

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 **SURrogate REsiduals**. 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 effects.

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). In other studies, the outcome may be an ordinal variable; 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 modeling 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

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

where $F_a(\cdot)$ is a discrete cumulative distribution function, \mathbf{X} is an $n \times p$ model matrix, and $\boldsymbol{\beta}$ is a $p \times 1$ vector of unknown regression coefficients. This functional form includes binary regression as a special case. For example, the probit model has

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

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

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

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

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

$$G^{-1}(\Pr\{\mathcal{Y} \geq j\}) = \alpha_j^* + f(\mathbf{X}, \boldsymbol{\beta}^*). \tag{2}$$

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); see, for example, [Agresti \(2010\)](#).

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

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

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

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

Common choices for the link function $G^{-1}(\cdot)$ 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 `c1m` function for fitting cumulative link models. The popular **rms** package (Harrell, 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).

The remainder of the paper is organized as follows. In the next section, we discuss the idea of *surrogate residuals* (Liu and Zhang, 2017) and talk about some important properties. Next, we briefly discuss jittering, bootstrapping, and how they apply to the surrogate approach. The Section “*Surrogate residuals in R*” introduces the **sure** package and discusses the various modeling packages it supports. The sections following demonstrate how **sure** can be used to detect misspecified mean structures, heteroscedasticity, misspecified link functions, and interaction effects, as well as check the proportionality assumption. The Section “*Bitterness of wine*” provides a real data analysis example. We end with a closing summary.

Surrogate residuals

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) propose 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) show that the sign-based statistic (SBS)

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

can be used as a residual for ordinal outcomes. Though Li and Shepherd refer to these as *probability-based residuals*, 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) and the **PResiduals** package (Dupont et al., 2016). A limitation with the SBS residuals is that they are based on a discrete outcome and are discrete themselves, which makes them less useful in diagnostic plots.

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

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

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 $\mathcal{Z} = -f(\mathbf{X}, \boldsymbol{\beta}) + \epsilon$ using the interval (α_{y-1}, α_y) . The benefit of the surrogate residual (4) is that it is based on a continuous variable S , such that 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:

- (a) **symmetry around zero** $E(R_S|X) = 0$;
- (b) **homogeneity** $Var(R_S|X) = c$, a constant that is independent of X ;
- (c) **reference distribution** the empirical distribution of R_S approximates an explicit distribution that is related to the link function $G^{-1}(\cdot)$. In particular, independent of X , $R_S \sim G(c + \int udG(u))$, where c is a constant.

According to property (a), if $\int udG(u) = 0$, then $R_S \sim G(\cdot)$. Properties (a)–(c) 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 “Surrogate residuals” 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} is

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

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

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 $E(R_S|X) = 0$. For the latter case, if the hypothesized model is correct then $R_S|X \sim \mathcal{U}(-1/2, 1/2)$. In other words, jittering on the probability scale has the additional property that the conditional distribution of R_S given X has an explicit form which 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 $\{(X_{1b}^*, \mathcal{Y}_{1b}^*), \dots, (X_{nb}^*, \mathcal{Y}_{nb}^*)\}$.

Step 2 Using the procedure outlined in the previous section, obtain a sample of surrogate residuals $R_{S_{1b}}^*, \dots, R_{S_{nb}}^*$ 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- $f(X, \hat{\beta})$) 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;

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 modeling 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.

- `surrogate`—for generating the surrogate response values used in the residuals;
- `autoplot`—for producing various diagnostic plots using `ggplot2` graphics (Wickham, 2009);
- `gof`—for simulating p -values from various goodness-of-fit tests.

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 cumulative link models. The R code used to generate these data sets is available on the projects GitHub page: <https://github.com/koalaverse/sure/blob/master/data-raw/data.R>.

