# Package 'PResiduals'

October 12, 2022

**Type** Package

**Title** Probability-Scale Residuals and Residual Correlations

**Version** 1.0-1

**Author** Charles Dupont, Jeffrey Horner, Chun Li, Qi Liu, Bryan Shepherd

**Maintainer** Chun Li <cli77199@usc.edu>

**Description** Computes probability-scale residuals and residual correlations
for continuous, ordinal, binary, count, and time-to-event data <doi:10.18637/jss.v094.i12>.

**Imports** MASS, Formula, rms, SparseM,

**Suggests** survival, testthat

**License** GPL (>= 2)

**Encoding** UTF-8

**LazyData** true

**Collate** 'GKGamma.R' 'PResidData.R' 'PResiduals-package.R' 'pgumbel.R'
'diagn.R' 'newPolr.R' 'cobot.R' 'cocobot.R' 'condis.R'
'conditional_Spearman.R' 'corTS.R' 'corr.R' 'countbot.R'
'getCI.R' 'kernel.function.R' 'lm.scores.R' 'megabot.R'
'nb.scores.R' 'orm.scores.R' 'partial_Spearman.R'
'plot.conditional_Spearman.R' 'poisson.scores.R' 'presid.R'
'print.cobot.R' 'print.cocobot.R'
'print.conditional_Spearman.R' 'print.partial_Spearman.R'

**NeedsCompilation** no

**RoxygenNote** 7.1.1

**Repository** CRAN

**Date/Publication** 2021-06-24 06:40:05 UTC

## R topics documented:

---

`PResiduals-package` *Computes probability-scale residuals and residual correlations.*

---

## Description

This package outputs probability-scale residuals from multiple models and computes residual correlation. Probability-scale residual can be computed for continuous, ordinal, binary, count, and time-to-event data (although the current implementation is only for ordinal variables). Plots of probability-scale residuals can be useful for model diagnostics. Residual correlation can be used to test for conditional independence between multiple types of variables.

## Author(s)

Bryan Shepherd <bryan.shepherd@vanderbilt.edu>

Chun Li <cxl791@case.edu>

Qi Liu <qi.liu4@merck.com>

Charles Dupont <charles.dupont@vanderbilt.edu>

Jeffrey Horner <jeffrey.horner@vanderbilt.edu>

cobot *Conditional ordinal by ordinal tests for association.*

## Description

cobot tests for independence between two ordered categorical variables, $X$ and $Y$ conditional on other variables, $Z$. The basic approach involves fitting models of $X$ on $Z$ and $Y$ on $Z$ and determining whether there is any remaining information between $X$ and $Y$. This is done by computing one of 3 test statistics. T1 compares empirical distribution of $X$ and $Y$ with the joint fitted distribution of $X$ and $Y$ under independence conditional on $Z$. T2 computes the correlation between ordinal (probability-scale) residuals from both models and tests the null of no residual correlation. T3 evaluates the concordance–disconcordance of data drawn from the joint fitted distribution of $X$ and $Y$ under conditional independence with the empirical distribution. Details are given in *Li C and Shepherd BE, Test of association between two ordinal variables while adjusting for covariates. Journal of the American Statistical Association 2010, 105:612-620.*

## Usage

```
cobot(
  formula,
  link = c("logit", "probit", "cloglog", "loglog", "cauchit"),
  link.x = link,
  link.y = link,
  data,
  subset,
  na.action = na.fail,
  fisher = TRUE,
  conf.int = 0.95
)
```

## Arguments

| | |
|---|---|
| formula | an object of class Formula (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'. |
| link | The link family to be used for ordinal models of both $X$ and $Y$. Defaults to 'logit'. Other options are 'probit', 'cloglog','loglog', and 'cauchit'. |
| link.x | The link function to be used for a model of the first ordered variable. Defaults to value of link. |
| link.y | The link function to be used for a model of the second variable. Defaults to value of link. |
| data | an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which cobot is called. |

| | |
|---|---|
| subset | an optional vector specifying a subset of observations to be used in the fitting process. |
| na.action | how NAs are treated. |
| fisher | logical; if TRUE, Fisher transformation and delta method a used to compute p value for the test statistic based on correlation of residuals. |
| conf.int | numeric specifying confidence interval coverage. |

## Details

formula is specified as $X$ | $Y$ ~ $Z$. This indicates that models of $X$ ~ $Z$ and $Y$ ~ $Z$ will be fit. The null hypothsis to be tested is $H_0 : X$ independant of $Y$ conditional on $Z$.

