

Package ‘sampleSelection’

March 5, 2012

Version 0.7-0

Date 2011/11/13

Title Sample Selection Models

Author Arne Henningsen <arne.henningsen@googlemail.com>, Ott Toomet <otoomet@ut.ee>

Maintainer Arne Henningsen <arne.henningsen@googlemail.com>

Depends R (>= 2.10), maxLik (>= 0.7-3), systemfit (>= 1.0-0)

Imports miscTools (>= 0.6-3)

Suggests VGAM, MASS, mvtnorm, plm

Description Estimation of Sample Selection Models

License GPL (>= 2)

URL <http://www.sampleSelection.org>

Repository CRAN

Date/Publication 2012-03-05 07:20:18

R topics documented:

coef.selection	2
fitted.probit	3
fitted.selection	4
heckitVcov	5
invMillsRatio	6
linearPredictors	10
model.frame.binaryChoice	11
model.frame.selection	12
model.matrix.binaryChoice	12
model.matrix.selection	13
Mroz87	14

nlswork	15
nObs.probit	17
probit	17
RandHIE	20
residuals.probit	22
residuals.selection	23
sampleSelection-deprecated	24
selection	25
summary.probit	31
summary.selection	31
vcov.selection	33

Index 35

coef.selection	<i>Extract Coefficients from Selection Models</i>
----------------	---------------------------------------------------

Description

This function extracts coefficients from sample selection models

Usage

```
## S3 method for class 'selection'
coef(object, part = "full", ...)
## S3 method for class 'summary.selection'
coef(object, part = "full", ...)
## S3 method for class 'coef.selection'
print(x, prefix = TRUE,
      digits = max(3, getOption("digits") - 3), ... )
```

Arguments

object	object of class selection or summary.selection.
part	character string indication which parts to extract: "full" for all estimated parameters (selection estimates, outcome estimates, error variance and correlation) or "outcome" for the outcome estimates only (including the coefficient of the inverse Mill's ratio in case of a two-step estimation).
x	object returned by coef.selection.
prefix	logical. Add a prefix to the names of the coefficients that indicates to which equation the coefficient belongs.
digits	numeric, (suggested) number of significant digits.
...	currently not used.

Value

`coef.selection` returns a vector of the estimated coefficients.

`coef.summary.selection` returns a matrix of the estimated coefficients, their standard errors, t-values, and p-values.

Author(s)

Arne Henningsen, Ott Toomet (<otoomet@ut.ee>)

See Also

[coefficients](#), [selection](#), [summary.selection](#), and [vcov.selection](#)

Examples

```
## Estimate a simple female wage model taking into account the labour
## force participation
data(Mroz87)
a <- heckit(lfp ~ huswage + kids5 + mtr + fatheduc + educ + city,
           log(wage) ~ educ + city, data=Mroz87)
## extract all coefficients of the model:
coef( a )

## now extract the coefficients of the outcome model only:
coef( a, part="outcome")

## extract all coefficients, standard errors, t-values
## and p-values of the model:
coef( summary( a ) )

## now extract the coefficients, standard errors, t-values
## and p-values of the outcome model only:
coef( summary( a ), part="outcome")
```

fitted.probit

Fitted values of probit models

Description

Calculate fitted values of [probit](#) models.

Usage

```
## S3 method for class 'probit'
fitted( object, ... )
```

Arguments

object an object of class probit.
 ... further arguments (currently ignored).

Value

A numeric vector of the fitted values.

Author(s)

Arne Henningsen

See Also

[fitted](#), [probit](#).

fitted.selection *Fitted Values of Selection Models*

Description

Calculate fitted values of sample selection models

Usage

```
## S3 method for class 'selection'
fitted(object, part = "outcome", ... )
```

Arguments

object object of class selection.
 part character string indication which fitted values to extract: "outcome" for the fitted values of the outcome equation(s) or "selection" for the fitted values of the selection equation.
 ... further arguments passed to other methods (e.g. [fitted.probit](#) or [fitted](#)).

Details

If the model was estimated by the 2-step method, the fitted values of an outcome equation are calculated using all regressors of this equation including the inverse Mill's ratios.

If the model was estimated by the maximum likelihood method, the fitted values are calculated using only the regressors of the respective (outcome/selection) equation. In the future, we might add the option to include expectations of the error term based on the regressors of the other (selection/outcome) equation.

Value

A numeric vector of the fitted values.

Author(s)

Arne Henningsen

See Also

[selection](#), [residuals.selection fitted](#), and [fitted.probit](#)

heckitVcov

Heckit Variance Covariance Matrix

Description

Calculate the asymptotic covariance matrix for the coefficients of a Heckit estimation

Usage

```
heckitVcov( xMat, wMat, vcovProbit, rho, delta, sigma,  
saveMemory = TRUE )
```

Arguments

xMat	model matrix of the 2nd step estimation.
wMat	model matrix of the 1st step probit estimation.
vcovProbit	variance covariance matrix of the 1st step probit estimation.
rho	the estimated ρ , see Greene (2003, p. 784).
delta	the estimated δ s, see Greene (2003, p. 784).
sigma	the estimated σ , see Greene (2003, p. 784).
saveMemory	logical. Save memory by using a different implementation of the formula? (this should not influence the results).

Details

The formula implemented in `heckitVcov` is available, e.g., in Greene (2003), last formula on page 785.

Value

the variance covariance matrix of the coefficients.