Detecting a misspecified mean structure

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

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

where $\alpha_1 = -16$, $\alpha_2 = -12$, $\alpha_3 = -8$, $\beta_1 = -8$, $\beta_2 = 1$, and $X \sim \mathcal{U}(1, 7)$. These parameters were chosen to ensure that 1) the sample from the latent variable \mathcal{Z} is spread out, rather than clustering in a small interval, and 2) each category of \mathcal{Y} is well represented in the sample; we follow these guidelines throughout the simulated examples. The simulated data for this example 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 $U(-1, 1)$ distribution.) As can be seen in the left side of Figure 1, the SBS residuals are inherently discrete and often display unusual patterns in diagnostic plots, making them less useful as a diagnostic tool.

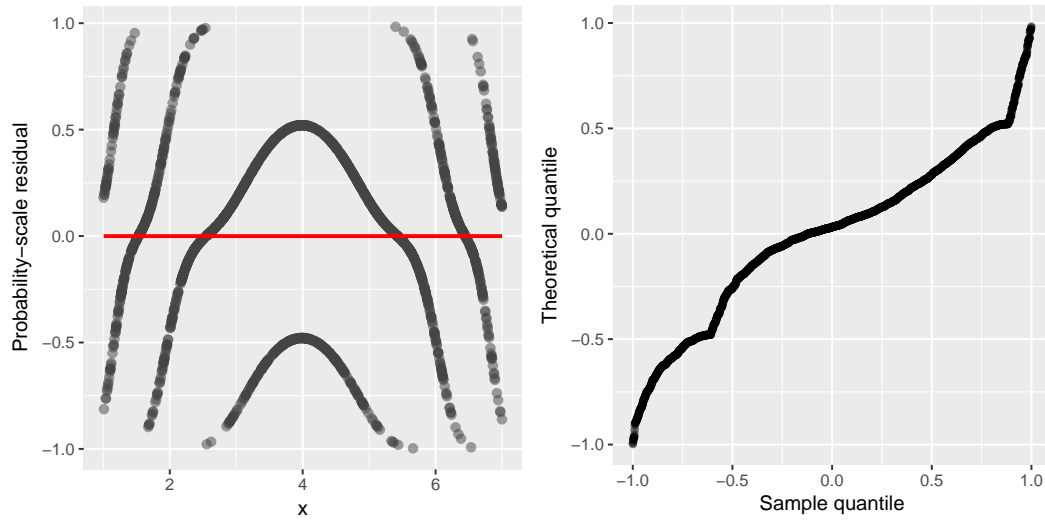


Figure 1: SBS residual plots for the (correctly specified) probit model fit to the df1 data set. *Left:* Residual-vs-covariate plot with a nonparametric smooth (red curve). *Right:* Q-Q plot of the residuals.

Similarly, we can use the `resids` function in package `sure` to obtain the surrogate residuals discussed in Section “Surrogate residuals.” 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)

# 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
```

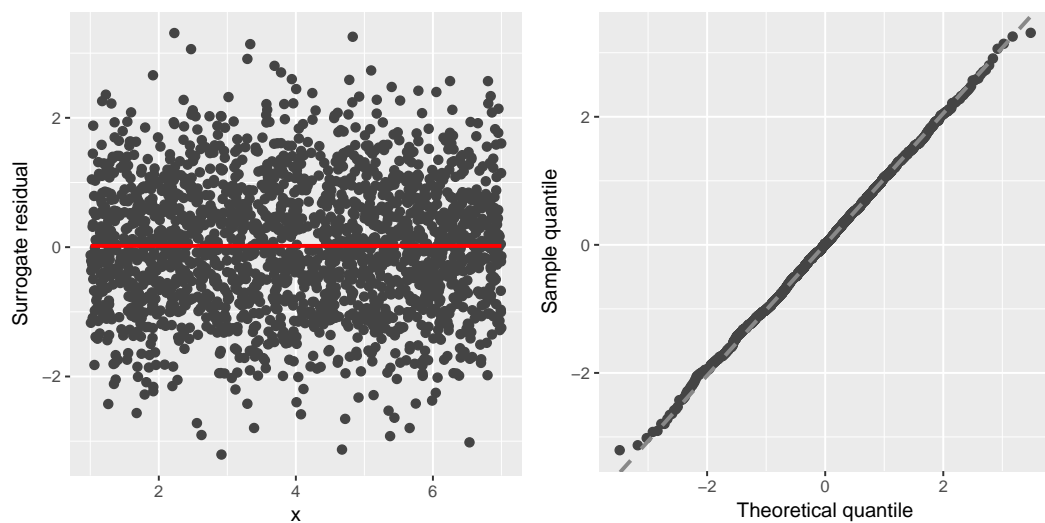


Figure 2: Surrogate residual plots for the (correctly specified) probit model fit to the df1 data set. *Left:* Residual-vs-covariate plot with a nonparametric smooth (red curve). *Right:* Q-Q plot of the residuals.

The `sure` package also includes autoplot methods for the various classes of models listed in Table 2, so that the user can 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 used the following:

```
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.

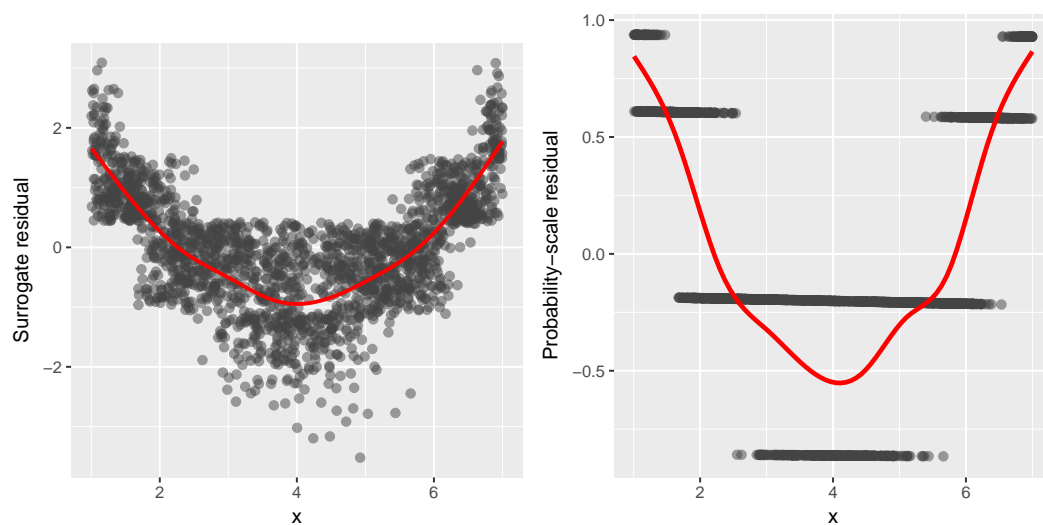


Figure 3: Residual-vs-covariate plots with nonparametric smooths (red curves) for a probit model with a misspecified mean structure fit to the simulated data from model (6). *Left:* Surrogate residuals. *Right:* SBS residuals.

Detecting heteroscedasticity

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 heteroscedasticity whenever present and take appropriate action (e.g., transformations). In ordinary linear regression, this topic has been covered extensively. Not much has been proposed in the literature for categorical models.

As discussed in Section "Surrogate residuals," one of the properties of the surrogate residual R_S is that, if the model is specified correctly, then $\text{Var}(R_S|X) = c$, where c is a constant.

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

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

where $\alpha_1 = -36$, $\alpha_2 = -6$, $\alpha_3 = 34$, $\alpha_4 = 64$, $\beta = -4$, $X \sim \mathcal{U}(2, 7)$, and $\sigma_X = X^2$. Notice how the variability is an increasing function of X . These data are available in the `df2` data frame 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 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, $\sigma = \sigma(X)$. The plot of the SBS residuals (right side of Figure 4), on the other hand, gives no indication of an issue with nonconstant variance.

