Package ‘sure’

May 9, 2026

Type Package

Title Surrogate Residuals for Ordinal and General Regression Models

Description An implementation of the surrogate approach to residuals and diagnostics for ordinal and general regression models; for details, see Liu and Zhang (2017) <doi:10.1080/01621459.2017.1292915>. These residuals can be used to construct standard residual plots for model diagnostics (e.g., residual-vs-fitted value plots, residual-vs-covariate plots, Q-Q plots, etc.). The package also provides an 'autoplot' function for producing standard diagnostic plots using 'ggplot2' graphics. The package currently supports cumulative link models from packages 'MASS', 'ordinal', 'rms', and 'VGAM'. Support for binary regression models using the standard 'glm' function is also available.

Version 0.2.0

Depends R (>= 3.1)

Imports ggplot2 (>= 2.2.1), goftest, gridExtra, stats

Suggests MASS, ordinal, rms, testthat, VGAM

License GPL (>= 2)

URL `https://github.com/AFIT-R/sure`

BugReports `https://github.com/AFIT-R/sure/issues`

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

NeedsCompilation no

Author Brandon Greenwell [aut, cre],
Andrew McCarthy [aut],
Brad Boehmke [aut],
Dungang Liu [ctb]

Maintainer Brandon Greenwell <`greenwell.brandon@gmail.com`>

Repository CRAN

Date/Publication 2017-09-19 18:04:46 UTC

Contents

`autoplot.resid` *Residual Plots for Cumulative Link and General Regression Models*

Description

Residual-based diagnostic plots for cumulative link and general regression models using `ggplot2` graphics.

Usage

```
autoplot.resid(object, what = c("qq", "fitted", "covariate"), x = NULL,
  fit = NULL, distribution = qnorm, alpha = 1, xlab = NULL,
  color = "#444444", shape = 19, size = 2, qqpoint.color = "#444444",
  qqpoint.shape = 19, qqpoint.size = 2, qqline.color = "#888888",
  qqline.linetype = "dashed", qqline.size = 1, smooth = TRUE,
  smooth.color = "red", smooth.linetype = 1, smooth.size = 1,
  fill = NULL, ...)

autoplot.clm(object, what = c("qq", "fitted", "covariate"), x = NULL,
  alpha = 1, xlab = NULL, color = "#444444", shape = 19, size = 2,
  qqpoint.color = "#444444", qqpoint.shape = 19, qqpoint.size = 2,
  qqline.color = "#888888", qqline.linetype = "dashed", qqline.size = 1,
  smooth = TRUE, smooth.color = "red", smooth.linetype = 1,
  smooth.size = 1, fill = NULL, ...)

autoplot.glm(object, what = c("qq", "fitted", "covariate"), x = NULL,
  alpha = 1, xlab = NULL, color = "#444444", shape = 19, size = 2,
  qqpoint.color = "#444444", qqpoint.shape = 19, qqpoint.size = 2,
  qqline.color = "#888888", qqline.linetype = "dashed", qqline.size = 1,
  smooth = TRUE, smooth.color = "red", smooth.linetype = 1,
  smooth.size = 1, fill = NULL, ...)

autoplot.lrm(object, what = c("qq", "fitted", "covariate"), x = NULL,
```

```
  alpha = 1, xlab = NULL, color = "#444444", shape = 19, size = 2,
  qqpoint.color = "#444444", qqpoint.shape = 19, qqpoint.size = 2,
  qqline.color = "#888888", qqline.linetype = "dashed", qqline.size = 1,
  smooth = TRUE, smooth.color = "red", smooth.linetype = 1,
  smooth.size = 1, fill = NULL, ...)

autoplot.orm(object, what = c("qq", "fitted", "covariate"), x = NULL,
  alpha = 1, xlab = NULL, color = "#444444", shape = 19, size = 2,
  qqpoint.color = "#444444", qqpoint.shape = 19, qqpoint.size = 2,
  qqline.color = "#888888", qqline.linetype = "dashed", qqline.size = 1,
  smooth = TRUE, smooth.color = "red", smooth.linetype = 1,
  smooth.size = 1, fill = NULL, ...)

autoplot.polr(object, what = c("qq", "fitted", "covariate"), x = NULL,
  alpha = 1, xlab = NULL, color = "#444444", shape = 19, size = 2,
  qqpoint.color = "#444444", qqpoint.shape = 19, qqpoint.size = 2,
  qqline.color = "#888888", qqline.linetype = "dashed", qqline.size = 1,
  smooth = TRUE, smooth.color = "red", smooth.linetype = 1,
  smooth.size = 1, fill = NULL, ...)

autoplot.vgam(object, what = c("qq", "fitted", "covariate"), x = NULL,
  alpha = 1, xlab = NULL, color = "#444444", shape = 19, size = 2,
  qqpoint.color = "#444444", qqpoint.shape = 19, qqpoint.size = 2,
  qqline.color = "#888888", qqline.linetype = "dashed", qqline.size = 1,
  smooth = TRUE, smooth.color = "red", smooth.linetype = 1,
  smooth.size = 1, fill = NULL, ...)

autoplot.vglm(object, what = c("qq", "fitted", "covariate"), x = NULL,
  alpha = 1, xlab = NULL, color = "#444444", shape = 19, size = 2,
  qqpoint.color = "#444444", qqpoint.shape = 19, qqpoint.size = 2,
  qqline.color = "#888888", qqline.linetype = "dashed", qqline.size = 1,
  smooth = TRUE, smooth.color = "red", smooth.linetype = 1,
  smooth.size = 1, fill = NULL, ...)
```