Author(s)

Arne Henningsen

References

Greene, W. H. (2003) *Econometric Analysis, Fifth Edition*, Prentice Hall.

Lee, L., G. Maddala and R. Trost (1980) Asymmetric covariance matrices of two-stage probit and two-stage tobit methods for simultaneous equations models with selectivity. *Econometrica*, 48, p. 491-503.

See Also

[heckit](#).

invMillsRatio

Inverse Mill's Ratio of probit models

Description

Calculates the 'Inverse Mill's Ratios' of univariate and bivariate probit models.

Usage

```
invMillsRatio( x, all = FALSE )
```

Arguments

x	probit model estimated by probit , glm or vglm .
all	a logical value indicating whether the inverse Mill's Ratios should be calculated for all observations.

Details

The formula to calculate the inverse Mill's ratios for univariate probit models is taken from Greene (2003, p. 785), whereas the formulas for bivariate probit models are derived in Henning and Henningsen (2005).

Value

A data frame that contains the Inverse Mill's Ratios (IMR) and the delta values (see Greene, 2003, p. 784).

If a univariate probit estimation is provided, the variables IMR1 and IMR0 are the Inverse Mill's Ratios to correct for a sample selection bias of $y = 1$ and $y = 0$, respectively. Accordingly, 'delta1' and 'delta0' are the corresponding delta values.

If a bivariate probit estimation is provided, the variables IMRa1, IMRa0, IMRb1, and IMRb0 are the Inverse Mills Ratios to correct for a sample selection bias of $y = 1$ and $y = 0$ in equations 'a' and 'b', respectively. Accordingly, 'deltaa1', 'deltaa0', 'deltab1' and 'deltab0' are the corresponding delta values.

Author(s)

Arne Henningsen

References

Greene, W. H. (2003) *Econometric Analysis, Fifth Edition*, Prentice Hall.

Henning, C.H.C.A and A. Henningsen (2005) Modeling Price Response of Farm Households in Imperfect Labor Markets in Poland: Incorporating Transaction Costs and Heterogeneity into a Farm Household Approach. Unpublished, University of Kiel, Germany.

Examples

```
## Wooldridge( 2003 ): example 17.5, page 590
data(Mroz87)
myProbit <- glm( lfp ~ nwifeinc + educ + exper + I( exper^2 ) + age +
  kids5 + kids618, family = binomial( link = "probit" ), data=Mroz87 )
Mroz87$IMR <- invMillsRatio( myProbit )$IMR1
myHeckit <- lm( log( wage ) ~ educ + exper + I( exper^2 ) + IMR,
  data = Mroz87[ Mroz87$lfp == 1, ] )

# using NO labor force participation as endogenous variable
Mroz87$noflp <- 1 - Mroz87$lfp
myProbit2 <- glm( noflp ~ nwifeinc + educ + exper + I( exper^2 ) + age +
  kids5 + kids618, family = binomial( link = "probit" ), data=Mroz87 )
all.equal( invMillsRatio( myProbit )$IMR1, invMillsRatio( myProbit2 )$IMR0 )
# should be true

# example for bivariate probit
## Not run:
library( MASS )
library( VGAM, pos = 1e7, warn.conflicts = FALSE )

nObs <- 10000

# error terms (trivariate normal)
sigma <- symMatrix( c( 2, 0.7, 1.2, 1, 0.5, 1 ) )
myData <- as.data.frame( mvrnorm( nObs, c( 0, 0, 0 ), sigma ) )
names( myData ) <- c( "e0", "e1", "e2" )

# exogenous variables (indepently normal)
myData$x0 <- rnorm( nObs )
myData$x1 <- rnorm( nObs )
myData$x2 <- rnorm( nObs )

# endogenous variables
myData$y0 <- -1.5 + 0.8 * myData$x1 + myData$e0
myData$y1 <- ( 0.3 + 0.4 * myData$x1 + 0.3 * myData$x2 + myData$e1 ) > 0
myData$y2 <- ( -0.1 + 0.6 * myData$x1 + 0.7 * myData$x2 + myData$e2 ) > 0

# bivariate probit (using rhobit transformation)
bProbit <- vglm( cbind( y1, y2 ) ~ x1 + x2, family = binom2.rho,
```

```

    data = myData )
summary( bProbit )

# bivariate probit (NOT using rprobit transformation)
bProbit2 <- vglm( cbind( y1, y2 ) ~ x1 + x2, family = binom2.rho(
  lrho = "identity" ), data = myData )
summary( bProbit2 )

# inverse Mills Ratios
imr <- invMillsRatio( bProbit )
imr2 <- invMillsRatio( bProbit2 )
all.equal( imr, imr2, tolerance = .Machine$double.eps ^ 0.25)

# tests
# E[ e0 | y1* > 0 & y2* > 0 ]
mean( myData$e0[ myData$y1 & myData$y2 ] )
mean( sigma[1,2] * imr$IMR11a + sigma[1,3] * imr$IMR11b, na.rm = TRUE )
# E[ e0 | y1* > 0 & y2* <= 0 ]
mean( myData$e0[ myData$y1 & !myData$y2 ] )
mean( sigma[1,2] * imr$IMR10a + sigma[1,3] * imr$IMR10b, na.rm = TRUE )
# E[ e0 | y1* <= 0 & y2* > 0 ]
mean( myData$e0[ !myData$y1 & myData$y2 ] )
mean( sigma[1,2] * imr$IMR01a + sigma[1,3] * imr$IMR01b, na.rm = TRUE )
# E[ e0 | y1* <= 0 & y2* <= 0 ]
mean( myData$e0[ !myData$y1 & !myData$y2 ] )
mean( sigma[1,2] * imr$IMR00a + sigma[1,3] * imr$IMR00b, na.rm = TRUE )
# E[ e0 | y1* > 0 ]
mean( myData$e0[ myData$y1 ] )
mean( sigma[1,2] * imr$IMR1X, na.rm = TRUE )
# E[ e0 | y1* <= 0 ]
mean( myData$e0[ !myData$y1 ] )
mean( sigma[1,2] * imr$IMR0X, na.rm = TRUE )
# E[ e0 | y2* > 0 ]
mean( myData$e0[ myData$y2 ] )
mean( sigma[1,3] * imr$IMRX1, na.rm = TRUE )
# E[ e0 | y2* <= 0 ]
mean( myData$e0[ !myData$y2 ] )
mean( sigma[1,3] * imr$IMRX0, na.rm = TRUE )

# estimation for y1* > 0 and y2* > 0
selection <- myData$y1 & myData$y2
# OLS estimation
ols11 <- lm( y0 ~ x1, data = myData, subset = selection )
summary( ols11 )
# heckman type estimation
heckit11 <- lm( y0 ~ x1 + IMR11a + IMR11b, data = cbind( myData, imr ),
  subset = selection )
summary( heckit11 )

# estimation for y1* > 0 and y2* <= 0
selection <- myData$y1 & !myData$y2
# OLS estimation
ols10 <- lm( y0 ~ x1, data = myData, subset = selection )