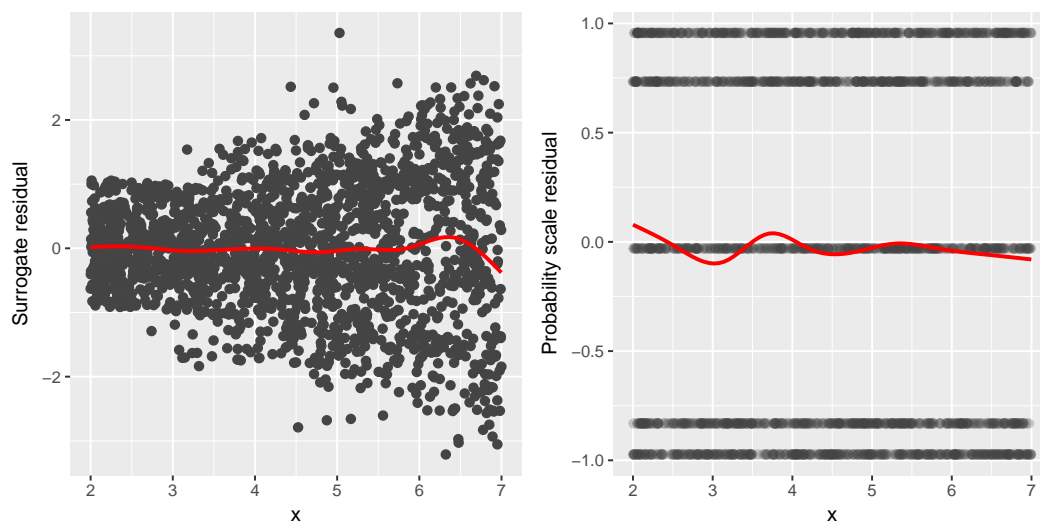


Figure 4: Residual-vs-covariate plots with nonparametric smooths (red curves) for the simulated heteroscedastic data. *Left:* Surrogate residuals. *Right:* SBS residuals.

