Package ‘polycor’

August 27, 2016

Version 0.7-9
Date 2016-08-26
Title Polychoric and Polyserial Correlations
Depends R (>= 3.3.0)
Imports stats, mvtnorm, Matrix
ByteCompile yes
LazyLoad yes
Description Computes polychoric and polyserial correlations by quick "two-step" methods or ML, optionally with standard errors; tetrachoric and biserial correlations are special cases.
License GPL (>= 2)

URL https://r-forge.r-project.org/projects/polycor/, 
http://CRAN.R-project.org/package=polycor

Author John Fox [aut, cre]
Maintainer John Fox <jfox@mcmaster.ca>
Repository CRAN
Repository/R-Forge/Project polycor
Repository/R-Forge/Revision 13
Repository/R-Forge/Date/TimeStamp 2016-08-26 18:25:37
Date/Publication 2016-08-27 00:22:11

R topics documented:

hetcor ................................................................. 2
polychor ............................................................... 4
polyserial ............................................................. 6
print.polycor ......................................................... 8

Index 10
hetcor

Heterogeneous Correlation Matrix

Description

Computes a heterogenous correlation matrix, consisting of Pearson product-moment correlations
between numeric variables, polyserial correlations between numeric and ordinal variables, and poly-
choric correlations between ordinal variables.

Usage

hetcor(data, ..., ML = FALSE, std.err = TRUE, bins=4, pd=TRUE)
## S3 method for class 'data.frame'
hetcor(data, ML = FALSE, std.err = TRUE,
       use = c("complete.obs", "pairwise.complete.obs"), bins=4, pd=TRUE, ...)
## Default S3 method:
hetcor(data, ..., ML = FALSE, std.err = TRUE, bins=4, pd=TRUE)
## S3 method for class 'hetcor'
print(x, digits = max(3,getOption("digits") - 3), ...)
## S3 method for class 'hetcor'
as.matrix(x, ...)

Arguments

data a data frame consisting of factors, ordered factors, logical variables, and/or nu-
meric variables, or the first of several variables.
...
variables and/or arguments to be passed down.
ML if TRUE, compute maximum-likelihood estimates; if FALSE, compute quick two-
step estimates.
std.err if TRUE, compute standard errors.
Bins number of bins to use for continuous variables in testing bivariate normality; the
default is 4.
pd if TRUE and if the correlation matrix is not positive-definite, an attempt will
be made to adjust it to a positive-definite matrix, using the nearPD function in
the Matrix package. Note that default arguments to nearPD are used (except
corr=TRUE); for more control call nearPD directly.
use if "complete.obs", remove observations with any missing data; if "pairwise.complete.obs",
compute each correlation using all observations with valid data for that pair of
variables.
x an object of class "hetcor" to be printed, or from which to extract the correla-
tion matrix.
digits number of significant digits.
**Value**

Returns an object of class "hetcor" with the following components:

- **correlations**: the correlation matrix.
- **type**: the type of each correlation: "Pearson", "Polychoric", or "Polyserial".
- **std.errors**: the standard errors of the correlations, if requested.
- **n**: the number (or numbers) of observations on which the correlations are based.
- **tests**: p-values for tests of bivariate normality for each pair of variables.
- **NA.method**: the method by which any missing data were handled: "complete.obs" or "pairwise.complete.obs".
- **ML**: TRUE for ML estimates, FALSE for two-step estimates.

**Note**

Although the function reports standard errors for product-moment correlations, transformations (the most well known is Fisher's z-transformation) are available that make the approach to asymptotic normality much more rapid.

**Author(s)**

John Fox <jfox@mcmaster.ca>

**References**


**See Also**

polychor, polyserial, nearPD

**Examples**

```r
if(require(mvtnorm)){
  set.seed(12345)
  R <- matrix(0, 4, 4)
  R[upper.tri(R)] <- runif(6)
  diag(R) <- 1
  R <- cov2cor(t(R) %*% R)
  round(R, 4)  # population correlations
```
data <- rmvnorm(1000, rep(0, 4), R)
round(cor(data), 4)  # sample correlations
}
if(require(mvtnorm)){
x1 <- data[,1]
x2 <- data[,2]
y1 <- cut(data[,3], c(-Inf, .75, Inf))
y2 <- cut(data[,4], c(-Inf, -1, .5, 1.5, Inf))
data <- data.frame(x1, x2, y1, y2)
hetcor(data)  # Pearson, polychoric, and polyserial correlations, 2-step est.
}
if(require(mvtnorm)){
  hetcor(x1, x2, y1, y2, ML=TRUE)  # Pearson, polychoric, polyserial correlations, ML est.
}

---

### polychor

#### Polychoric Correlation

**Description**

Computes the polychoric correlation (and its standard error) between two ordinal variables or from their contingency table, under the assumption that the ordinal variables dissect continuous latent variables that are bivariate normal. Either the maximum-likelihood estimator or a (possibly much) quicker “two-step” approximation is available. For the ML estimator, the estimates of the thresholds and the covariance matrix of the estimates are also available.