```

```
summary( ols10 )
# heckman type estimation
heckit10 <- lm( y0 ~ x1 + IMR10a + IMR10b, data = cbind( myData, imr ),
  subset = selection )
summary( heckit10 )

# estimation for  $y1^* \leq 0$  and  $y2^* > 0$ 
selection <- !myData$y1 & myData$y2
# OLS estimation
ols01 <- lm( y0 ~ x1, data = myData, subset = selection )
summary( ols01 )
# heckman type estimation
heckit01 <- lm( y0 ~ x1 + IMR01a + IMR01b, data = cbind( myData, imr ),
  subset = selection )
summary( heckit01 )

# estimation for  $y1^* \leq 0$  and  $y2^* \leq 0$ 
selection <- !myData$y1 & !myData$y2
# OLS estimation
ols00 <- lm( y0 ~ x1, data = myData, subset = selection )
summary( ols00 )
# heckman type estimation
heckit00 <- lm( y0 ~ x1 + IMR00a + IMR00b, data = cbind( myData, imr ),
  subset = selection )
summary( heckit00 )

# estimation for  $y1^* > 0$ 
selection <- myData$y1
# OLS estimation
ols1X <- lm( y0 ~ x1, data = myData, subset = selection )
summary( ols1X )
# heckman type estimation
heckit1X <- lm( y0 ~ x1 + IMR1X, data = cbind( myData, imr ),
  subset = selection )
summary( heckit1X )

# estimation for  $y1^* \leq 0$ 
selection <- !myData$y1
# OLS estimation
ols0X <- lm( y0 ~ x1, data = myData, subset = selection )
summary( ols0X )
# heckman type estimation
heckit0X <- lm( y0 ~ x1 + IMR0X, data = cbind( myData, imr ),
  subset = selection )
summary( heckit0X )

# estimation for  $y2^* > 0$ 
selection <- myData$y2
# OLS estimation
olsX1 <- lm( y0 ~ x1, data = myData, subset = selection )
summary( olsX1 )
# heckman type estimation
heckitX1 <- lm( y0 ~ x1 + IMRX1, data = cbind( myData, imr ),
```

```
subset = selection )
summary( heckitX1 )

# estimation for y2* <= 0
selection <- !myData$y2
# OLS estimation
olsX0 <- lm( y0 ~ x1, data = myData, subset = selection )
summary( olsX0 )
# heckman type estimation
heckitX0 <- lm( y0 ~ x1 + IMRX0, data = cbind( myData, imr ),
subset = selection )
summary( heckitX0 )

## End(Not run)
```

linearPredictors	<i>Calculates linear predictors for different models</i>
------------------	----------------------------------------------------------

Description

Calculates the (unobservable) linear predictors for probability models.

Usage

```
linearPredictors(x, ...)
```

Arguments

x	model of an appropriate class
...	other arguments depending on the method

Details

It is a generic function with a method for 'probit'.

Value

A matrix with nrow equal to the number of observations and one column: the linear predictors for observations

Author(s)

Ott Toomet <otoomet@ut.ee>

Examples

```
data(Mroz87)
Mroz87$kids <- ( Mroz87$kids5 + Mroz87$kids618 > 0 )
a <- probit(lfp ~ kids + educ + hushrs + huseduc + huswage + mtr +
  motheduc, data=Mroz87)
b <- linearPredictors(a)
cor(Mroz87$lfp, b) # should be positive and highly significant
```

model.frame.binaryChoice

Data of Binary Choice Models

Description

Return the variables used for estimating a binary choice model

Usage

```
## S3 method for class 'binaryChoice'
model.frame( formula, ... )
```

Arguments

formula object of class binaryChoice.
... further arguments passed to other methods (e.g. [model.frame](#)).

