = E\left\{sign\left(y-\mathcal{Y}\right)\right\} = Pr\left\{y > \mathcal{Y}\right\} - Pr\left\{y < \mathcal{Y}\right\}, \tag{3}$$

can be used as a residual for ordinal outcomes; these are referred to later by Li and Shepherd as *probability-based residuals*, but we will follow Liu and Zhang (2017) and refer to them as SBS residuals. For an overview of the theoretical and graphical properties of the SBS residual (3), see Liu and Zhang (2017). These are available in the **PResiduals** package (Dupont et al., 2016). A limitation with the SBS residuals is that they are based on a discrete outcome and hence, are discrete themselves. This makes using them in various diagnostic plots far less useful.

Surrogate residuals

Liu and Zhang (2017) propose a new type of residual that is based on a continuous variable S that acts as a surrogate for the ordinal outcome $\mathcal{Y}$. This surrogate residual is defined as

$$R_S = S - E\left(S|\boldsymbol{X}\right), \tag{4}$$

where S is a continuous variable based on the conditional distribution of the latent variable $\mathcal{Z}$ given $\mathcal{Y}$. In particular, given $\mathcal{Y}=y$, Liu and Zhang (2017) show that S follows a truncated distribution obtained by truncating the distribution of $\mathcal{Z} = -f\left(\boldsymbol{X}, \boldsymbol{\beta}\right) + \epsilon$ using the interval $\left(\alpha_{y-1}, \alpha_y\right)$. The benefit of the surrogate residual (4) is that it is based on a continuous variable S, hence, R_S is also continuous.

Furthermore, it can be shown (Liu and Zhang, 2017) that if the hypothesized model agrees with the true model, then R_S will have the following properties:

symmetry around zero $E\left(R_S|\boldsymbol{X}\right) = 0$;

homogeneity $Var\left(R_S|\boldsymbol{X}\right) = c$, a constant that is independent of $\boldsymbol{X}$;

reference distribution the empirical distribution of R_S approximates an explicit distribution that is related to the link function $G^{-1}\left(\cdot\right)$. In particular, independent of $\boldsymbol{X}$, $R_S \sim G\left(c + \int u dG(u)\right)$, where c is a constant.

According to the last property, if $\int u dG\left(u\right) = 0$, then $R_S \sim G\left(\cdot\right)$. These properties allow for a thorough examination of the residuals to check model adequacy and misspecification of the mean structure and link function.

Jittering for general models

The latent method discussed in Section 2.2 applies to cumulative link models for ordinal outcomes. For more general models, we can define a surrogate using a technique called *jittering*. Suppose the true model for an ordinal outcome $\mathcal{Y}$

$$\mathcal{Y} \sim F_a\left(y; \boldsymbol{X}, \boldsymbol{\beta}\right), \tag{5}$$

where $F\left(\cdot\right)$ is a discrete cumulative distribution function. This model is general enough to cover the cumulative link models (1) and (2), and nearly any parametric or nonparametric model for ordinal outcomes.

Liu and Zhang (2017) suggest defining the surrogate S using either of the following two approaches:

1. jittering on the outcome scale: $S|\mathcal{Y} = y \sim \mathcal{U}\left[y, y+1\right]$;
2. jittering on the probability scale: $S|\mathcal{Y} = y \sim \mathcal{U}\left[F_a\left(y-1\right), F_a\left(y\right)\right]$.

Once a surrogate is obtained, we define the surrogate residuals in the same way as Equation (4). In either case, if the hypothesized model is correct, then symmetry around zero still holds; that is $E\left(R_S|\boldsymbol{X}\right) = 0$. For the latter case, if the hypothesized model is correct then $R_S|\boldsymbol{X} \sim \mathcal{U}\left(-1/2, 1/2\right)$. In other words, jittering on the probability scale has the additional property that the conditional distribution of R_S given $\boldsymbol{X}$ has an explicit form. This allows for a full examination of the distributional information of the residual.

Bootstrapping

Since the surrogate residuals are based on random sampling, additional variability is introduced. One way to account for this sample variability and help stabilize any patterns in diagnostic plots is to use the bootstrap (Efron, 1979).

The procedure for bootstrapping surrogate residuals is similar to the model-based bootstrap algorithm used in linear regression. To obtain the b-th bootstrap replicate of the residuals, Liu and Zhang (2017) suggest the following algorithm:

Step 1 Perform a standard case-wise bootstrap of the original data to obtain the bootstrap sample $\left\{\left(\boldsymbol{X}_{1b}^\star, \mathcal{Y}_{1b}^\star\right), \dots, \left(\boldsymbol{X}_{nk}^\star, \mathcal{Y}_{nk}^\star\right)\right\}$.

Step 2 Using the procedure outlined in the previous section, obtain a sample of surrogate residuals $R_{S_{1b}}^\star, \dots, R_{S_{nb}}^\star$ using the bootstrap sample obtained in **Step 1**.

This procedure is repeated a total of B times. For residual vs. covariate (i.e., R vs. X) plots and residual vs. fitted value (i.e., R vs. $f\left(\boldsymbol{X}, \boldsymbol{\beta}\right)$) plots, we simply scatter all $B \times n$ residuals on the same plot. This approach is valid since the bootstrap samples are drawn independently. For large data sets, we find it useful to lower the opacity of the data points to help alleviate any issues with overplotting. For Q-Q plots, on the other hand, Liu and Zhang (2017) suggest using the median of the B bootstrap distributions, which is the implementation used in the **sure** package (Greenwell et al., 2017).

Surrogate residuals in R

The **sure** package supports a variety of R packages for fitting cumulative link and other types of models. The supported packages and their corresponding functions are described in Table 2.

The **sure** package currently exports four functions:

- `resids`—for constructing surrogate residuals;
- `surrogate`—for generating the surrogate response values used in the residuals;
- `autoplot`—produce various diagnostic plots using **ggplot2** graphics (Wickham, 2009);
- `gof`—simulate p-values from a goodness-of-fit test.

In addition, the package also includes five simulated data sets: `df1`, `df2`, `df3`, `df4`, and `df5`. These data sets are used throughout the paper to demonstrate how the surrogate residual can be useful as a diagnostic tool for ordinal regression models. The R code used to generate these data sets is available on the projects GitHub page: `https://github.com/AFIT-R/sure/blob/master/data-raw/data.R`.