Note that T2 can be thought of as an adjusted rank correlation.(*Li C and Shepherd BE, A new residual for ordinal outcomes. Biometrika 2012; 99:473-480*)

## Value

object of 'cobot' class.

## References

Li C and Shepherd BE, Test of association between two ordinal variables while adjusting for covariates. Journal of the American Statistical Association 2010, 105:612-620.

Li C and Shepherd BE, A new residual for ordinal outcomes. Biometrika 2012; 99:473-480

## See Also

Formula, as.data.frame

## Examples

```
data(PResidData)
cobot(x|y~z, data=PResidData)
```

---

cocobot *Conditional continuous by ordinal tests for association.*

---

## Description

cocobot tests for independence between an ordered categorical variable, $X$, and a continuous variable, $Y$, conditional on other variables, $Z$. The basic approach involves fitting an ordinal model of $X$ on $Z$, a linear model of $Y$ on $Z$, and then determining whether there is any residual information between $X$ and $Y$. This is done by computing residuals for both models, calculating their correlation, and testing the null of no residual correlation. This procedure is analogous to test statistic T2 in cobot. Two test statistics (correlations) are currently output. The first is the correlation between probability-scale residuals. The second is the correlation between the observed-minus-expected residual for the continuous outcome model and a latent variable residual for the ordinal model (Li C and Shepherd BE, 2012).

## Usage

```
cocobot(
  formula,
  data,
  link = c("logit", "probit", "cloglog", "loglog", "cauchit"),
  subset,
  na.action = getOption("na.action"),
  emp = TRUE,
  fisher = TRUE,
  conf.int = 0.95
)
```

## Arguments

| | |
|---|---|
| formula | an object of class Formula (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'. |
| data | an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which cocobot is called. |
| link | The link family to be used for the ordinal model of $X$ on $Z$. Defaults to 'logit'. Other options are 'probit', 'cloglog', 'loglog', and 'cauchit'. |
| subset | an optional vector specifying a subset of observations to be used in the fitting process. |
| na.action | action to take when NA present in data. |
| emp | logical indicating whether the residuals from the model of $Y$ on $Z$ are computed based on the assumption of normality (FALSE) or empirically (TRUE). |
| fisher | logical indicating whether to apply fisher transformation to compute confidence intervals and p-values for the correlation. |
| conf.int | numeric specifying confidence interval coverage. |

## Details

Formula is specified as $X$ | $Y$ ~ $Z$. This indicates that models of $X$ ~ $Z$ and $Y$ ~ $Z$ will be fit. The null hypothsis to be tested is $H_0$ : $X$ independant of $Y$ conditional on $Z$. The ordinal variable, $X$, must precede the | and be a factor variable, and $Y$ must be continuous.

## Value

object of 'cocobot' class.

## References

Li C and Shepherd BE (2012) A new residual for ordinal outcomes. *Biometrika*. **99**: 473–480.

Shepherd BE, Li C, Liu Q (2016) Probability-scale residuals for continuous, discrete, and censored data. *The Canadian Journal of Statistics*. **44**: 463–479.

## Examples

```
data(PResidData)
cocobot(y|w ~ z, data=PResidData)
```

---

`conditional_Spearman` *Conditional Partial Spearman's Rank Correlation*

---

## Description

`conditional_Spearman` computes the partial Spearman's rank correlation between variable $X$ and variable $Y$ adjusting for variable $Z$ conditional on $Zc$. $X$ and $Y$ can be any orderable variables, including continuous and discrete variables. Covariate $Z$ can be multidimensional. $X$, $Y$, and $Z$ are specified by the argument '`formula`'. $Zc$ is a one-dimensional covariate, specified by the argument '`conditional.by`'. The basic approach involves fitting a specified model of $X$ on $Z$, a specified model of $Y$ on $Z$, obtaining the probability-scale residuals, *Xres* and *Yres*, from both models, and then modeling their Pearson's correlation conditional on $Zc$. Different methods are provided to model the Pearson's correlation between the two sets of probability-scale residuals. See details in '`conditional.method`'. As in '`partial.Spearman`', by default `conditional_Spearman` uses cumulative link models for both continous and discrete ordinal variables $X$ and $Y$ to preserve the rank-based nature of Spearman's correlation. For some specific types of variables, options of fitting parametric models are also available. See details in '`fit.x`' and '`fit.y`'.