Value

A data.frame containing all variables used for the estimation.

Author(s)

Arne Henningsen, Ott Toomet <otoomet@ut.ee>

See Also

[probit](#), and [model.frame](#)

model.frame.selection *Data of Selection Models*

Description

Return the variables used for estimating a sample selection model

Usage

```
## S3 method for class 'selection'  
model.frame(formula, ... )
```

Arguments

formula object of class selection.
... further arguments passed to other methods (e.g. [model.frame](#) or [model.frame.binaryChoice](#)).

Value

A data.frame containing all variables used for the estimation. The “terms” attribute contains the terms for the selection equation.

Author(s)

Arne Henningsen

See Also

[selection](#), [model.frame](#), and [model.frame.binaryChoice](#)

model.matrix.binaryChoice
Design Matrix of Binary Choice Models

Description

Create design matrix of binary choice models

Usage

```
## S3 method for class 'binaryChoice'  
model.matrix( object, ... )
```

Arguments

object object of class binaryChoice.
... currently not used.

Value

The design matrix of binary choice models.

Author(s)

Arne Henningsen, Ott Toomet <otoomet@ut.ee>

See Also

[binaryChoice](#), [model.matrix](#), and [model.frame.binaryChoice](#)

model.matrix.selection

Design Matrix of Selection Models

Description

Create design matrix of sample selection models

Usage

```
## S3 method for class 'selection'  
model.matrix(object, part = "outcome", ... )
```

Arguments

object object of class selection.
part character string indication which design matrix/matrices to extract: "outcome" for the design matrix/matrices of the outcome equation(s) or "selection" for the design matrix of the selection equation.
... further arguments passed to other methods (e.g. [model.matrix.binaryChoice](#) or [model.matrix](#)).

Value

If argument part is "selection", the design matrix of the selection equation is returned.

If argument part is "outcome", the design matrix of the outcome equation of a tobit-2 model or a list that contains the design matrices of the (two) outcome equations of a tobit-5 model is returned. All values of observations (rows of the design matrix) that were not selected for estimating the corresponding outcome equation are set to NA.

Author(s)

Arne Henningsen

See Also

[selection](#), [model.matrix](#), and [model.matrix.binaryChoice](#)

Mroz87

U.S. Women's Labor Force Participation

Description

The Mroz87 data frame contains data about 753 married women. These data are collected within the "Panel Study of Income Dynamics" (PSID). Of the 753 observations, the first 428 are for women with positive hours worked in 1975, while the remaining 325 observations are for women who did not work for pay in 1975. A more complete discussion of the data is found in Mroz (1987), Appendix 1.

Usage

```
data(Mroz87)
```

Format

This data frame contains the following columns:

lfp Dummy variable for labor-force participation.

hours Wife's hours of work in 1975.

kids5 Number of children 5 years old or younger.

kids618 Number of children 6 to 18 years old.

age Wife's age.

educ Wife's educational attainment, in years.

wage Wife's average hourly earnings, in 1975 dollars.

repwage Wife's wage reported at the time of the 1976 interview.

hushrs Husband's hours worked in 1975.

husage Husband's age.

huseduc Husband's educational attainment, in years.

huswage Husband's wage, in 1975 dollars.

faminc Family income, in 1975 dollars.

mtr Marginal tax rate facing the wife.

motheduc Wife's mother's educational attainment, in years.

fatheduc Wife's father's educational attainment, in years.

unem Unemployment rate in county of residence, in percentage points.

city Dummy variable = 1 if live in large city, else 0.

exper Actual years of wife's previous labor market experience.

nwifeinc Non-wife income.

wifecoll Dummy variable for wife's college attendance.

huscoll Dummy variable for husband's college attendance.

Source

Mroz, T. A. (1987) The sensitivity of an empirical model of married women's hours of work to economic and statistical assumptions. *Econometrica* **55**, 765–799.

PSID Staff, The Panel Study of Income Dynamics, Institute for Social Research Panel Study of Income Dynamics, University of Michigan, <http://psidonline.isr.umich.edu>.

References

Fox, J. (2004) car: The Companion to Applied Regression Library - An R and S-PLUS Companion to Applied Regression, <http://socserv.mcmaster.ca/jfox/Books/Companion/car.html>.

nlswork

National Longitudinal Survey of Young Working Women

Description

The nlswork data frame contains data about 4711 young working women who had an age of 14–26 years in 1968. These data are collected within the "National Longitudinal Survey" over the years 1968-1988 (with gaps). There are 28534 observations in total.

Usage

```
data(nlswork)
```

Format

This data frame contains the following columns:

idcode NLS ID.

year interview year.

birth_yr birth year.

age age in current year.

race 1=white, 2=black, 3=other.

msp 1 if married, spouse present.

nev_mar 1 if never married.

grade current grade completed.

collgrad 1 if college graduate.
not_smsa 1 if not SMSA.
c_city 1 if central city.
south 1 if south.
ind_code industry of employment.
occ_code occupation.
union 1 if union.
wks_ue weeks unemployed last year.
tll_exp total work experience.
tenure job tenure, in years.
hours usual hours worked.
wks_work weeks worked last year.
ln_wage ln(wage/GNP deflator).

Details

Two different versions of this data set are available on the internet. They are slightly different: The variable `wks_work` (weeks worked last year) is 101 in this version (from Stata), but NA in the version provided by the Boston College for the observation with `idcode = 1` and `year = 83`. Moreover, this variable is NA in this version (from Stata), but 104 in the version provided by the Boston College for the observation with `idcode = 2` and `year = 87`.