As outlined in Section “[Jittering for general models](#),” 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 (Figure 5) 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
```

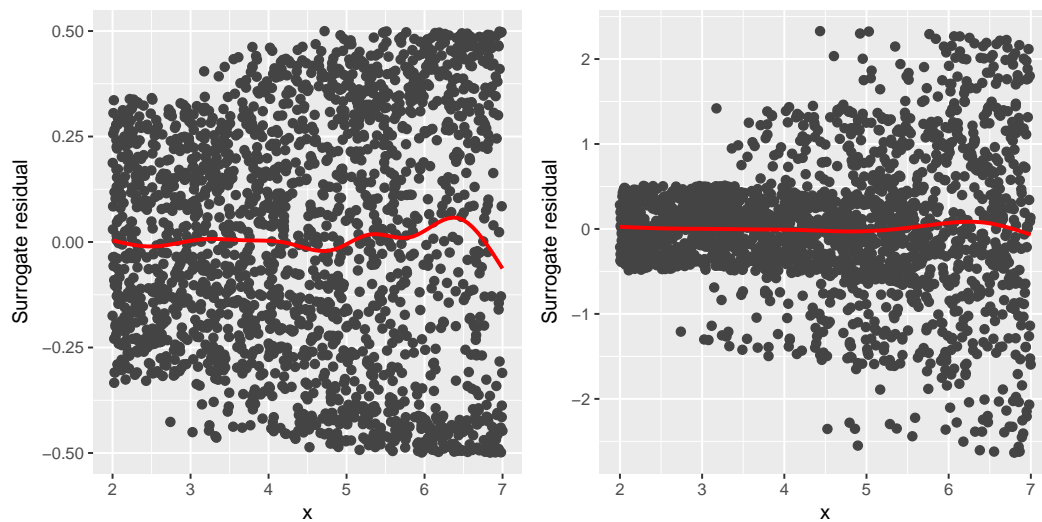


Figure 5: Residual-vs-covariate plots with nonparametric smooths (red curves) from a vector generalized additive model fit to the simulated heteroscedastic data. *Left:* Jittering on the probability scale (default). *Right:* Jittering on the response scale.

Detecting a misspecified link function

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

$$\Pr(\mathcal{Y} \leq j) = G(\alpha_j - 8X + X^2), \quad j = 1, 2, 3, 4,$$

where $G(\cdot)$ is the CDF for the Gumbel (max) distribution (see Table 1), $\alpha_1 = -16$, $\alpha_2 = -12$, $\alpha_3 = -8$, $\beta_1 = -8$, $\beta_2 = 1$, and $X \sim \mathcal{U}(1, 7)$. The data are available in the data frame `df3` within the package; see `?df3` for details.

We fit a model with various link functions, where the true link function is log-log. From these output, 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
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 (max) 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

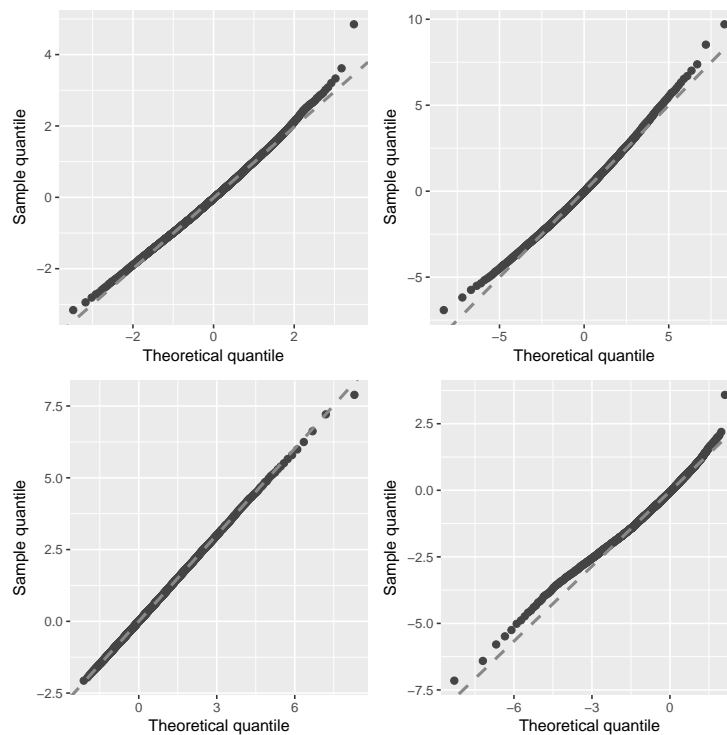


Figure 6: Q-Q plots of the residuals for various cumulative link models fit to simulated data with Gumbel (max) errors. *Top left:* A model with probit link. *Top right:* A model with logit link. *Bottom left:* A model with log-log link (i.e., the correct model). *Bottom right:* A model with complementary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

p -values; hence, the EDF would be relatively straight with a slope of one. The results 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 = "")
```

Checking the proportionality assumption

An important feature of the cumulative link model (1) is the proportional odds assumption, which assumes that the mean structure, $f(X, \beta)$, 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 β , 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

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

where $\alpha_1 = -1.5$, $\alpha_2 = 0$, $\alpha_3 = 1$, $\alpha_4 = 3$, $\beta_1 = 1$, $\beta_2 = 1.5$, and $X \sim \mathcal{U}(-3, 3)$. 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 $S_2 \sim$