## Usage

```
conditional_Spearman(
  formula,
  conditional.by,
  data,
  conditional.method = c("lm", "kernel", "stratification"),
  conditional.formula = paste("~", conditional.by, sep = ""),
  kernel.function = c("normal", "gaussian", "triweight", "quartic", "biweight",
    "epanechnikov", "uniform", "triangle"),
  kernel.bandwidth = "silverman",
  fit.x = "orm",
  fit.y = "orm",
  link.x = c("logit", "probit", "cloglog", "loglog", "cauchit", "logistic"),
  link.y = c("logit", "probit", "cloglog", "loglog", "cauchit", "logistic"),
  subset,
  na.action = getOption("na.action"),
  fisher = TRUE,
  conf.int = 0.95
)
```

## Arguments

| | |
|---|---|
| `formula` | an object of class `Formula` (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'. |

`conditional.by` the name of the variable on which the partial Spearman's correlation is conditional. See 'Details'.

`data` an optional data frame, list or environment (or object coercible by `as.data.frame` to a data frame) containing the variables in the model. If not found in `data`, the variables are taken from `environment(formula)`, typically the environment from which `conditional_Spearman` is called.

`conditional.method`
the method to be used for modeling conditional correlation between probability-scale residuals. The default option is '`lm`', which fits linear regression models for *Xres*~*Yres* on *Zc*, *Xres^2* on *Zc*, and *Yres^2* on *Zc*, and then uses the fitted values to compute the Pearson's correlation between *Xres* and *Yres* conditional on *Zc*. Other options include '`kernel`', which computes correlation between *Xres* and *Yres* conditional on *Zc* using a kernel weighted method, and '`stratification`', which computes the correlation between *Xres* and *Yres* seperately for each value of *Zc*.

`conditional.formula`
the formula to be used when '`conditional.method`' is specified as '`lm`'.

`kernel.function`
the kernel function to be used when '`conditional.method`' is specified as '`kernel`'. Defaults to '`normal`'. Other options are '`triweight`', '`quartic`', '`biweight`', '`epanechnikov`', '`uniform`', and '`triangle`'.

`kernel.bandwidth`
the kernel bandwidth to be used when '`conditional.method`' is specified as '`kernel`'. The default value is calculated using Silverman' rule. Users can also specify a positive numeric value.

`fit.x, fit.y` the fitting functions used for the model of $X$ or $Y$ on $Z$. The default function is '`orm`', which fits cumulative link models for continuous or discrete ordinal variables. Other options include '`lm`' (fit linear regression models and obtain the probability-scale residuals by assuming normality), '`lm.emp`' (fit linear regression and obtain the probability-scale residuals by empirical ranking), '`poisson`' (fit Poisson models for count variables), '`nb`' (fit negative binomial models for count variables), and '`logistic`' (fit logistic regression models for binary variables).

`link.x, link.y` the link family to be used for the ordinal model of $X$ on $Z$. Defaults to '`logit`'. Other options are '`probit`', '`cloglog`', '`cauchit`', and '`logistic`' (equivalent with '`logit`'). Used only when '`fit.x`' is '`orm`'.

`subset` an optional vector specifying a subset of observations to be used in the fitting process.

`na.action` action to take when `NA` present in data.

`fisher` logical indicating whether to apply fisher transformation to compute confidence intervals and p-values for the correlation.

`conf.int` numeric specifying confidence interval coverage.

## Details

To compute the partial Spearman's rank correlation between $X$ and $Y$ adjusting for $Z$ conditional on *Zc*, '`formula`' is specified as $X$ `|` $Y$ `~` $Z$ and '`conditional.by`' is specified as *Zc*. This indi-

cates that models of $X$ ~ $Z$ and $Y$ ~ $Z$ will be fit, and the correlation between the probability-scale residuals from these two models will be modeled conditional on $Zc$.