**Usage**

```r
polychor(x, y, ML = FALSE, control = list(), std.err = FALSE, maxcor=.9999)
```

**Arguments**

- **x**
  a contingency table of counts or an ordered categorical variable; the latter can be numeric, logical, a factor, or an ordered factor, but if a factor, its levels should be in proper order.

- **y**
  if x is a variable, a second ordered categorical variable.

- **ML**
  if TRUE, compute the maximum-likelihood estimate; if FALSE, the default, compute a quicker “two-step” approximation.

- **control**
  optional arguments to be passed to the optim function.

- **std.err**
  if TRUE, return the estimated variance of the correlation (for the two-step estimator) or the estimated covariance matrix (for the ML estimator) of the correlation and thresholds; the default is FALSE.

- **maxcor**
  maximum absolute correlation (to insure numerical stability).
**polycholr**

**Details**

The ML estimator is computed by maximizing the bivariate-normal likelihood with respect to the thresholds for the two variables \( \tau_x^i, i = 1, \ldots, r - 1 \); \( \tau_y^j, j = 1, \ldots, c - 1 \) and the population correlation (\( \rho \)). Here, \( r \) and \( c \) are respectively the number of levels of \( x \) and \( y \). The likelihood is maximized numerically using the `optim` function, and the covariance matrix of the estimated parameters is based on the numerical Hessian computed by `optim`.

The two-step estimator is computed by first estimating the thresholds \( \tau_x^i, i = 1, \ldots, r - 1 \) and \( \tau_y^j, i = j, \ldots, c - 1 \) separately from the marginal distribution of each variable. Then the one-dimensional likelihood for \( \rho \) is maximized numerically, using `optim` if standard errors are requested, or `optimise` if they are not. The standard error computed treats the thresholds as fixed.

**Value**

If `std.err` is `TRUE`, returns an object of class "polychor" with the following components:

- `type` set to "polychoric".
- `rho` the polychoric correlation.
- `row.cuts` estimated thresholds for the row variable (\( x \)), for the ML estimate.
- `col.cuts` estimated thresholds for the column variable (\( y \)), for the ML estimate.
- `var` the estimated variance of the correlation, or, for the ML estimate, the estimated covariance matrix of the correlation and thresholds.
- `n` the number of observations on which the correlation is based.
- `chisq` chi-square test for bivariate normality.
- `df` degrees of freedom for the test of bivariate normality.
- `ML` `TRUE` for the ML estimate, `FALSE` for the two-step estimate.

Otherwise, returns the polychoric correlation.

**Author(s)**

John Fox <jfox@mcmaster.ca>

**References**


**See Also**

`hetcor`, `polyserial`, `print.polychor`, `optim`
Examples

```r
if(require(mvtnorm)){
  set.seed(12345)
  data <- rmvnorm(1000, c(0, 0), matrix(c(1, .5, .5, 1, 2, 2), 2))
  x <- data[,1]
  y <- data[,2]
  cor(x, y) # sample correlation
}
if(require(mvtnorm)){
  x <- cut(x, c(-Inf, .75, Inf))
  y <- cut(y, c(-Inf, -1, .5, 1.5, Inf))
  polychor(x, y) # 2-step estimate
}
if(require(mvtnorm)){
  set.seed(12345)
  polychor(x, y, ML=TRUE, std.err=TRUE) # ML estimate
}
```

---

**polyserial**  
*Polyserial Correlation*

**Description**

Computes the polyserial correlation (and its standard error) between a quantitative variable and an ordinal variable, based on the assumption that the joint distribution of the quantitative variable and a latent continuous variable underlying the ordinal variable is bivariate normal. Either the maximum-likelihood estimator or a quicker “two-step” approximation is available. For the ML estimator the estimates of the thresholds and the covariance matrix of the estimates are also available.