Arguments

`object`	An object of class `clm`, `glm`, `lrm`, `orm`, `polr`, or `vglm`.
`what`	Character string specifying what to plot. Default is `"qq"` which produces a quantile-quantile plots of the residuals.
`x`	A vector giving the covariate values to use for residual-by- covariate plots (i.e., when `what = "covariate"`).
`fit`	The fitted model from which the residuals were extracted. (Only required if `what = "fitted"` and `object` inherits from class `"resid"`.)
`distribution`	Function that computes the quantiles for the reference distribution to use in the quantile-quantile plot. Default is `qnorm` which is only appropriate for models using a probit link function. When `jitter.scale = "probability"`, the reference distribution is always U(-0.5, 0.5). (Only required if `object` inherits from class `"resid"`.)

`alpha`	A single values in the interval [0, 1] controlling the opacity alpha of the plotted points. Only used when `nsim` > 1.
`xlab`	Character string giving the text to use for the x-axis label in residual-by-covariate plots. Default is `NULL`.
`color`	Character string or integer specifying what color to use for the points in the residual vs fitted value/covariate plot. Default is `"black"`.
`shape`	Integer or single character specifying a symbol to be used for plotting the points in the residual vs fitted value/covariate plot.
`size`	Numeric value specifying the size to use for the points in the residual vs fitted value/covariate plot.
`qqpoint.color`	Character string or integer specifying what color to use for the points in the quantile-quantile plot.
`qqpoint.shape`	Integer or single character specifying a symbol to be used for plotting the points in the quantile-quantile plot.
`qqpoint.size`	Numeric value specifying the size to use for the points in the quantile-quantile plot.
`qqline.color`	Character string or integer specifying what color to use for the points in the quantile-quantile plot.
`qqline.linetype`	Integer or character string (e.g., `"dashed"`) specifying the type of line to use in the quantile-quantile plot.
`qqline.size`	Numeric value specifying the thickness of the line in the quantile-quantile plot.
`smooth`	Logical indicating whether or not too add a nonparametric smooth to certain plots. Default is `TRUE`.
`smooth.color`	Character string or integer specifying what color to use for the nonparametric smooth.
`smooth.linetype`	Integer or character string (e.g., `"dashed"`) specifying the type of line to use for the nonparametric smooth.
`smooth.size`	Numeric value specifying the thickness of the line for the nonparametric smooth.
`fill`	Character string or integer specifying the color to use to fill the boxplots for residual-by-covariate plots when `x` is of class `"factor"`. Default is `NULL` which colors the boxplots according to the factor levels.
`...`	Additional optional arguments to be passed onto `resids`.

Value

A `"ggplot"` object.

Examples

```
# See ?resids for an example
?resids
```

`df1` *Simulated Quadratic Data*

Description

Data simulated from a probit model with a quadratic trend. The data are described in Example 2 of Liu and Zhang (2017).