Package	Function(s)	Model	Parameterization
stats	`glm`	binary regression	NA
MASS	`polr`	cumulative link	$Pr\{\mathcal{Y} \leq j\}$
rms	`lrm`	cumulative link	$Pr\{\mathcal{Y} \geq j\}$
	`lrm`	logistic regression	NA
	`orm`	cumulative link	$Pr\{\mathcal{Y} \geq j\}$
ordinal	`clm`	cumulative link	$Pr\{\mathcal{Y} \leq j\}$
VGAM	`vglm`	cumulative link	$Pr\{\mathcal{Y} \leq j\}$
	`vgam`	cumulative link	$Pr\{\mathcal{Y} \leq j\}$

Table 2: Ordinal regression modelling packages supported by **sure** and the corresponding parameterization they use for fitting cumulative link models. **Note:** by default, `vglm` uses the same parameterization as in Equation (1). This can be reversed by setting `reverse = TRUE` in the `family` argument.

Detecting a misspecified mean structure

For illustration, the data frame `df1` contains $n = 2000$ observations from the following cumulative link model:

$$Pr\{\mathcal{Y} \leq j\} = \Phi\left(\alpha_j + \beta_1 X + \beta_2 X^2\right), \quad j = 1, 2, 3, 4, \tag{6}$$

where $\alpha_1 = -16$, $\alpha_2 = -12$, $\alpha_3 = -8$, $\beta_1 = 8$, $\beta_2 = -1$, and $X \sim \mathcal{U}(1, 7)$. These simulated data are available in the `df1` data frame from the **sure** package and are loaded automatically with the package; see `?df1` for details. Below, we fit a (correctly specified) probit model using the `polr` function from the **MASS** package.

```
# Fit a cumulative link model with probit link
library(sure)  # for residual function and sample data sets
library(MASS)  # for polr function
fit.polr <- polr(y ~ x + I(x ^ 2), data = df1, method = "probit")
```

The code chunk below obtains the SBS residuals (3) from the previously fitted probit model `fit.polr` using the **PResiduals** package and constructs a couple of diagnostic plots. The results are displayed in Figure 1.

```
# Obtain the SBS/probability-scale residuals
library(PResiduals)
pres <- presid(fit.polr)

# Residual vs. covariate plot and Q-Q plot
library(ggplot2)  # for plotting
p1 <- ggplot(data.frame(x = df1$x, y = pres), aes(x, y)) +
  geom_point(color = "#444444", shape = 19, size = 2, alpha = 0.5) +
  geom_smooth(color = "red", se = FALSE) +
  ylab("Probability-scale residual")
p2 <- ggplot(data.frame(y = pres), aes(sample = y)) +
  stat_qq(distribution = qunif, dparams = list(min = -1, max = 1), alpha = 0.5) +
  xlab("Sample quantile") +
  ylab("Theoretical quantile")
grid.arrange(p1, p2, ncol = 2)  # Figure 1
```

(**Note:** the reference distribution for the SBS residual is the $\mathcal{U}(-1, 1)$ distribution.) As can be seen in Figure 1, the SBS residuals, which are inherently discrete, often display unusual patterns in diagnostic plots, making them less useful as a diagnostic tool. There is a pattern for each of the $J = 4$ classes!

Similarly, we can use the `resids` function in package **sure** to obtain the surrogate residuals discussed in Section 2.2. This is illustrated in the following code chunk. the results are displayed in Figure 2. (**Note:** since the surrogate residuals are based on random sampling, we specify the seed via the `set.seed` function throughout this paper for reproducibility.)

```
# Obtain surrogate residuals
library(sure)
set.seed(101)  # for reproducibility
sres <- resids(fit.polr)
```

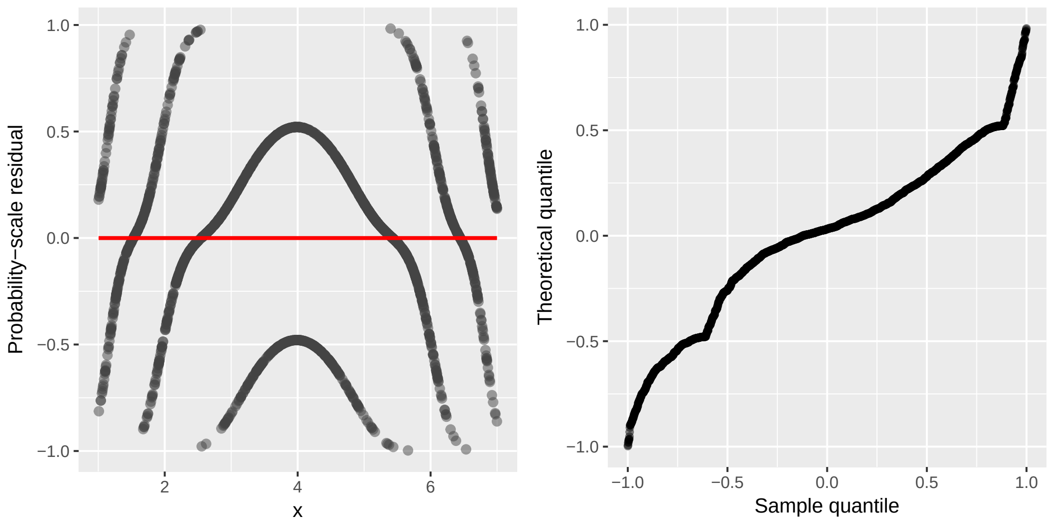

Figure 1: SBS residual plots for the (correctly specified) probit model fit to the `df1` data set. *Left*: Residual vs. covariate plot with a nonparametric smooth (red curve). *Right*: Q-Q plot of the residuals.

```
# Residual vs. covariate plot and Q-Q plot
library(ggplot2)  # needed for autoplot function
p1 <- autoplot(sres, what = "covariate", x = df1$x, xlab = "x")
p2 <- autoplot(sres, what = "qq", distribution = qnorm)
grid.arrange(p1, p2, ncol = 2)  # Figure 2
```