## Value

object of 'conditional_Spearman' class.

## References

Li C and Shepherd BE (2012) A new residual for ordinal outcomes. *Biometrika*. **99**: 473–480.

Shepherd BE, Li C, Liu Q (2016) Probability-scale residuals for continuous, discrete, and censored data. *The Canadian Jouranl of Statistics*. **44**:463–476.

Liu Q, Shepherd BE, Wanga V, Li C (2018) Covariate-Adjusted Spearman's Rank Correlation with Probability-Scale Residuals. *Biometrics*. **74**:595–605.

## See Also

print.conditional_Spearman,print.conditional_Spearman

## Examples

```
data(PResidData)
library(rms)
#### fitting cumulative link models for both Y and W
result <- conditional_Spearman(c|y~ x + w, conditional.by="w",
                                  conditional.method="lm", conditional.formula="~rcs(w)",
                                        fit.x="poisson",fit.y="orm",
                                        data=PResidData, fisher=TRUE)
plot(result)
```

---

# corr *Calculates the weighted correlation given a data set and a set of weights.*

---

## Description

This is a copy of corr function from the boot package. It calculates the correlation coefficient in weighted form.

## Usage

```
corr(d, w = rep(1, nrow(d))/nrow(d))
```

## Arguments

| | |
|---|---|
| d | a matrix with two columns corresponding to the two variables whose correlation we wish to calculate. |
| w | a vector of weights to be applied to each pair of observations. The default is equal weights for each pair. Normalization takes place within the function so sum(w) need not equal 1. |

## Value

the correlation coefficient between d[,1] and d[,2].

---

`countbot` *Conditional count by ordinal tests for association.*

---

## Description

`countbot` tests for independence between an ordered categorical variable, $X$, and a count variable, $Y$, conditional on other variables, $Z$. The basic approach involves fitting an ordinal model of $X$ on $Z$, a Poisson or Negative Binomial model of $Y$ on $Z$, and then determining whether there is any residual information between $X$ and $Y$. This is done by computing residuals for both models, calculating their correlation, and testing the null of no residual correlation. This procedure is analogous to test statistic `T2` in `cobot`. Two test statistics (correlations) are currently output. The first is the correlation between probability-scale residuals. The second is the correlation between the Pearson residual for the count outcome model and a latent variable residual for the ordinal model (Li C and Shepherd BE, 2012).

## Usage

```
countbot(
  formula,
  data,
  link.x = c("logit", "probit", "loglog", "cloglog", "cauchit"),
  fit.y = c("poisson", "negative binomial"),
  subset,
  na.action = getOption("na.action"),
  fisher = TRUE,
  conf.int = 0.95
)
```

## Arguments

| | |
|---|---|
| `formula` | an object of class `Formula` (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'. |
| `data` | an optional data frame, list or environment (or object coercible by `as.data.frame` to a data frame) containing the variables in the model. If not found in `data`, the variables are taken from `environment(formula)`, typically the environment from which `countbot` is called. |
| `link.x` | The link family to be used for the ordinal model of $X$ on $Z$. Defaults to '`logit`'. Other options are '`probit`', '`cloglog`','`loglog`', and '`cauchit`'. |
| `fit.y` | The error distribution for the count model of $Y$ on $Z$. Defaults to '`poisson`'. The other option is '`negative binomial`'. If '`negative binomial`' is specified, `glm.nb` is called to fit the count model. |

| | |
|---|---|
| subset | an optional vector specifying a subset of observations to be used in the fitting process. |
| na.action | action to take when NA present in data. |
| fisher | logical indicating whether to apply fisher transformation to compute confidence intervals and p-values for the correlation. |
| conf.int | numeric specifying confidence interval coverage. |

## Details

Formula is specified as $X$ | $Y$ ~ $Z$. This indicates that models of $X$ ~ $Z$ and $Y$ ~ $Z$ will be fit. The null hypothesis to be tested is $H_0 : X$ independent of $Y$ conditional on $Z$. The ordinal variable, $X$, must precede the | and be a factor variable, and $Y$ must be an integer.