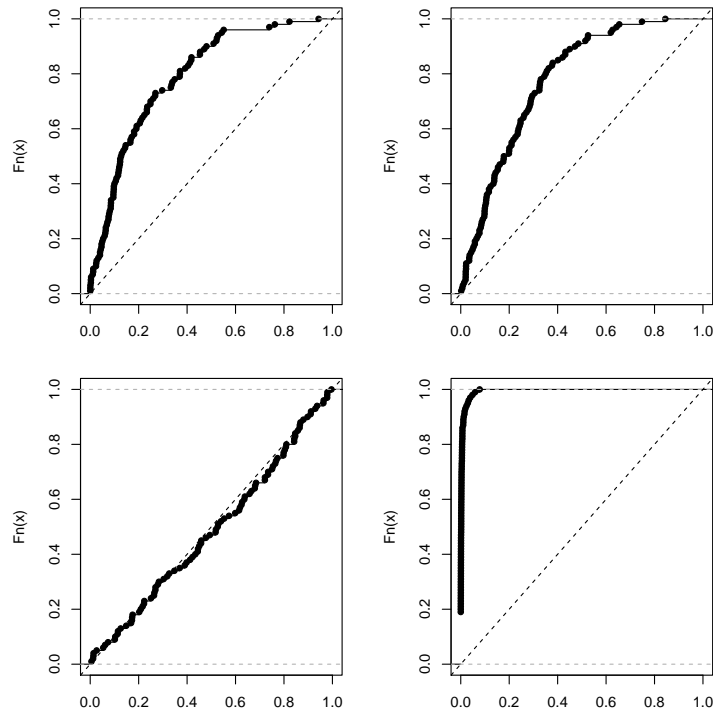


Figure 7: EDFs of the simulated p -values from an Anderson-Darling GOF test for various cumulative link models fit to simulated data with gumbel errors. *Top left:* A model with probit link. *Top right:* A model with logit link. *Bottom left:* A model with log-log link (i.e., the correct model). *Bottom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

$\mathcal{N}(-\beta_2 X, 1)$, 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.

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

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

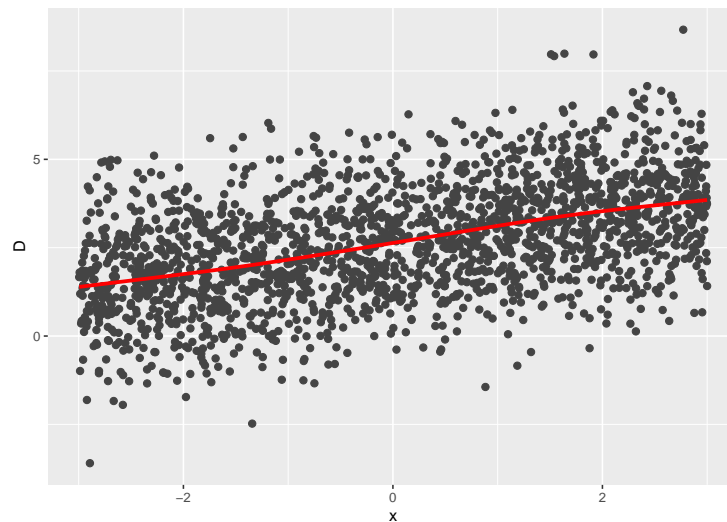


Figure 8: Scatterplot of $D = S_1 - S_2$ vs. x with a nonparametric smooth (red curve).

where $\alpha_1 = -16$, $\alpha_2 = -12$, $\alpha_3 = -8$, $\beta_1 = -5$, $\beta_2 = 3$, $\beta_3 = 10$, $x_1 \sim \mathcal{U}(1, 7)$, 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 `c1m` function from the **ordinal** package. The first model corresponds to the simulated control group, while the second model corresponds to the treatment group.

```
library(ordinal) # for c1m function
fit1 <- c1m(y ~ x1, data = df5[df5$x2 == "Control", ], link = "probit")
fit2 <- c1m(y ~ x1, data = df5[df5$x2 == "Treatment", ], link = "probit")
```