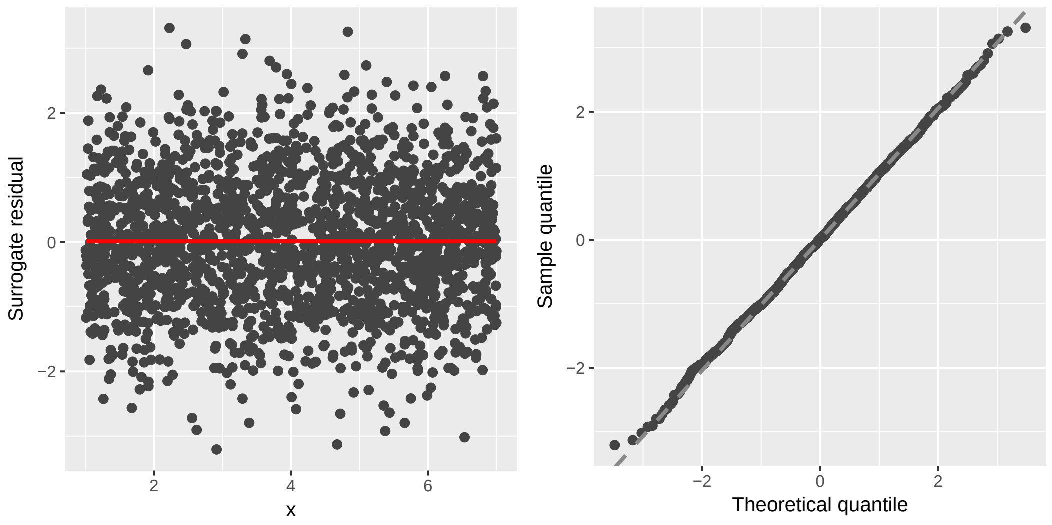

Figure 2: surrogate residual plots for the (correctly specified) probit model fit to the `df1` data set. *Left*: Residual vs. covariate plot with a nonparametric smooth (red curve). *Right*: Q-Q plot of the residuals.

The **sure** package also includes `autoplot` methods for the various classes of models listed in Table 2, so you can just give `autoplot` the fitted model directly. The benefit of this approach is that the fitted values and reference distribution (used in Q-Q plots) are automatically extracted. For example, to reproduce the Q-Q plot in Figure 2, we could have just used

```
set.seed(101)  # for reproducibility
autoplot(fit.polr, what = "qq")  # same as top right of Figure 1
```

Suppose that we did not include the quadratic term in our fitted model. We would expect a residual-vs-*x* plot to indicate that such a quadratic term is missing. Below we update the previously fitted model by removing the quadratic term, then update the residual-vs-covariate plots (code not shown). The updated residual plots are displayed in Figure 3.

```
fit.polr <- update(fit.polr, y ~ x)  # remove quadratic term
```

The SBS residuals gives some indication of a misspecified mean structure, but this only becomes more clear with increasing J, and the plot is still discrete. This is overcome by the surrogate residuals which produces a residual plot not unlike those seen in ordinary linear regression models.

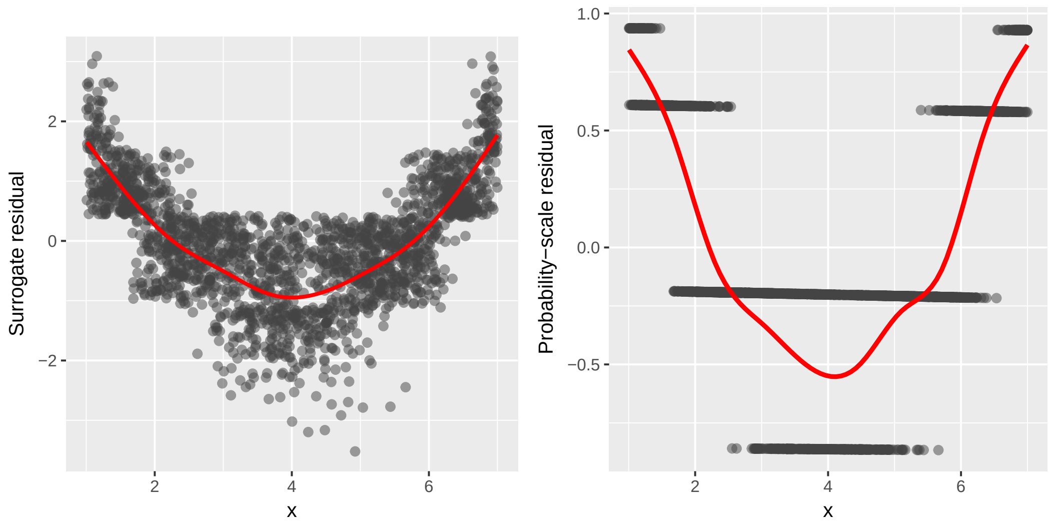

Figure 3: Residual-vs-covariate plots with nonparametric smooths (red curves) for a probit model with a misspecified mean structure fit to the simulated data from model (6). *Left*: Surrogate residuals. *Right*: SBS residuals.

Detecting heteroscedasticty

One issue that often raises concerns in statistical inference is that of heteroscedasticity; that is, when the error term has non constant variance. Heteroscedasticity can bias the statistical inference and lead to improper standard errors, confidence intervals, and p-values. Therefore, it is imperative to identify heteroscedacticity whenever present and take appropriate action (e.g., transformations, etc.). In ordinary linear regression, this topic has been covered extensively. For categorical models, on the other hand, not much has been proposed in the literature.

As discussed in Section 2.2, one of the properties of the surrogate residual R_S is that, if the model is specified correctly, then $Var\left(R_S|X\right) = c$, where c is a constant.

For this example, we generated $n = 2000$ observations from the following ordered probit model:

$$Pr\left\{\mathcal{Y} \le j\right\} = \Phi\left\{\left(\alpha_j + \beta X\right)/\sigma_X\right\}, \quad j = 1,2,3,4,5,$$

where $\alpha_1 = -36$, $\alpha_2 = -6$, $\alpha_3 = 34$, $\alpha_4 = 64$, $\beta = -4$, $X \sim \mathcal{U}\left(2,7\right)$, and $\sigma_X = X^2$. Notice how the variability is an increasing function of X. These data are available in the df2 data frame that is automatically loaded with the **sure** package; see ?df2 for details.