## Value

object of 'cocobot' class.

## References

Li C and Shepherd BE (2012) A new residual for ordinal outcomes. *Biometrika*. **99**: 473–480.

Shepherd BE, Li C, Liu Q (2016) Probability-scale residuals for continuous, discrete, and censored data. *The Canadian Journal of Statistics*. **44**: 463–479.

## Examples

```
data(PResidData)
countbot(x|c ~z, fit.y="poisson",data=PResidData)
countbot(x|c ~z, fit.y="negative binomial",data=PResidData)
```

---

# diagn *Extract or construct a diagonal matrix.*

---

## Description

This works like diag except when x is a single integer value. If x is a single integer value then it assumes that you want a 1 by 1 matrix with the value set to x

## Usage

```
diagn(x = 1, nrow = length(x), ncol = nrow)
```

## Arguments

| | |
|---|---|
| x | a matrix, vector or 1D array, or missing. |
| nrow, ncol | optional dimensions for the result when x is not a matrix. |

## Value

matrix with diagonal elements set to x

## See Also

diag

## Examples

```
diag(5)
diagn(5)
```

---

GKGamma *Goodman-Kruskal's $\gamma$*

---

## Description

Computes Goodman-Kruskal's $\gamma$

## Usage

```
GKGamma(M)
```

## Arguments

| | |
|---|---|
| M | a matrix |

## Value

| | |
|---|---|
| scon | concordance |
| sdis | disconcordance |
| gamma | a real number between -1 and 1. calculated as gamma $= \frac{\texttt{scon}-\texttt{sdis}}{\texttt{scon}+\texttt{sdis}}$ |

## References

Goodman LA, Kruskal WH (1954) Measures of association for cross classifications, Journal of the American Statistical Association, 49, 732-764.

## kernel.function *kernel.function*

### Description

kernel.function calculates several kernel functions (uniform, triangle, epanechnikov, biweight, triweight, gaussian).

### Usage

```
kernel.function(u, kernel = "normal", product = TRUE)
```

### Arguments

| | |
|---|---|
| u | n x d matrix |
| kernel | text string |
| product | or spherical kernel if d>1 |

### Details

slightly modified version of the kernel.function from the gplm package. The kernel parameter is a text string specifying the univariate kernel function which is either the gaussian pdf or proportional to (1-|u|^p)^q. Possible text strings are "triangle" (p=q=1), "uniform" (p=1, q=0), "epanechnikov" (p=2, q=1), "biweight" or "quartic" (p=q=2), "triweight" (p=2, q=3), "gaussian" or "normal" (gaussian pdf). The multivariate kernels are obtained by a product of unvariate kernels K(u_1)...K(u_d) or by a spherical (radially symmetric) kernel proportional to K(||u||). (The resulting kernel is a density, i.e. integrates to 1.)

### Value

matrix with diagonal elements set to x

## megabot *Conditional tests for association.*

### Description

megabot tests for correlation between a variable, $X$, and another variable, $Y$, conditional on other variables, $Z$. The basic approach involves fitting an specified model of $X$ on $Z$, a specified model of $Y$ on $Z$, and then determining whether there is any remaining information between $X$ and $Y$. This is done by computing residuals for both models, calculating their correlation, and testing the null of no residual correlation. The test statistic output is the correlation between probability-scale residuals. $X$ and $Y$ can be continous or ordered discrete variables. megabot replicates the functionality of cobot, cocobot, and countbot

## Usage

```
megabot(
  formula,
  data,
  fit.x,
  fit.y,
  link.x = c("logit", "probit", "cloglog", "loglog", "cauchit", "logistic"),
  link.y = c("logit", "probit", "cloglog", "loglog", "cauchit", "logistic"),
  subset,
  na.action = getOption("na.action"),
  fisher = TRUE,
  conf.int = 0.95
)
```