If the true model contains an interaction term x_1x_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 the 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 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).

```
# Figure 9
set.seed(1105) # for reproducibility
df5$s <- c(surrogate(fit1), surrogate(fit2)) # surrogate response values
ggplot(df5, aes(x = x1, y = s)) +
  geom_point(color = "#444444", shape = 19, size = 2, alpha = 0.5) +
  geom_smooth(se = FALSE, size = 1.2, color = "red2") +
  facet_wrap(~ x2) +
  ylab("Surrogate response") +
  xlab(expression(x[1]))
```

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 response is an ordered factor with levels $1 < 2 < 3 < 4 < 5$. The data are available in the **ordinal** package; see `?ordinal::wine` for details.

```
data(wine, package = "ordinal") # load wine data set
wine.c1m <- c1m(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

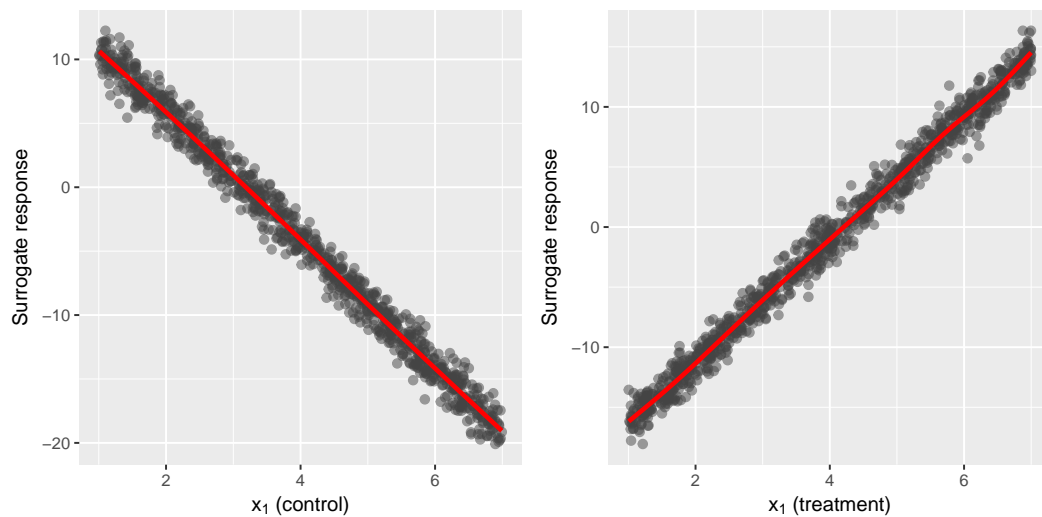


Figure 9: Scatterplot of the surrogate response S versus x_1 with a nonparametric smooth (red line). *Left:* Control group. *Right:* Treatment group.

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).