The following block of code uses the orm function from the popular **rms** package to fit a probit model to the simulated data. **Note:** we had to set x = TRUE in the call to orm in order to use the presid function later.

```
# Fit a cumulative link model with probit link
library(rms)  # for orm function
fit.orm <- orm(y ~ x, data = df2, family = "probit", x = TRUE)
```

If heteroscedasticity is present, we would expect this to show up in various diagnostic plots, such as a residual vs. covariate plot. Below we obtain the SBS and surrogate residuals as before and plot them against X. The results are displayed in Figure 4.

```
set.seed(102)  # for reproducibility
p1 <- autoplot(resids(fit.orm), what = "covariate", x = df2$x, xlab = "x")
p2 <- ggplot(data.frame(x = df2$x, y = presid(fit.orm)), aes(x, y)) +
  geom_point(color = "#444444", shape = 19, size = 2, alpha = 0.25) +
  geom_smooth(col = "red", se = FALSE) +
  ylab("Probability scale residual")
grid.arrange(p1, p2, ncol = 2)  # Figure 4
```

In Figure 4, it is clear from the plot of the surrogate residuals (left side of Figure 4) that the variance increases with X, a sign of heteroscedasticity. As a matter of fact, the plot suggests that the true link function has a varying scale parameter, $\sigma = \sigma(\boldsymbol{X})$. The plot of the SBS residuals (right side of Figure 4), on the other hand, gives no indication of an issue with nonconstant variance.

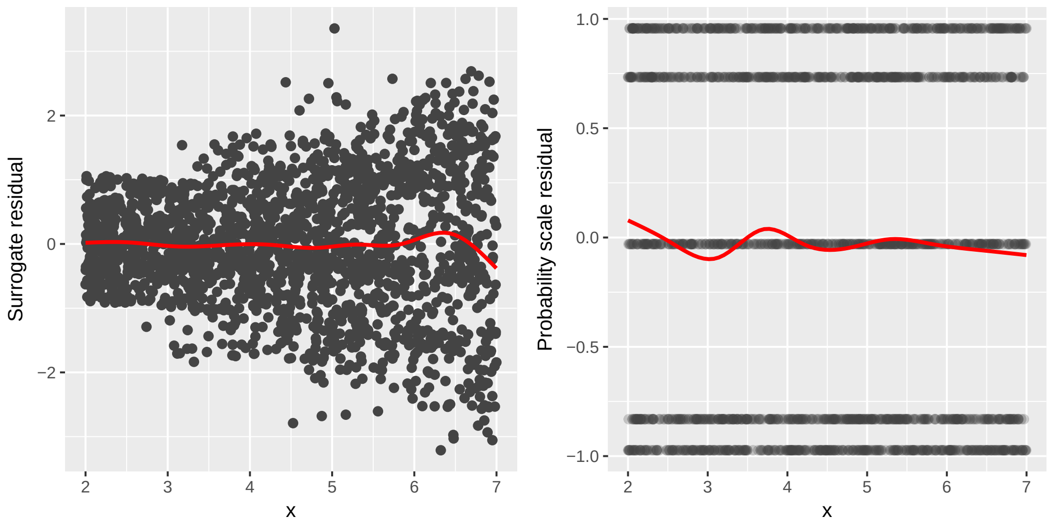

Figure 4: Residual vs. covariate plots with nonparametric smooths (red curves) for the simulated heteroscedastic data. *Left*: Surrogate residuals. *Right*: SBS residuals.

As outlined in Section 2.2.1, the jittering technique is broadly applicable to virtually all parametric and nonparametric models for ordinal responses. To illustrate, the code chunk below uses the **VGAM** package to fit a *vector generalized additive* model to the same data using a nonparametric smooth for x.

```
library(VGAM)  # for vgam and vglm functions
fit.vgam <- vgam(y ~ s(x), family = cumulative(link = probit, parallel = TRUE),
                 data = df2)
```

To obtain a surrogate residual using the jittering technique, we can set `method = "jitter"` in the call to `resids` or `autoplot`. There is also the option `jitter.scale` which can be set to either `"probability"`, for jittering on the probability scale (the default), or `"response"`, for jittering on the response scale. In the code chunk below, we use the `autoplot` function to obtain residual-by-covariate plots using both types of jittering. The results, which are displayed in Figure 5, clearly indicate that the variance increases with increasing x.

```
set.seed(103)  # for reproducibility
p1 <- autoplot(fit.vgam, what = "covariate", x = df2$x, method = "jitter",
               xlab = "x")
p2 <- autoplot(fit.vgam, what = "covariate", x = df2$x, method = "jitter",
               jitter.scale = "response", xlab = "x")
grid.arrange(p1, p2, ncol = 2)  # Figure 5
```

Detecting a misspecified link function

For this example, we simulated $n = 2000$ observations from the following model

$$Pr(\mathcal{Y} \le j) = \Phi\left(\alpha_j + \beta_1 X + \beta_2 X^2\right), \quad j = 1, 2, 3, 4$$

The data are available in the data frame `df3` within the package; see `?df3` for details.

Below we fit a model with various link functions. For this model, however, the correct link function to use is the log-log link. From these models, we construct Q-Q plots of the residuals using $R = 100$ bootstrap replicates. The results are displayed in Figure 6.

```
# Fit models with various link functions to the simulated data
fit.probit <- polr(y ~ x + I(x ^ 2), data = df3, method = "probit")
fit.logistic <- polr(y ~ x + I(x ^ 2), data = df3, method = "logistic")
fit.loglog <- polr(y ~ x + I(x ^ 2), data = df3, method = "loglog")  # correct link
```