**Usage**

```r
polyserial(x, y, ML = FALSE, control = list(), std.err = FALSE, maxcor = 0.9999, bins = 4)
```

**Arguments**

- **x**: a numerical variable.
- **y**: an ordered categorical variable; can be numeric, logical, a factor, or an ordered factor, but if a factor, its levels should be in proper order.
- **ML**: if TRUE, compute the maximum-likelihood estimate; if FALSE, the default, compute a quicker “two-step” approximation.
- **control**: optional arguments to be passed to the optim function.
- **std.err**: if TRUE, return the estimated variance of the correlation (for the two-step estimator) or the estimated covariance matrix of the correlation and thresholds (for the ML estimator); the default is FALSE.
- **maxcor**: maximum absolute correlation (to insure numerical stability).
- **bins**: the number of bins into which to dissect x for a test of bivariate normality; the default is 4.
Details
The ML estimator is computed by maximizing the bivariate-normal likelihood with respect to the thresholds for \( y (\tau_y^i, i = 1, \ldots, c - 1) \) and the population correlation (\( \rho \)). The likelihood is maximized numerically using the `optim` function, and the covariance matrix of the estimated parameters is based on the numerical Hessian computed by `optim`.

The two-step estimator is computed by first estimating the thresholds (\( \tau_y^i, i = 1, \ldots, c - 1 \)) from the marginal distribution of \( y \). Then if the standard error of \( \hat{\rho} \) is requested, the one-dimensional likelihood for \( \rho \) is maximized numerically, using `optim` if standard errors are requested; the standard error computed treats the thresholds as fixed. If the standard error isn’t request, \( \hat{\rho} \) is computed directly.

Value
If `std.err` is `TRUE`, returns an object of class "polycor" with the following components:

- `type`: set to "polyserial".
- `rho`: the polyserial correlation.
- `cuts`: estimated thresholds for the ordinal variable (\( y \)), for the ML estimator.
- `var`: the estimated variance of the correlation, or, for the ML estimator, the estimated covariance matrix of the correlation and thresholds.
- `n`: the number of observations on which the correlation is based.
- `chisq`: chi-square test for bivariate normality.
- `df`: degrees of freedom for the test of bivariate normality.
- `ML`: `TRUE` for the ML estimate, `FALSE` for the two-step estimate.

Otherwise, returns the polyserial correlation.

Author(s)
John Fox <jfox@mcmaster.ca>

References

See Also
`hetcor`, `polychor`, `print.polycor`, `optim`

Examples
```r
if(require(mvtnorm)){
  set.seed(12345)
  data <- rmvnorm(1000, c(0, 0), matrix(c(1, .5, .5, 1), 2, 2))
  x <- data[,1]
  y <- data[,2]
  cor(x, y)  # sample correlation
}
print.polycor

Print Method for polycor Objects

Description

print method for objects of class polycor, produced by polychor and polyserial.

Usage

## S3 method for class 'polycor'
print(x, digits = max(3, getOption("digits") - 3), ...)

Arguments

x an object of class polycor, as returned by polychor or polyserial.
digits number of significant digits to be printed.
... not used.

Value

Invisibly returns x; used for its side effect — i.e., printing.

Author(s)

John Fox <jfox@mcmaster.ca>

See Also

polychor, polyserial

Examples

if(require(mvtnorm)){
  set.seed(12345)
  data <- rmvnorm(1000, c(0, 0), matrix(c(1, .5, .5, 1), 2, 2))
  x <- data[,1]
  y <- data[,2]
  cor(x, y) # sample correlation
}
if(require(mvtnorm)){
  y <- cut(y, c(-Inf, -1, .5, 1.5, Inf))
  polyserial(x, y) # 2-step estimate
}
if(require(mvtnorm)){
  polyserial(x, y, ML=TRUE, std.err=TRUE) # ML estimate
}
x <- cut(x, c(-Inf, .75, Inf))
y <- cut(y, c(-Inf, -1, .5, 1.5, Inf))
polychor(x, y, ML=TRUE, std.err=TRUE)  # polychoric correlation, ML estimate
Index

*Topic methods
  print.polycor, 8

*Topic models
  hetcor, 2
  polychor, 4
  polyserial, 6

*Topic print
  print.polycor, 8

as.matrix.hetcor (hetcor), 2

hetcor, 2, 5, 7

nearPD, 2, 3

optim, 5, 7
optimise, 5

polychor, 3, 4, 7, 8
polyserial, 3, 5, 6, 8
print.hetcor (hetcor), 2
print.polycor, 5, 7, 8