Usage

```
data(df1)
```

Format

A data frame with 2000 rows and 2 variables.

- `y` The response variable; an ordered factor.
- `x` The predictor variable.

References

Liu, Dungang and Zhang, Heping. Residuals and Diagnostics for Ordinal Regression Models: A Surrogate Approach. *Journal of the American Statistical Association* (accepted).

Examples

```
head(df1)
```

`df2` *Simulated Heteroscedastic Data*

Description

Data simulated from a probit model with heteroscedasticity. The data are described in Example 4 of Liu and Zhang (2017).

Usage

```
data(df2)
```

Format

A data frame with 2000 rows and 2 variables.

- `y` The response variable; an ordered factor.
- `x` The predictor variable.

References

Liu, Dungang and Zhang, Heping. Residuals and Diagnostics for Ordinal Regression Models: A Surrogate Approach. *Journal of the American Statistical Association* (accepted).

Examples

```
head(df2)
```

`df3` *Simulated Gumbel Data*

Description

Data simulated from a log-log model with a quadratic trend. The data are described in Example 3 of Liu and Zhang (2017).

Usage

```
data(df3)
```

Format

A data frame with 2000 rows and 2 variables.

- `y` The response variable; an ordered factor.
- `x` The predictor variable.

References

Liu, Dungang and Zhang, Heping. Residuals and Diagnostics for Ordinal Regression Models: A Surrogate Approach. *Journal of the American Statistical Association* (accepted).

Examples

```
head(df3)
```

`df4` *Simulated Proportionality Data*

Description

Data simulated from two separate ordered probit models with different coefficients. The data are described in Example 5 of Liu and Zhang (2017).

Usage

```
data(df4)
```

Format

A data frame with 2000 rows and 2 variables.

- `y` The response variable; an ordered factor.
- `x` The predictor variable.

References

Liu, Dungang and Zhang, Heping. Residuals and Diagnostics for Ordinal Regression Models: A Surrogate Approach. *Journal of the American Statistical Association* (accepted).

Examples

```
head(df4)
```

`df5` *Simulated Interaction Data*

Description

Data simulated from an ordered probit model with an interaction term.

Usage

```
data(df5)
```

Format

A data frame with 2000 rows and 3 variables.

- `y` The response variable; an ordered factor.
- `x1` A continuous predictor.
- `x2` A factor with two levels: `Control` and `Treatment`.

Examples

```
head(df5)
```

---

gof *Goodness-of-Fit Simulation*

Description

Simulate p-values from a goodness-of-fit test.

Usage

```
gof(object, nsim = 10, test = c("ks", "ad", "cvm"), ...)

## Default S3 method:
gof(object, nsim = 10, test = c("ks", "ad", "cvm"), ...)

## S3 method for class 'gof'
plot(x, ...)
```

Arguments

`object`	An object of class `clm`, `glm`, `lrm`, `orm`, `polr`, or `vglm`.
`nsim`	Integer specifying the number of bootstrap replicates to use.
`test`	Character string specifying which goodness-of-fit test to use. Current options include: `"ks"` for the Kolmogorov-Smirnov test, `"ad"` for the Anderson-Darling test, and `"cvm"` for the Cramer-Von Mises test. Default is `"ks"`.
`...`	Additional optional arguments. (Currently ignored.)
`x`	An object of class `"gof"`.

Details

Under the null hypothesis, the distribution of the p-values should appear uniformly distributed on the interval [0, 1]. This can be visually investigated using the `plot` method. A 45 degree line is indicative of a "good" fit.

Value

A numeric vector of class `"gof"`, `"numeric"` containing the simulated p-values.

Examples

```
# See ?resids for an example
?resids
```

`resids` *Surrogate Residuals*

Description

Surrogate-based residuals for cumulative link and general regression models.