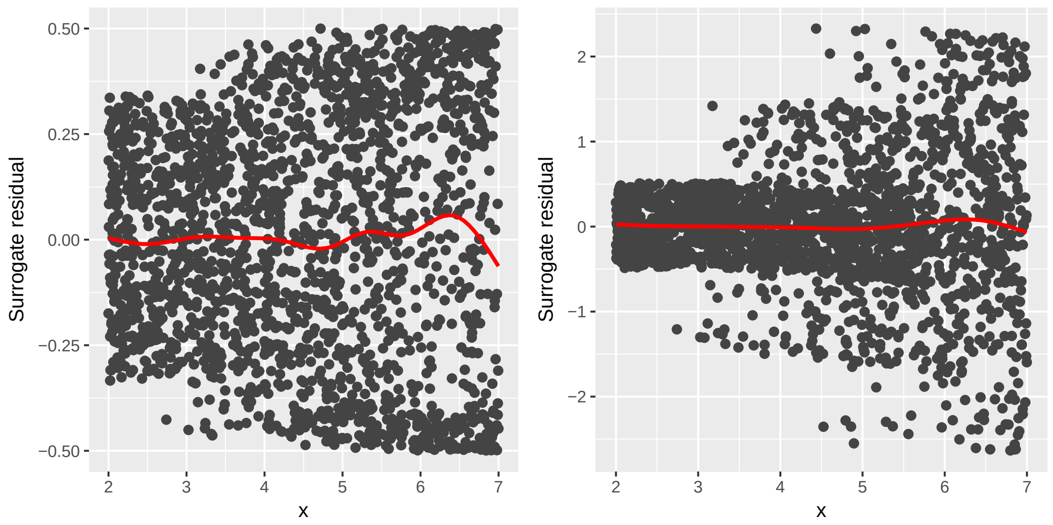

Figure 5: Residual vs. covariate plots with nonparametric smooths (red curves) from a vector generalized additive model fit to the simulated heteroscedastic data. *Left*: Jittering on the probability scale (default). *Right*: Jittering on the response scale.

```
fit.cloglog <- polr(y ~ x + I(x ^ 2), data = df3, method = "cloglog")

# Construct Q-Q plots of the surrogate residuals for each model
set.seed(1056)  # for reproducibility
p1 <- autoplot(fit.probit, nsim = 100, what = "qq")
p2 <- autoplot(fit.logistic, nsim = 100, what = "qq")
p3 <- autoplot(fit.loglog, nsim = 100, what = "qq")
p4 <-  autoplot(fit.cloglog, nsim = 100, what = "qq")

# Figure 6
grid.arrange(p1, p2, p3, p4, ncol = 2)  # bottom left plot is correct model
```

From the Q-Q plots in Figure 6, it is clear the the model with the log-log link (which corresponds to gumbel errors in the latent variable formulation) is the most appropriate, while the other plots indicate deviations from the hypothesized model.

Alternatively, we could also use the surrogate residuals to make use of existing distance-based goodness-of-fit (GOF) tests; for example, the Kolmogorov-Smirnov distance. The gof function in **sure** can be used to produce simulated p-values from such tests.

Currently, the gof function supports three goodness-of-fit tests: the Kolmogorov-Smirnov test (test = "ks"), the Anderson-Darling test (test = "ad"), and the Cramer-Von Mises test (test = "cvm"). Below, we use the gof function to simulate p-values from the Anderson-Darling test for each of the four models; we also set nsim to 100 to produce smoother plots and reduce the sampling error induced by the surrogate procedure. The plot method is then used to display the empirical distribution function (EDF) of the simulated p-values. A good fit would imply uniformly distributed p-values; hence, the EDF would be relatively straight with a slope of one. The results, which are displayed in Figure 7, agree with the Q-Q plots from Figure 6 in that the log-log link is the most appropriate for these data. (**Note:** the plotting method for "gof" objects uses base R graphics; hence, we can use the par function to set various graphical parameters.)

```
# Figure 7
par(mfrow = c(2, 2), mar = c(2, 4, 2, 2) + 0.1)
set.seed(8491)  # for reproducibility
plot(gof(fit.probit, nsim = 100, test = "ad"), main = "")
plot(gof(fit.logistic, nsim = 100, test = "ad"), main = "")
plot(gof(fit.loglog, nsim = 100, test = "ad"), main = "")
plot(gof(fit.cloglog, nsim = 100, test = "ad"), main = "")
```

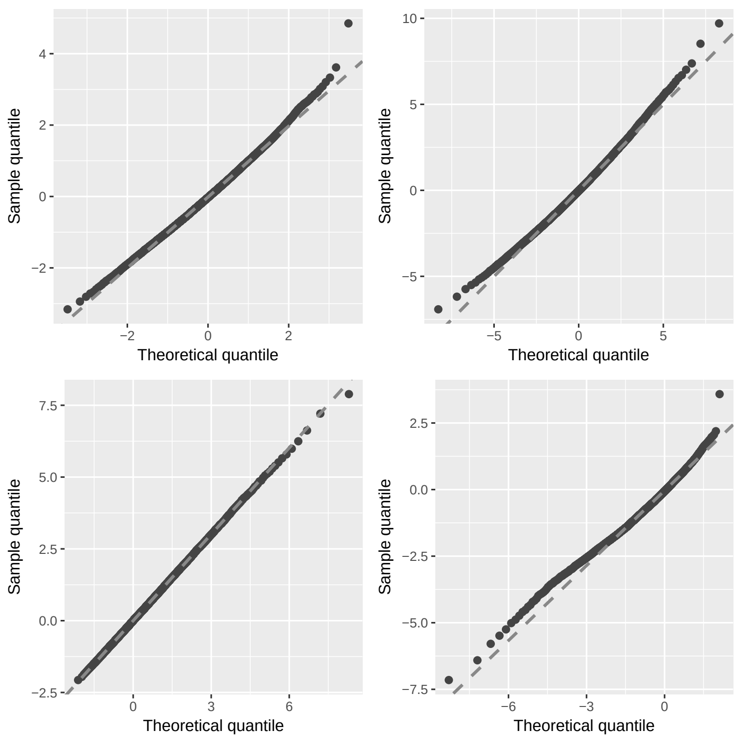

Figure 6: Q-Q plots of the residuals for various cumulative link models fit to simulated data with gumbel errors. *Top left*: A model with probit link. *Top right*: A model with logit link. *Bottom left*: A model with log-log link (i.e., the correct model). *Bottom right*: A model with complementary log-log link.