## Arguments

| | |
|---|---|
| formula | an object of class Formula (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'. |
| data | an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which megabot is called. |
| fit.x, fit.y | The fitting function used for the model of $X$ or $Y$ on $Z$. Options are 'ordinal', 'lm', 'lm.emp', 'poisson', 'nb', and 'orm'. |
| link.x, link.y | The link family to be used for the ordinal model of $X$ on $Z$. Defaults to 'logit'. Other options are 'probit', 'cloglog','loglog', 'cauchit', and 'logistic'(equivalent with 'logit'). Used only when 'fit.x' is either 'ordinal' or 'orm'. |
| subset | an optional vector specifying a subset of observations to be used in the fitting process. |
| na.action | action to take when NA present in data. |
| fisher | logical indicating whether to apply fisher transformation to compute confidence intervals and p-values for the correlation. |
| conf.int | numeric specifying confidence interval coverage. |

## Details

Formula is specified as $X$ | $Y$ ~ $Z$. This indicates that models of $X$ ~ $Z$ and $Y$ ~ $Z$ will be fit. The null hypothesis to be tested is $H_0 : X$ independent of $Y$ conditional on $Z$.

## Value

object of 'cocobot' class.

## References

Li C and Shepherd BE (2012) A new residual for ordinal outcomes. *Biometrika*. **99**: 473–480.

Shepherd BE, Li C, Liu Q (2016) Probability-scale residuals for continuous, discrete, and censored data. *The Canadian Journal of Statistics*. **44**: 463–479.

## Examples

```
data(PResidData)
megabot(y|w ~ z, fit.x="ordinal", fit.y="lm.emp", data=PResidData)
```

---

`newpolr` *slightly modified version of polr from MASS*

---

## Description

slightly modified version of polr from MASS

## Usage

```
newpolr(
  formula,
  data,
  weights,
  start,
  ...,
  subset,
  na.action,
  contrasts = NULL,
  Hess = FALSE,
  model = TRUE,
  method = c("logit", "probit", "cloglog", "loglog", "cauchit")
)
```

## Arguments

| | |
|---|---|
| `formula` | a formula |
| `data` | an optional data frame, list or environment (or object coercible by `as.data.frame` to a data frame) containing the variables in the model. If not found in `data`, the variables are taken from `environment(formula)`, typically the environment from which `cobot` is called. |
| `weights` | optional case weights in fitting. Default to 1. |
| `start` | initial values for the parameters. |
| `...` | additional arguments to be passed to `optim`, most often a `control` argument. |
| `subset` | an optional vector specifying a subset of observations to be used in the fitting process. |

| | |
|---|---|
| na.action | a function which indicates what should happen when the data contain NAs. The default is is na.fail. Another possible value is NULL, no action. Value na.exclude can be useful. |
| contrasts | a list of contrasts to be used for some or all of the factors appearing as variables in the model formula. |
| Hess | logical for whether the Hessian (the observed information matrix) should be returned. Use this if you intend to call summary or vcov on the fit. |
| model | logical for whether the model matrix should be returned. |
| method | logistic or probit or complementary log-log, loglog, or cauchit (corresponding to a Cauchy latent variable). |

## Value

A object of class "polr". This has components

| | |
|---|---|
| coefficients | the coefficients of the linear predictor, which has no intercept. |
| zeta | the intercepts for the class boundaries. |
| deviance | the residual deviance. |
| fitted.values | a matrix, with a column for each level of the response. |
| lev | the names of the response levels. |
| terms | the terms structure describing the model. |
| df.residual | the number of residual degrees of freedoms, calculated using the weights. |
| edf | the (effective) number of degrees of freedom used by the model |
| n, nobs | the (effective) number of observations, calculated using the weights. (nobs is for use by stepAIC). |
| call | the matched call. |
| method | the matched method used. |
| convergence | the convergence code returned by optim. |
| niter | the number of function and gradient evaluations used by optim. |
| lp | the linear predictor (including any offset). |
| Hessian | (if Hess is true). Note that this is a numerical approximation derived from the optimization proces. |
| model | (if model is true). |