Source

Datasets for Stata Longitudinal/Panel-Data Reference Manual, Release 10: National Longitudinal Survey. Young Women 14-26 years of age in 1968, <http://www.stata-press.com/data/r10/nlswork.dta>.

References

Boston College, National Longitudinal Survey. Young Women 14-26 years of age in 1968, <http://fmwww.bc.edu/ec-p/data/stata/nlswork.dta>.

Examples

```
data( "nlswork" )
library( "plm" )
nlswork <- plm.data( nlswork, c( "idcode", "year" ) )
plmResult <- plm( ln_wage ~ union + age + grade + not_smsa + south + occ_code,
  data = nlswork, model = "random" )
```

nObs.probit	<i>Number of Observations of Probit Models</i>
-------------	------------------------------------------------

Description

Extract the number of observations from `probit` models.

Usage

```
## S3 method for class 'probit'
nObs( x, ... )
```

Arguments

`x` an object of class `probit`.
`...` further arguments (currently ignored).

Value

A numeric scalar: the number of observations.

Author(s)

Arne Henningsen

See Also

`probit`, `nObs`.

<code>probit</code>	<i>Binary choice models.</i>
---------------------	------------------------------

Description

Binary Choice models. These models are estimated by `binaryChoice`, intended to be called by wrappers like `probit`.

Usage

```
probit(formula, ...)
binaryChoice(formula, subset, na.action, start = NULL, data = sys.frame(sys.parent()),
             x=FALSE, y = FALSE, model = FALSE, method="ML",
             userLogLik=NULL,
             cdfLower, cdfUpper=function(x) 1 - cdfLower(x),
             logCdfLower=NULL, logCdfUpper=NULL,
             pdf, logPdf=NULL, gradPdf,
             maxMethod="Newton-Raphson",
             ... )
```

Arguments

formula	a symbolic description of the model to be fit, in the form <code>response ~ explanatory variables</code> (see also details).
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 'NA's. The default is set by the 'na.action' setting of 'options', and is 'na.fail' if that is unset. The 'factory-fresh' default is 'na.omit'. Another possible value is 'NULL', no action. Value 'na.exclude' can be useful.
start	initial value of parameters.
data	an optional data frame containing the variables in the model. If not found in data, the variables are taken from <code>environment(formula)</code> , typically the environment from which <code>probit</code> is called.
x, y, model	logicals. If TRUE the corresponding components of the fit (the model matrix, the response, the model frame) are returned.
method	the method to use; for fitting, currently only <code>method = "ML"</code> (Maximum Likelihood) is supported; <code>method = "model.frame"</code> returns the model frame (the same as with <code>model = TRUE</code> , see below).
userLogLik	log-likelihood function. A function of the parameter to be estimated, which computes the log likelihood. If supplied, it will be used instead of <code>cdfLower</code> and similar parameters. This allows user to fine-tune the likelihood function such as introducing robust approximations. It might return the corresponding gradient and Hessian as approximations, see maxNR .
<code>cdfLower</code> , <code>cdfUpper</code> , <code>pdf</code> , <code>gradPdf</code>	function, lower and upper tail of the cumulative distribution function of the disturbance term, corresponding probability density function, and gradient of the density function. These functions must take a numeric vector as the argument, and return numeric vector of the probability/gradient values.
<code>logCdfLower</code> , <code>logCdfUpper</code> , <code>logPdf</code>	logs of the corresponding functions. Providing these may improve precision in extreme tail. If not provided, simply logs are taken of the corresponding non-log values.
<code>maxMethod</code>	character, a maximisation method supported by maxLik . This is only useful if using a user-supplied likelihood function.
...	further arguments for <code>binaryChoice</code> and maxLik .

Details

The dependent variable for the binary choice models must have exactly two levels (e.g. '0' and '1', 'FALSE' and 'TRUE', or 'no' and 'yes'). Internally, the first level is always coded '0' ('failure') and the second level as '1' ('success'), no matter of the actual value. However, by default the levels are ordered alphabetically and this makes puts '1' after '0', 'TRUE' after 'FALSE' and 'yes' after 'no'.

Via the distribution function parameters, `binaryChoice` supports generic latent linear index binary choice models with additive disturbance terms. It is intended to be called by wrappers like `probit`.

However, it is also visible in the namespace as the user may want to implement her own models using another distribution of the disturbance term.

The model is estimated using Maximum Likelihood and Newton-Raphson optimizer.

`probit` implements an outlier-robust log-likelihood (Demidenko, 2001). In case of large outliers the analytic Hessian is singular while Fisher scoring approximation (used, for instance, by `glm`) is invertible. Those values are not reliable in case of outliers.

No attempt is made to establish the existence of the estimator.

Value

An object of class "binaryChoice". It is a list with following components:

LRT	Likelihood ration test. The full model is tested against H0: the parameters (besides constant) have no effect on the result. This is a list with components <ul style="list-style-type: none"> LRTThe LRT value dfDegrees of freedom for LRT (= df of the model - 1) LRT is distributed by $\chi^2(df)$ under H0.
param	A list with following background information: <ul style="list-style-type: none"> nParamNumber of parameters of the model including constant nObsNumber of the observations N1Number of observations with non-zero (true) response N0Number of observations with zero (false) response
df	Number of free parameters
x	if requested, the model matrix used.
y	if requested, the model response used. The response is represented internally as 0/1 integer vector.
model	the model frame, only if <code>model = TRUE</code> or <code>method = "model.frame"</code> .
na.action	information returned by <code>model.frame</code> on the special handling of NA s.