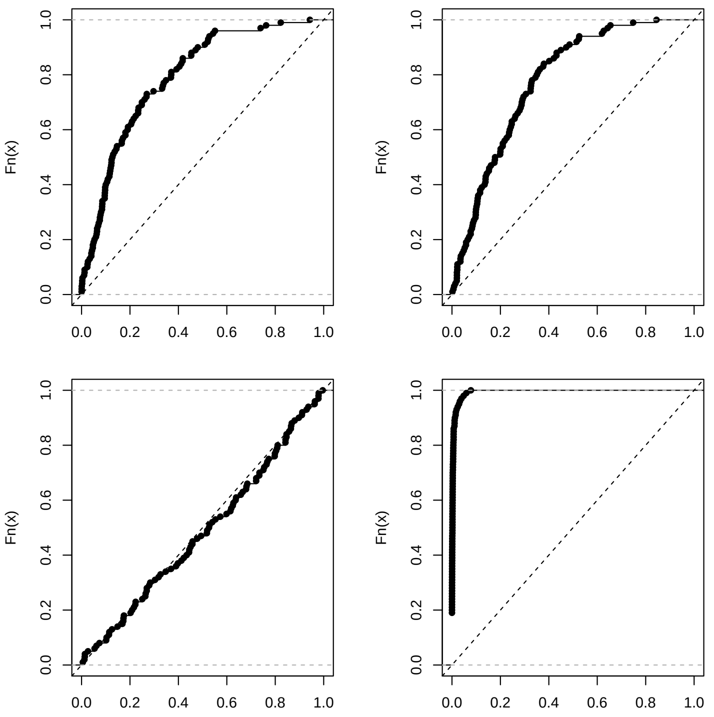

Figure 7: EDFs of the simulated p-values from an Anderson-Darling GOF test for various cumulative link models fit to simulated data with gumbel errors. *Top left*: A model with probit link. *Top right*: A model with logit link. *Bottom left*: A model with log-log link (i.e., the correct model). *Bottom right*: A model with complementary log-log link.

Checking the proportionality assumption

An important feature of the cumulative link model (1) is the proportional odds assumption, which assumes that the mean structure, $f(\boldsymbol{X}, \boldsymbol{\beta})$, remains the same for each of the J categories; for the logit case (see row one of Table 1), this is also referred to as the proportional odds assumption. Harrell (2001, pp. 334–335) suggests computing each observation's contribution to the first derivative of the log likelihood function with respect to $\boldsymbol{\beta}$, averaging them within each of the J categories, and examining any trends in the residual plots, but these plots can be difficult to interpret. Fortunately, it is relatively straightforward to use the simulated surrogate response values S to check the proportionality assumption.

To illustrate, we generated 2000 observations from each of the following probit models

$$Pr(\mathcal{Y} \le j) = \Phi\left(\alpha_j + \beta_1 x\right), \quad j = 1, 2, 3, \quad \text{and} \quad Pr(\mathcal{Y} \le j) = \Phi\left(\alpha_j + \beta_2 x\right), \quad j = 4, 5, 6,$$

where $\alpha_1 = -1.5$, $\alpha_2 = 0$, $\alpha_3 = 1$, $\alpha_4 = 3$, $\beta_1 = 1$, and $\beta_2 = 1.5$. The data are available in the data frame `df4` within the package; see `?df4` for details.

Checking the proportionality assumption here amounts to checking whether or not $\beta_1 - \beta_2 = 0$. As outlined in Liu and Zhang (2017), we can generate surrogates $S_1 \sim \mathcal{N}(-\beta_1 x, 1)$ and $S_s \sim \mathcal{N}(-\beta_2 x, 1)$, both conditional on x. We then define the difference $D = S_2 - S_1$ which, conditional on x, follows a $\mathcal{N}((\beta_1 - \beta_2)x, 1)$ distribution. If $\beta_1 - \beta_2 = 0$, then D should be independent of x. This can be easily checked by plotting D against x. Below, we use the `surrogate` function to generate the surrogate response values directly (as opposed to the residuals) and generate the D vs. x plot shown in Figure 8.

```
# Fit separate models (VGAM should already be loaded)
fit1 <- vglm(y ~ x, data = df4[1:2000, ],
             cumulative(link = probit, parallel = TRUE))
fit2 <- update(fit1, data = df4[2001:4000, ])

# Generate surrogate response values
set.seed(8671)  # for reproducibility
s1 <- surrogate(fit1)
s2 <- surrogate(fit2)

# Figure 8
ggplot(data.frame(D = s1 - s2, x = df4[1:2000, ]$x) , aes(x = x, y = D)) +
  geom_point(color = "#444444", shape = 19, size = 2) +
  geom_smooth(se = FALSE, size = 1.2, color = "red")
```

From Figure 8, it is clear that $\beta_1 - \beta_2 \neq 0$; hence, the proportionality assumption does not hold.

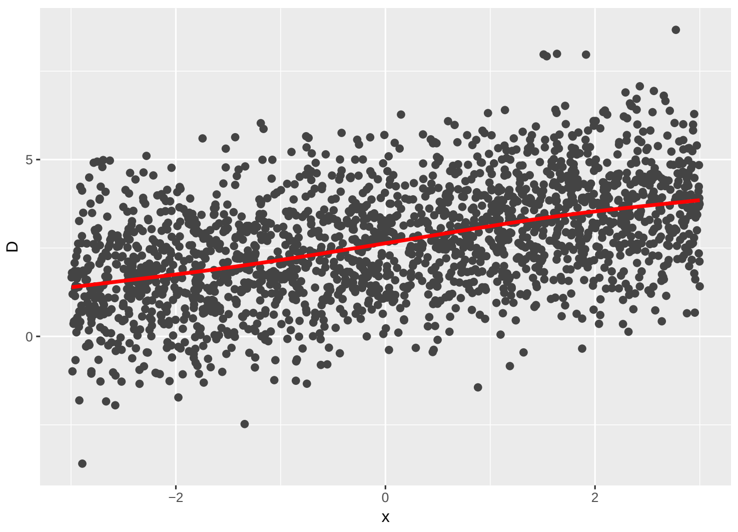

Figure 8: Scatterplot of $D = S_1 - S_2$ vs. x with a nonparametric smooth (red curve).

Detecting interaction effects

A common challenge in model building is determining whether or not there are important interactions between the predictors in the data. Using the surrogate residuals, it is rather straightforward to determine if such an interaction effect is missing from the assumed model.