## References

polr from MASS

## See Also

optim, glm, multinom

partial_Spearman *Partial Spearman's Rank Correlation*

## Description

partial_Spearman computes the partial Spearman's rank correlation between variable $X$ and variable $Y$ adjusting for other variables, $Z$. The basic approach involves fitting a specified model of $X$ on $Z$, a specified model of $Y$ on $Z$, obtaining the probability-scale residuals from both models, and then calculating their Pearson's correlation. $X$ and $Y$ can be any orderable variables, including continuous or discrete variables. By default, partial_Spearman uses cumulative probability models (also referred as cumulative link models in literature) for both $X$ on $Z$ and $Y$ on $Z$ to preserve the rank-based nature of Spearman's correlation, since the model fit of cumulative probability models only depends on the order information of variables. However, for some specific types of variables, options of fitting parametric models are also available. See details in fit.x and fit.y

## Usage

```
partial_Spearman(
  formula,
  data,
  fit.x = "orm",
  fit.y = "orm",
  link.x = c("logit", "probit", "cloglog", "loglog", "cauchit", "logistic"),
  link.y = c("logit", "probit", "cloglog", "loglog", "cauchit", "logistic"),
  subset,
  na.action = getOption("na.action"),
  fisher = TRUE,
  conf.int = 0.95
)
```

## Arguments

| | |
|---|---|
| formula | an object of class Formula (or one that can be coerced to that class): a symbolic description of the model to be fitted. The details of model specification are given under 'Details'. |
| data | an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which partial_Spearman is called. |
| fit.x, fit.y | the fitting functions used for the models of X or Y on Z. The default function is 'orm', which fits cumulative probability models for continuous or discrete ordinal variables. Other options include 'lm', which fits linear regression models and obtains the probability-scale residuals by assuming normality; 'lm.emp', which fits linear regression models and obtains the probability-scale residuals by empirical ranking; 'poisson', which fits Poisson models for count variables; 'nb', which fits negative binomial models for count variables; and 'logistic', which fits logistic regression models for binary variables. |

| | |
|---|---|
| link.x, link.y | the link family to be used for the ordinal model of $X$ on $Z$. Defaults to 'logit'. Other options are 'probit', 'cloglog', 'loglog', 'cauchit' and 'logistic' (equivalent with 'logit'). Used only when 'fit.x' is 'orm'. |
| subset | an optional vector specifying a subset of observations to be used in the fitting process. |
| na.action | action to take when NA present in data. |
| fisher | logical indicating whether to apply fisher transformation to compute confidence intervals and p-values for the correlation. |
| conf.int | numeric specifying confidence interval coverage. |

## Details

To compute the partial Spearman's rank correlation between $X$ and $Y$ adjusting for $Z$, 'formula' is specified as $X$ | $Y$ ~ $Z$. This indicates that models of $X$ ~ $Z$ and $Y$ ~ $Z$ will be fit.

## Value

object of 'partial_Spearman' class.

## References

Li C and Shepherd BE (2012) A new residual for ordinal outcomes. *Biometrika*. **99**: 473–480.

Shepherd BE, Li C, Liu Q (2016) Probability-scale residuals for continuous, discrete, and censored data. *The Canadian Jouranl of Statistics*. **44**:463–476.

Liu Q, Shepherd BE, Wanga V, Li C (2018) Covariate-Adjusted Spearman's Rank Correlation with Probability-Scale Residuals. *Biometrics*. **74**:595–605.

## See Also

print.partial_Spearman

## Examples

```
data(PResidData)
#### fitting cumulative probability models for both Y and W
partial_Spearman(c|w ~ z,data=PResidData)
#### fitting a cumulative probability model for W and a poisson model for c
partial_Spearman(c|w~z, fit.x="poisson",data=PResidData)
partial_Spearman(c|w~z, fit.x="poisson", fit.y="lm.emp", data=PResidData )
```

## plot.conditional_Spearman *conditional_Spearman class plot method*

### Description

conditional_Spearman class plot method

### Usage

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

### Arguments

| | |
|---|---|
| x | conditional_Spearman object |
| ... | arguments passed to plot.default |

## presid *Probability-scale residual*

### Description

presid calculates the probability-scale residual for various model function objects. Currently supported models include glm (Poisson, binomial, and gaussian families), lm in the **stats** library; survreg (Weibull, exponential, gaussian, logistic, and lognormal distributions) and coxph in the **survival** library; polr and glm.nb in the **MASS** library; and ols, cph, lrm, orm, psm, and Glm in the **rms** library.