Usage

```
resids(object, ...)

## Default S3 method:
resids(object, method = c("latent", "jitter"),
  jitter.scale = c("probability", "response"), nsim = 1L, ...)
```

Arguments

`object`	An object of class `clm`, `glm`, `lrm`, `orm`, `polr`, `vgam` (jittering only), or `vglm`.
`...`	Additional optional arguments. (Currently ignored.)
`method`	Character string specifying the type of surrogate to use; for details, see Liu and Zhang (2017). Can be one of `"latent"` or `"jitter"`.
`jitter.scale`	Character string specifying the scale on which to perform the jittering. Should be one of `"probability"` or `"response"`. (Currently ignored for cumulative link models.)
`nsim`	Integer specifying the number of bootstrap replicates to use. Default is `1L` meaning no bootstrap samples.

Value

A numeric vector of class `c("numeric", "resid")` containing the residuals. Additionally, if `nsim` > 1, then the result will contain the attributes:

`boot.reps` A matrix with `nsim` columns, one for each bootstrap replicate of the residuals. Note, these are random and do not correspond to the original ordering of the data;

`boot.id` A matrix with `nsim` columns. Each column contains the observation number each residual corresponds to in `boot.reps`. (This is used for plotting purposes.)

Note

Surrogate residuals require sampling from a continuous distribution; consequently, the result will be different with every call to `resids`. The internal functions used for sampling from truncated distributions when `method = "latent"` are based on modified versions of `rtrunc` and `qtrunc`.

References

Liu, Dungang and Zhang, Heping. Residuals and Diagnostics for Ordinal Regression Models: A Surrogate Approach. *Journal of the American Statistical Association* (accepted). URL http://www.tandfonline.com/doi/abs/1

Nadarajah, Saralees and Kotz, Samuel. R Programs for Truncated Distributions. *Journal of Statistical Software, Code Snippet*, 16(2), 1-8, 2006. URL https://www.jstatsoft.org/v016/c02.

Examples

```
#
# Residuals for binary GLMs using the jittering method
#

# Load the MASS package (for the polr function)
library(MASS)

# Simulated probit data with quadratic trend
data(df1)

# Fit logistic regression models (with and without quadratic trend)
fit1 <- polr(y ~ x + I(x ^ 2), data = df1, method = "probit")
fit2 <- polr(y ~ x, data = df1, method = "probit")

# Construct residuals
set.seed(102)  # for reproducibility
res1 <- resids(fit1)
res2 <- resids(fit2)

# Residual-vs-covariate plots
par(mfrow = c(1, 2))
scatter.smooth(df1$x, res1, lpars = list(lwd = 2, col = "red"),
               xlab = expression(x), ylab = "Surrogate residual",
               main = "Correct model")
scatter.smooth(df1$x, res2, lpars = list(lwd = 2, col = "red"),
               xlab = expression(x), ylab = "Surrogate residual",
               main = "Incorrect model")

## Not run:
#
# Residuals for cumulative link models using the latent method
#

# Load required packages
library(ggplot2)  # for autoplot function
library(MASS)     # for polr function
library(ordinal)  # for clm function

#
# Detecting a misspecified mean structure
#

# Data simulated from a probit model with a quadratic trend
```

```
data(df1)
?df1

# Fit a probit model with/without a quadratic trend
fit.bad <- polr(y ~ x, data = df1, method = "probit")
fit.good <- polr(y ~ x + I(x ^ 2), data = df1, method = "probit")

# Some residual plots
p1 <- autoplot(fit.bad, what = "covariate", x = df1$x)
p2 <- autoplot(fit.bad, what = "qq")
p3 <- autoplot(fit.good, what = "covariate", x = df1$x)
p4 <- autoplot(fit.good, what = "qq")

# Display all four plots together (top row corresponds to bad model)
grid.arrange(p1, p2, p3, p4, ncol = 2)

#
# Detecting heteroscedasticity
#

# Data simulated from a probit model with heteroscedasticity.
data(df2)
?df2

# Fit a probit model with/without a quadratic trend
fit <- polr(y ~ x, data = df2, method = "probit")

# Some residual plots
p1 <- autoplot(fit, what = "covariate", x = df1$x)
p2 <- autoplot(fit, what = "qq")
p3 <- autoplot(fit, what = "fitted")

# Display all three plots together
grid.arrange(p1, p2, p3, ncol = 3)

#
# Detecting a misspecified link function
#

# Data simulated from a log-log model with a quadratic trend.
data(df3)
?df3

# Fit models with correctly specified link function
clm.loglog <- clm(y ~ x + I(x ^ 2), data = df3, link = "loglog")
polr.loglog <- polr(y ~ x + I(x ^ 2), data = df3, method = "loglog")

# Fit models with misspecified link function
clm.probit <- clm(y ~ x + I(x ^ 2), data = df3, link = "probit")
polr.probit <- polr(y ~ x + I(x ^ 2), data = df3, method = "probit")

# Q-Q plots of the residuals (with bootstrapping)
p1 <- autoplot(clm.probit, nsim = 50, what = "qq") +
```

```
  ggtitle("clm: probit link")
p2 <- autoplot(clm.loglog, nsim = 50, what = "qq") +
  ggtitle("clm: log-log link (correct link function)")
p3 <- autoplot(polr.probit, nsim = 50, what = "qq") +
  ggtitle("polr: probit link")
p4 <- autoplot(polr.loglog, nsim = 50, what = "qq") +
  ggtitle("polr: log-log link (correct link function)")
grid.arrange(p1, p2, p3, p4, ncol = 2)

# We can also try various goodness-of-fit tests
par(mfrow = c(1, 2))
plot(gof(clm.probit, nsim = 50))
plot(gof(clm.loglog, nsim = 50))

## End(Not run)
```