For illustration, we generated $n = 2000$ observations from the following ordered probit model

$$Pr\left(\mathcal{Y} \leq j\right) = \Phi\left(\alpha_j + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2\right), \quad j = 1, 2, 3, 4,$$

where $\alpha_1 = -16$, $\alpha_2 = -12$, $\alpha_3 = -8$, $\beta_1 = 8$, $\beta_2 = 3$, $\beta_3 = 5$, $x_1 \sim \mathcal{U}\left(1, 7\right)$, and x_2 is a factor with levels `Treatment` and `Control`. The simulated data are available in the `df5` data frame loaded with the **sure** package; see `?df5` for details. Below, we fit two probit models using the `clm` function from the **ordinal** package (which should already be loaded). The first model only corresponds to the simulated control group, while the second model corresponds to the treatment group.

```
library(ordinal)  # for clm function
fit1 <- clm(y ~ x1, data = df5[df5$x2 == "Control", ], link = "probit")
fit2 <- clm(y ~ x1, data = df5[df5$x2 == "Treatment", ], link = "probit")
```

If the true model contains an interaction term $x_1 x_2$, but the fitted model does not include it, we can detect this misspecification using the surrogate residuals. We simply plot R_S versus x_1 for treatment group, and compare it to the plot of R_S versus x_1 for the controls—or better yet, we can just use the surrogate response S instead of R_S. The trends in these two plots should be different. Below, we use `ggplot` along with the `sure` function to construct such a plot for both fitted models. (**Note:** we used $B = 25$ bootstrap simulations to reduce the sample variability in the plots.) The results are displayed in Figure 9. The plot indicates a negative association between x_1 and the outcome within the control group, and a positive association between x_1 and the treatment group (i.e., an interaction between x_1 and x_2).

```
set.seed(1105)  # for reproducibility
d1 <- cbind(df5[df5$x2 == "Control",], sur = surrogate(fit1, nsim = 25))
d2 <- cbind(df5[df5$x2 == "Treatment", ], sur = surrogate(fit2, nsim = 25))
p1 <- ggplot(d1, aes(x = x1, y = sur)) +
  geom_point(color = "#444444", shape = 19, size = 2, alpha = 0.5) +
  geom_smooth(se = FALSE, size = 1.2, color = "red") +
  ylab("Surrogate response") +
  xlab(expression(paste(x[1], " (control)")))
p2 <- ggplot(d2, aes(x = x1, y = sur)) +
  geom_point(color = "#444444", shape = 19, size = 2, alpha = 0.5) +
  geom_smooth(se = FALSE, size = 1.2, color = "red") +
  ylab("Surrogate response") +
  xlab(expression(paste(x[1], " (treatment)")))
grid.arrange(p1, p2, ncol = 2)  # Figure 9
```

Bitterness of wine

Randall (1989) performed an experiment on factors determining the bitterness of wine. Two binary treatment factors, temperature and contact (between juice and skin), were controlled while crushing the grapes during wine production. Nine judges each assessed wine from two bottles from each of the four treatment conditions; for a total of $n = 72$. The data are available in the **ordinal** package; see `?ordinal::wine` for details.

```
data(wine, package = "ordinal")  # load wine data set
wine.clm <- clm(rating ~ temp + contact, data = wine, link = "probit")
```

Since both of the covariates in this model are binary factors, scatterplots are not appropriate for displaying the residual-by-covariate relationships. Instead, the `autoplot` function in `sure` uses boxplots; a future release is likely to include the additional option for producing nonparametric densities for each level of a factor. The code chunk below uses `autoplot` along with `grid.arrange` to produce some standard residual diagnostic plots. The results are displayed in Figure 10. The Q-Q plot and residual-vs-fitted value plot do not indicate any serious model misspecifications. Furthermore, the boxplots reveal that the medians of the surrogate residuals are very close to zero, and the distribution of the residuals within each level appear to be symmetric and have approximately the same variability (with the exception of a few outliers).

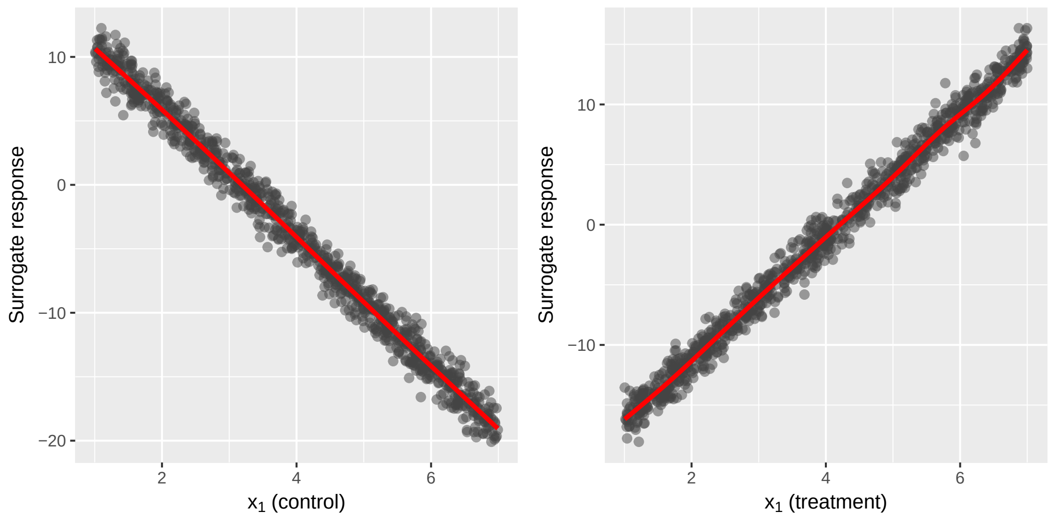

Figure 9: Scatterplot of the surrogate response S versus x_1 with a nonparametric smooth (red line). *Left*: Control group. *Right*: Treatment group.

```
set.seed(1225)  # for reproducibility
grid.arrange(  # Figure 10
  autoplot(wine.clm, nsim = 10, what = "qq"),
  autoplot(wine.clm, nsim = 10, what = "fitted", alpha = 0.5),
  autoplot(wine.clm, nsim = 10, what = "covariate", x = wine$temp,
           xlab = "Temperature"),
  autoplot(wine.clm, nsim = 10, what = "covariate", x = wine$contact,
           xlab = "Contact"),
  ncol = 2
)
```