### Usage

```
presid(object, ...)
```

### Arguments

| | |
|---|---|
| object | The model object for which the probability-scale residual is calculated |
| ... | Additional arguements passed to methods |

### Details

Probability-scale residual is $P(Y < y) - P(Y > y)$ where $y$ is the observed outcome and $Y$ is a random variable from the fitted distribution.

### Value

The probability-scale residual for the model

## References

Shepherd BE, Li C, Liu Q (2016) Probability-scale residuals for continuous, discrete, and censored data. *The Canadian Jouranl of Statistics*. **44**:463–476.

Li C and Shepherd BE (2012) A new residual for ordinal outcomes. *Biometrika*. **99**: 473–480.

## Examples

```
library(survival)
library(stats)

set.seed(100)
n <- 1000
x <- rnorm(n)
t <- rweibull(n, shape=1/3, scale=exp(x))
c <- rexp(n, 1/3)
y <- pmin(t, c)
d <- ifelse(t<=c, 1, 0)

mod.survreg <- survreg(Surv(y, d) ~ x, dist="weibull")
summary(presid(mod.survreg))
plot(x, presid(mod.survreg))

##### example for proprotional hazards model
n <- 1000
x <- rnorm(n)
beta0 <- 1
beta1 <- 0.5
t <- rexp(n, rate = exp(beta0 + beta1*x))
c <- rexp(n, rate=1)
y <- ifelse(t<c, t, c)
delta <- as.integer(t<c)

mod.coxph <- coxph(Surv(y, delta) ~ x)
presid <- presid(mod.coxph)
plot(x, presid, cex=0.4, col=delta+2)

#### example for Negative Binomial regression
library(MASS)

n <- 1000
beta0 <- 1
beta1 <- 0.5
x <- runif(n, min=-3, max=3)
y <- rnbinom(n, mu=exp(beta0 + beta1*x), size=3)

mod.glm.nb <- glm.nb(y~x)
presid <- presid(mod.glm.nb)
summary(presid)
plot(x, presid, cex=0.4)

##### example for proportional odds model
library(MASS)
```

```
n <- 1000
x  <- rnorm(n)
y  <- numeric(n)
alpha = c(-1, 0, 1, 2)
beta <- 1
py  <-  (1 + exp(- outer(alpha, beta*x, "+"))) ^ (-1)
aa = runif(n)
for(i in 1:n)
  y[i] = sum(aa[i] > py[,i])
y <-  as.factor(y)


mod.polr <- polr(y~x, method="logistic")
summary(mod.polr)
presid <- presid(mod.polr)
summary(presid)
plot(x, presid, cex=0.4)
```

---

PResidData *Example Dataset for PResiduals Package*

---

## Description

This is a dataset used in Examples Section of PResiduals package help files.

## Usage

```
PResidData
```

## Format

A data frame with 200 rows and 5 variables:

**x** an ordered categorical variable with 5 levels

**y** an ordered categorical variable with 4 levels

**z** a continuous variable

**w** a continuous variable

**c** a count variable

## Source

Simulated

## print.cobot *cobot class print method*

### Description

cobot class print method

### Usage

```
## S3 method for class 'cobot'
print(x, ...)
```

### Arguments

| | |
|---|---|
| x | cobot object |
| ... | arguments passed to print.default |

## print.cocobot *cocobot class print method*

### Description

cocobot class print method

### Usage

```
## S3 method for class 'cocobot'
print(x, ...)
```

### Arguments

| | |
|---|---|
| x | cocobot object |
| ... | arguments passed to print.default |

`print.conditional_Spearman` *conditional_Spearman class print method*

## Description

conditional_Spearman class print method

## Usage

```
## S3 method for class 'conditional_Spearman'
print(x, ...)
```

## Arguments

| | |
|---|---|
| `x` | conditional_Spearman object |
| `...` | arguments passed to print.default |

`print.partial_Spearman` *partial_Spearman class print method*

## Description

partial_Spearman class print method

## Usage

```
## S3 method for class 'partial_Spearman'
print(x, ...)
```

## Arguments

| | |
|---|---|
| `x` | partial_Spearman object |
| `...` | arguments passed to print.default |

# Index