Other components are inherited from `maxLik`.

`probit` adds class "probit" and following components to the "binaryChoice" object:

family	the family object used (<code>binomial</code> with <code>link="probit"</code>)
--------	----------------------------------------------------------------------------------

Author(s)

Ott Toomet <otoomet@ut.ee>

References

Demidenko, Eugene (2001) "Computational aspects of probit model", *Mathematical Communications* 6, 233-247

See Also

`maxLik` for ready-packaged likelihood maximisation routines and methods, `glm` for generalised linear models, including `probit`, `binomial`.

Examples

```
## A simple MC trial: note probit assumes normal errors
x <- runif(100)
e <- 0.5*rnorm(100)
y <- x + e
summary(probit((y > 0) ~ x))
## female labour force participation probability
data(Mroz87)
Mroz87$kids <- Mroz87$kids5 > 0 | Mroz87$kids618 > 0
Mroz87$age30.39 <- Mroz87$age < 40
Mroz87$age50.60 <- Mroz87$age >= 50
summary(probit(lfp ~ kids + age30.39 + age50.60 + educ + hushrs +
              huseduc + huswage + mtr + motheduc, data=Mroz87))
```

 RandHIE

RAND Health Insurance Experiment

Description

'The RAND Health Insurance Experiment (RAND HIE) was a comprehensive study of health care cost, utilization and outcome in the United States. It is the only randomized study of health insurance, and the only study which can give definitive evidence as to the causal effects of different health insurance plans. [...] Although the fieldwork of the study was conducted between 1974 and 1982, the results are still highly relevant, since RAND HIE is the only study which can make causal statements.' (Wikipedia, RAND Health Insurance Experiment, http://en.wikipedia.org/w/index.php?title=RAND_Health_Insurance_Experiment&oldid=110166949, accessed April 8, 2007).

Usage

```
data(RandHIE)
```

Format

This data frame contains the following columns:

plan HIE plan number.

site Participant's place of residence when the participant was initially enrolled.

coins Coinsurance rate.

tookphys Took baseline physical.

year Study year.

zper Person identifier.

black 1 if race of household head is black.

income Family income.

xage Age in years.

female 1 if person is female.

- educdec** Education of household head in years.
- time** Time eligible during the year.
- outpdol** Outpatient expenses: all covered outpatient medical services excluding dental care, outpatient psychotherapy, outpatient drugs or supplies.
- drugdol** Drug expenses: all covered outpatient and dental drugs.
- suppdol** Supply expenses: all covered outpatient supplies including dental.
- mentdol** Psychotherapy expenses: all covered outpatient psychotherapy services including injections excluding charges for visits in excess of 52 per year, prescription drugs, and inpatient care.
- inpdol** Inpatient expenses: all covered inpatient expenses in a hospital, mental hospital, or nursing home, excluding outpatient care and renal dialysis.
- meddol** Medical expenses: all covered inpatient and outpatient services, including drugs, supplies, and inpatient costs of newborns excluding dental care and outpatient psychotherapy.
- totadm** Hospital admissions: annual number of covered hospitalizations.
- inpmis** Incomplete Hospital Records: missing inpatient records.
- mentvis** Psychotherapy visits: indicates the annual number of outpatient visits for psychotherapy. It includes billed visits only. The limit was 52 covered visits per person per year. The count includes an initial visit to a psychiatrist or psychologist.
- mdvis** Face-to-Face visits to physicians: annual covered outpatient visits with physician providers (excludes dental, psychotherapy, and radiology/anesthesiology/pathology-only visits).
- notmdvis** Face-to-Face visits to nonphysicians: annual covered outpatient visits with nonphysician providers such as speech and physical therapists, chiropractors, podiatrists, acupuncturists, Christian Science etc. (excludes dental, healers, psychotherapy, and radiology/anesthesiology/pathology-only visits).
- num** Family size.
- mhi** Mental health index.
- disea** Number of chronic diseases.
- physlm** Physical limitations.
- ghindx** General health index.
- mdeoff** Maximum expenditure offer.
- pioff** Participation incentive payment.
- child** 1 if age is less than 18 years.
- fchild** female * child.
- lfam** log of num (family size).
- lpi** log of pioff (participation incentive payment).
- idp** 1 if individual deductible plan.
- logc** $\log(\text{coins}+1)$.
- fmde** 0 if $\text{idp}=1$, $\ln(\max(1, \text{mdeoff}/(0.01*\text{coins})))$ otherwise.
- hlthg** 1 if self-rated health is good – baseline is excellent self-rated health.
- hlthf** 1 if self-rated health is fair – baseline is excellent self-rated health.

hlthp 1 if self-rated health is poor – baseline is excellent self-rated health.

xghindx ghindx (general health index) with imputations of missing values.

linc log of income (family income).

lnum log of num (family size).

lnmeddol log of meddol (medical expenses).

binexp 1 if meddol > 0.

Source

Data sets of Cameron and Trivedi (2005), <http://cameron.econ.ucdavis.edu/mmabook/mmadata.html>.