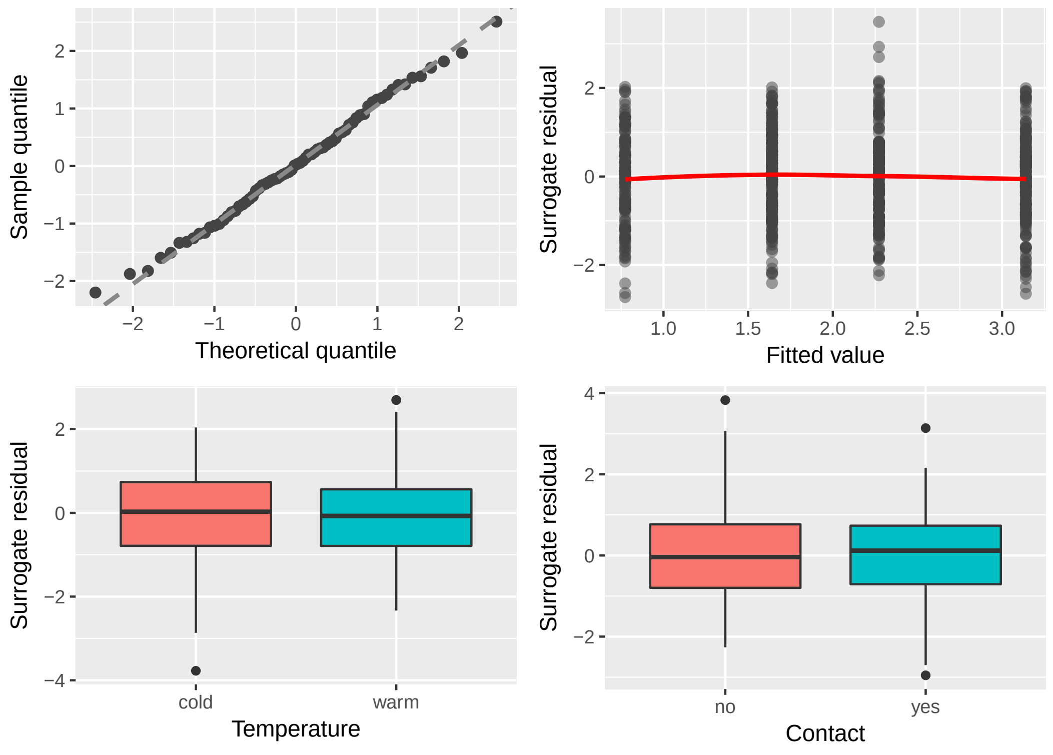

Figure 10: Residual diagnostic plots for the quality of wine example.

Summary

In this paper, we have introduced the **sure** package which implements the surrogate approach to residuals for ordinal regression models described in Liu and Zhang (2017). Using simulated data sets, we have illustrated the various properties of these residuals and how they can be used to check various assumptions in the ordinal regression model (e.g., heteroscedasticity, misspecified link functions, etc.) using continuous residuals which produce diagnostic plots not unlike those seen in ordinary linear regression. This offers a new way of performing typical diagnostic checks for ordinal regression models that are easily interpreted by the analyst. Furthermore, this new approach to constructing residuals for ordinal regression models is still under active development and new functionality will be added to **sure** accordingly in the future.

Acknowledgments

TBD.

Bibliography

R. H. B. Christensen. *ordinal—Regression Models for Ordinal Data*, 2015. URL http://www.cran.r-project.org/package=ordinal. R package version 2015.6-28. [p2]

C. Dupont, J. Horner, C. Li, Q. Liu, and B. Shepherd. *PResiduals: Probability-Scale Residuals and Residual Correlations*, 2016. URL https://CRAN.R-project.org/package=PResiduals. R package version 0.2-4. [p2]

B. Efron. Bootstrap methods: Another look at the jackknife. *Annals fo Statistics*, 7(1):1–26, 1979. URL http://dx.doi.org/10.1214/aos/1176344552. [p3]

B. Greenwell, A. McCarthy, and B. Boehmke. *sure: Surrogate Residuals for Ordinal and General Regression Models*, 2017. URL https://CRAN.R-project.org/package=sure. R package version 0.1.2.9000. [p3]

F. E. Harrell. *Regression Modeling Strategies: With Applications to Linear Models, Logistic Regression, and Survival Analysis*. Graduate Texts in Mathematics. Springer, 2001. ISBN 9780387952321. [p10]

F. E. Harrell Jr. *rms: Regression Modeling Strategies*, 2017. URL https://CRAN.R-project.org/package=rms. R package version 5.1-1. [p2]

C. Li and B. E. Shepherd. A new residual for ordinal outcomes. *Biometrika*, 99(2):473–480, 2012. URL http://dx.doi.org/10.1093/biomet/asr073. [p2]

D. Liu and H. Zhang. Residuals and diagnostics for ordinal regression models: A surrogate approach. *Journal of the American Statistical Association*, X(Y):XX–YY, 2017. URL http://dx.doi.org/10.1080/01621459.2017.1292915. [p2, 3, 10, 13]

I. Liu, B. Mukherjee, T. Suesse, D. Sparrow, and S. K. Park. Graphical diagnostics to check model misspecification for the proportional odds regression model. *Statistics in Medicine*, 28(3):412–429, 2009. URL http://dx.doi.org/10.1080/01621459.2017.1292915. [p2]

J. H. Randall. The analysis of sensory data by generalized linear model. *Biometrical Journal*, 31(7): 781–793, 1989. URL http://dx.doi.org/10.1002/bimj.4710310703. [p11]

W. N. Venables and B. D. Ripley. *Modern Applied Statistics with S*. Springer, New York, fourth edition, 2002. URL http://www.stats.ox.ac.uk/pub/MASS4. ISBN 0-387-95457-0. [p2]

H. Wickham. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2009. ISBN 978-0-387-98140-6. URL http://ggplot2.org. [p3]

T. W. Yee. *VGAM: Vector Generalized Linear and Additive Models*, 2017. URL https://CRAN.R-project.org/package=VGAM. R package version 1.0-3. [p2]

Brandon M. Greenwell
Illumination Works
6289 Commons Blvd
Suite 120
Beavercreek, OH 45431

United States of America
`greenwell.brandon@gmail.com`

Andrew McCarthy
The Perduco Group
3610 Pentagon Blvd
Suite 110
Beavercreek, OH 45431
United States of America
`andymc82000@yahoo.com`

Bradley C. Boehmke
Air Force Institute of Technology
2950 Hobson Way
Wright-Patterson AFB, OH 45433
United States of America
`bradleyboehmke@gmail.com`

Dungang Liu
University of Cincinnati
2925 Campus Green Dr
Cincinnati, OH 45221
United States of America
`dungang.liu@uc.edu`