```
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
)
```

Summary

In this paper, we introduce 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 illustrate the various properties of these residuals and how they can be used to check 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** in the future.

Acknowledgments

The authors would like to thank the anonymous reviewers and the Editor for their helpful comments and suggestions.

Bibliography

A. Agresti. *Analysis of Ordinal Categorical Data*. Wiley Series in Probability and Statistics. John Wiley & Sons, 2010. ISBN 9781118209998. [p381]

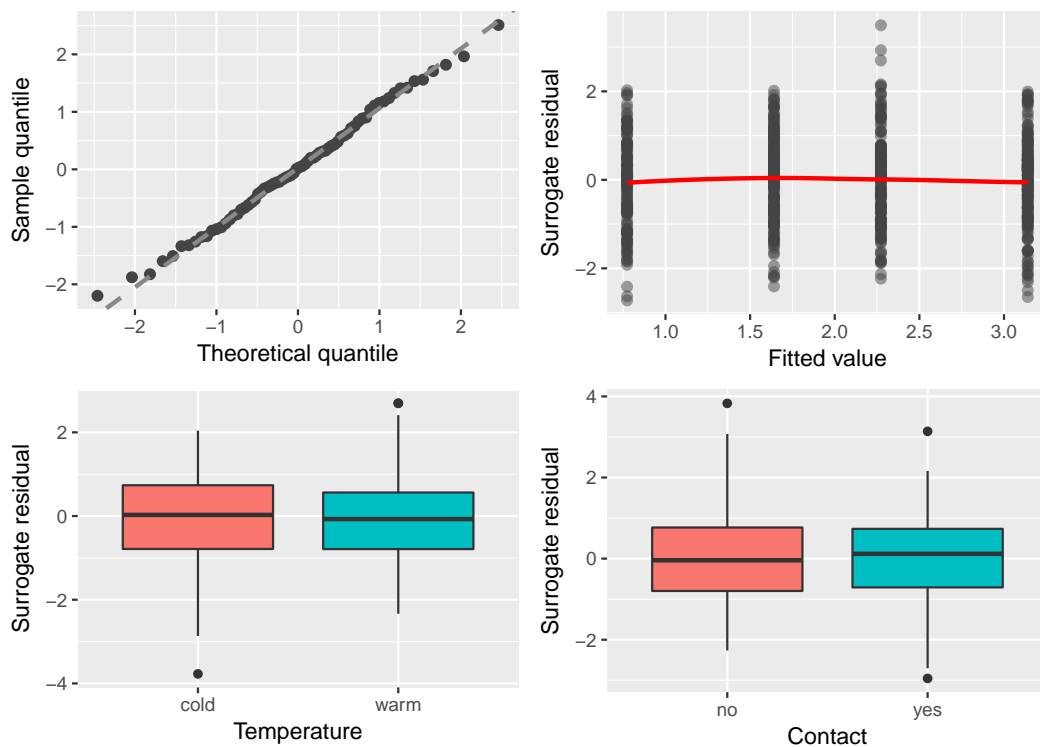


Figure 10: Residual diagnostic plots for the quality of wine example.

- 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. [p382]
- 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. [p382]
- B. Efron. Bootstrap methods: Another look at the jackknife. *Annals of Statistics*, 7(1):1–26, 1979. URL <http://dx.doi.org/10.1214/aos/1176344552>. [p383]
- 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. [p383]
- F. E. Harrell. *Regression Modeling Strategies: With Applications to Linear Models, Logistic Regression, and Survival Analysis*. Graduate Texts in Mathematics. Springer-Verlag, 2001. ISBN 9780387952321. [p389]
- F. E. Harrell. *Rms: Regression Modeling Strategies*, 2017. URL <https://CRAN.R-project.org/package=rms>. R package version 5.1-1. [p382]
- 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>. [p382]
- D. Liu and H. Zhang. Residuals and diagnostics for ordinal regression models: A surrogate approach. *Journal of the American Statistical Association*, 0(ja):0–0, 2017. URL <http://dx.doi.org/10.1080/01621459.2017.1292915>. [p382, 383, 389, 392]
- 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>. [p382]
- 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>. [p391]
- W. N. Venables and B. D. Ripley. *Modern Applied Statistics with S*. Springer-Verlag, New York, 4th edition, 2002. URL <http://www.stats.ox.ac.uk/pub/MASS4>. ISBN 0-387-95457-0. [p382]

H. Wickham. *Ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag, 2009. ISBN 978-0-387-98140-6. URL <http://ggplot2.org>. [p384]

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. [p382]

Brandon M. Greenwell
Department of Mathematics and Statistics
Wright State University
3640 Colonel Glenn Hwy
Dayton, OH 45435
United States of America
ORCID: [0000-0002-8120-0084](https://orcid.org/0000-0002-8120-0084)
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
University of Cincinnati
2925 Campus Green Dr
Cincinnati, OH 45221
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