Additional information of variables from Table 20.4 of Cameron and Trivedi (2005) and from Newhouse (1999).

References

Cameron, A. C. and Trivedi, P. K. (2005) *Microeconometrics: Methods and Applications*, Cambridge University Press.

Newhouse, J. P. (1999) *RAND Health Insurance Experiment [in Metropolitan and Non-Metropolitan Areas of the United States], 1974–1982*, ICPSR Inter-university Consortium for Political and Social Research, Aggregated Claims Series, Volume 1: Codebook for Fee-for-Service Annual Expenditures and Visit Counts, ICPSR 6439.

Wikipedia, *RAND Health Insurance Experiment*, http://en.wikipedia.org/wiki/RAND_Health_Insurance_Experiment.

residuals.probit	<i>Residuals of probit models</i>
------------------	-----------------------------------

Description

Calculate residuals of `probit` models.

Usage

```
## S3 method for class 'probit'
residuals( object, type = "deviance", ... )
```

Arguments

<code>object</code>	an object of class <code>probit</code> .
<code>type</code>	the type of residuals which should be returned. The alternatives are: "deviance" (default), "pearson", and "response" (see details).
<code>...</code>	further arguments (currently ignored).

Details

The residuals are calculated with following formulas:

Response residuals: $r_i = y_i - \hat{y}_i$

Pearson residuals: $r_i = (y_i - \hat{y}_i) / \sqrt{\hat{y}_i(1 - \hat{y}_i)}$

Deviance residuals: $r_i = \sqrt{-2 \log(\hat{y}_i)}$ if $y_i = 1$, $r_i = -\sqrt{-2 \log(1 - \hat{y}_i)}$ if $y_i = 0$

Here, r_i is the i th residual, y_i is the i th response, $\hat{y}_i = \Phi(x_i' \hat{\beta})$ is the estimated probability that y_i is one, Φ is the cumulative distribution function of the standard normal distribution, x_i is the vector of regressors of the i th observation, and $\hat{\beta}$ is the vector of estimated coefficients.

More details are available in Davison & Snell (1991).

Value

A numeric vector of the residuals.

Author(s)

Arne Henningsen

References

Davison, A. C. and Snell, E. J. (1991) *Residuals and diagnostics*. In: *Statistical Theory and Modelling*. In Honour of Sir David Cox, edited by Hinkley, D. V., Reid, N. and Snell, E. J., Chapman & Hall, London.

See Also

[probit](#), [residuals](#), [residuals.glm](#).

residuals.selection *Residuals of Selection Models*

Description

Calculate residuals of sample selection models

Usage

```
## S3 method for class 'selection'  
residuals(object, part = "outcome",  
          type = "deviance", ... )
```

Arguments

object	object of class selection.
part	character string indication which residuals to extract: "outcome" for the fitted values of the outcome equation(s) or "selection" for the fitted values of the selection equation.
type	the type of residuals of the selection equation. The alternatives are: "deviance" (default), "pearson", and "response" (see residuals.probit).
...	further arguments passed to other methods (e.g. residuals.probit or residuals).

Details

The calculation of the fitted values that are used to calculate the residuals is described in the details section of the documentation of [fitted.selection](#).

Value

A numeric vector of the residuals.

Author(s)

Arne Henningsen

See Also

[selection](#), [fitted.selection](#), [residuals](#), and [residuals.probit](#)

sampleSelection-deprecated

Deprecated Functions in the sampleSelection package

Description

These functions are provided for compatibility with older versions of micEcon/sampleSelection only, and may be defunct as soon as the next release.

Usage

```
tobit2(selection, formula, data = sys.frame(sys.parent()),
method="ml",
      start=NULL, print.level=0,
      y1=FALSE, z=FALSE, y2=FALSE, x=FALSE, model=FALSE,
      ...)
```

Arguments

selection	a symbolic formula for the selection equation. The response must be binary variable where 0 corresponds to invisibility of the outcome and 1 to visibility.
formula	a symbolic formula for the equation of interest.
data	an optional data frame containing the variables in the model. By default the variables are taken from environment(formula), typically the environment from which the function is called
method	the method of calculation: Maximum Likelihood ("ml") or Heckman 2-step ("2step")
start	numeric, initial values of the parameters. The order is as follows: gamma (selection equation), beta (the equation of interest), sigma, rho (distribution of the error terms) for tobit2;
print.level	information about calculations. 0 – nothing is printed, bigger numbers give more information. print.level is sent further to the maximisation algorithm, see maxNR .
y1, z, y2, x, model	logicals. If TRUE the corresponding components of the fit (the selection response, the selection model matrix, the equation response, the equation model matrix and both model frames) are returned.
...	further arguments to the maximisation algorithm, see maxNR .

Details

The original help page for these functions is often available at `help("oldName-deprecated")` (note the quotes).

Function `tobit2` are for historical compatibility with old versions of `micEcon/sampleSelection`. Use [selection](#) instead.

See Also

[Deprecated](#), [selection](#)

selection

Heckman-style selection models

Description

This is the frontend for estimating Heckman-style selection models either with one or two outcomes (also known as generalized tobit models). It supports binary outcomes in one outcome equation case.

For model specification and more details, see Henningsen and Toomet (2008) and the included vignette “Sample Selection Models”.