`sure` *sure: An R package for constructing surrogate-based residuals and diagnostics for ordinal and general regression models.*

Description

The `sure` package provides surrogate-based residuals for fitted ordinal and general (e.g., binary) regression models of class `clm`, `glm`, `lrm`, `orm`, `polr`, or `vglm`.

Details

The development version can be found on GitHub: `https://github.com/AFIT-R/sure`. As of right now, `sure` exports the following functions:

- `resids` - construct (surrogate-based) residuals;
- `autoplot` - plot diagnostics using `ggplot2`-based graphics;
- `gof` - simulate p-values from a goodness-of-fit test.

References

Liu, Dungang and Zhang, Heping. Residuals and Diagnostics for Ordinal Regression Models: A Surrogate Approach. *Journal of the American Statistical Association* (accepted).

`surrogate` *Surrogate Response*

Description

Simulate surrogate response values for cumulative link regression models using the latent method described in Liu and Zhang (2017).

Usage

```
surrogate(object, method = c("latent", "jitter"),
  jitter.scale = c("probability", "response"), nsim = 1L, ...)
```

Arguments

`object`	An object of class `clm`, `lrm`, `orm`, `polr`, or `vglm`.
`method`	Character string specifying the type of surrogate to use; for details, see Liu and Zhang (2017). For cumulative link models, the latent variable method is used. For binary GLMs, the jittering approach is employed. (Currently ignored.)
`jitter.scale`	Character string specifying the scale on which to perform the jittering. Should be one of `"probability"` or `"response"`. (Currently ignored for cumulative link models.)
`nsim`	Integer specifying the number of bootstrap replicates to use. Default is `1L` meaning no bootstrap samples.
`...`	Additional optional arguments. (Currently ignored.)

Value

A numeric vector of class `c("numeric", "surrogate")` containing the simulated surrogate response values. Additionally, if `nsim` > 1, then the result will contain the attributes:

`boot.reps` A matrix with `nsim` columns, one for each bootstrap replicate of the surrogate values. Note, these are random and do not correspond to the original ordering of the data;

`boot.id` A matrix with `nsim` columns. Each column contains the observation number each surrogate value corresponds to in `boot.reps`. (This is used for plotting purposes.)

Note

Surrogate response values require sampling from a continuous distribution; consequently, the result will be different with every call to `surrogate`. The internal functions used for sampling from truncated distributions are based on modified versions of `rtrunc` and `qtrunc`.

References

Liu, Dungang and Zhang, Heping. Residuals and Diagnostics for Ordinal Regression Models: A Surrogate Approach. *Journal of the American Statistical Association* (accepted). URL http://www.tandfonline.com/doi/abs/1

Nadarajah, Saralees and Kotz, Samuel. R Programs for Truncated Distributions. *Journal of Statistical Software, Code Snippet*, 16(2), 1-8, 2006. URL https://www.jstatsoft.org/v016/c02.

Index