Usage

```
selection(selection, outcome, data = sys.frame(sys.parent()),
  subset, method = "ml", start = NULL,
  ys = FALSE, xs = FALSE, yo = FALSE, xo = FALSE,
  mfs = FALSE, mfo = FALSE, print.level = 0, ...)

heckit( selection, outcome, data = sys.frame(sys.parent()),
  method = "2step", ... )
```

Arguments

<code>selection</code>	formula, the selection equation.
<code>outcome</code>	the outcome equation(s). Either a single equation (for tobit 2 models), or a list of two equations (tobit 5 models).
<code>data</code>	an optional data frame, list or environment (or object coercible by <code>as.data.frame</code> to a data frame) containing the variables in the model. If not found in <code>data</code> , the variables are taken from <code>environment(formula)</code> , typically the environment from which <code>selection</code> is called.
<code>subset</code>	an optional index vector specifying a subset of observations to be used in the fitting process.
<code>method</code>	how to estimate the model. Either "ml" for Maximum Likelihood, "2step" for 2-step estimation, or "model.frame" for returning the model frame (only).
<code>start</code>	vector, initial values for the ML estimation. If <code>start</code> does not have names, names are constructed based on the model frame.
<code>ys, yo, xs, xo, mfs, mfo</code>	logicals. If true, the response (y), model matrix (x) or the model frame (mf) of the selection (s) or outcome (o) equation(s) are returned.
<code>print.level</code>	integer. Various debugging information, higher value gives more information.
<code>...</code>	additional parameters for the corresponding fitting functions tobit2fit , tobit5fit , heckit2fit , and heckit5fit .

Details

The endogenous variable of the argument 'selection' must have exactly two levels (e.g. 'FALSE' and 'TRUE', or '0' and '1'). By default the levels are sorted in increasing order ('FALSE' is before 'TRUE', and '0' is before '1'). This also applies for the binary outcome equation. For continuous-outcome cases, the dependent variable(s) should be numeric.

For tobit-2 (sample selection) models, only those observations are included in the second step estimation (argument 'outcome'), where this variable equals the second element of its levels (e.g. 'TRUE' or '1').

For tobit-5 (switching regression) models, in the second step the first outcome equation (first element of argument 'outcome') is estimated only for those observations, where this endogenous variable of the selections equation equals the first element of its levels (e.g. 'FALSE' or '0'). The second outcome equation is estimated only for those observations, where this variable equals the second element of its levels (e.g. 'TRUE' or '1').

NA-s are allowed in the data. These are ignored if the corresponding outcome is unobserved, otherwise observations which contain NA (either in selection or outcome) are removed.

These selection models assume a known (multivariate normal) distribution of error terms. Because of this, the instruments (exclusion restrictions) are not necessary. However, if no instruments are supplied, the results are based solely on the assumption on multivariate normality. This may or may not be an appropriate assumption for a particular problem.

The (generic) function `'coef'` (`'coef.selection'`) can be used to extract the estimated coefficients. The (generic) function `'vcov'` (`'vcov.selection'`) can be used to extract the estimated variance covariance matrix of the coefficients. The (generic) function `'print'` (`'print.selection'`) can be used to print a few results. The (generic) function `'summary'` (`'summary.selection'`) can be used to obtain and print detailed results.

Value

`'selection'` returns an object of class "selection". If the model estimated by Maximum Likelihood (argument `method = "ml"`), this object is a list, which has all the components of `'maxLik'`, and in addition the elements `'twoStep'`, `'start'`, `'param'`, `termS`, `termO`, and if requested `'ys'`, `'xs'`, `'yo'`, `'xo'`, `'mfs'`, and `'mfo'`. If a tobit-2 (sample selection) model is estimated by the two-step method (argument `method = "2step"`), the returned object is list with components `'probit'`, `'coefficients'`, `'param'`, `'vcov'`, `'lm'`, `'sigma'`, `'rho'`, `'invMillsRatio'`, and `'imrDelta'`. If a tobit-2 (sample selection) model is estimated by the two-step method (argument `method = "2step"`), the returned object is list with components `'coefficients'`, `'vcov'`, `'probit'`, `'lm1'`, `'lm2'`, `'rho1'`, `'rho2'`, `'sigma1'`, `'sigma2'`, `'termsS'`, `'termsO'`, `'param'`, and if requested `'ys'`, `'xs'`, `'yo'`, `'xo'`, `'mfs'`, and `'mfo'`.

<code>probit</code>	object of class <code>'probit'</code> that contains the results of the 1st step (probit estimation) (only for two-step estimations).
<code>twoStep</code>	(only if initial values not given) results of the 2-step estimation, used for initial values
<code>start</code>	initial values for ML estimation
<code>termsS</code> , <code>termsO</code>	terms for the selection and outcome equation
<code>ys</code> , <code>xs</code> , <code>yo</code> , <code>xo</code> , <code>mfs</code> , <code>mfo</code>	response, matrix and frame of the selection- and outcome equations (as a list of two for the latter). NULL, if not requested. The response is represented internally as 0/1 integer vector with 0 denoting either the unobservable outcome (tobit 2) or the first selection (tobit 5).
<code>coefficients</code>	estimated coefficients, the complete model. coefficient for the Inverse Mills ratio is treated as a parameter ($= \rho\sigma$).
<code>vcov</code>	variance covariance matrix of the estimated coefficients.
<code>param</code>	a list with following components: <code>index</code> , a list of numeric vectors: where in the <code>coef</code> the component are located; <code>oIntercept</code> , a logical: whether the outcome equation includes intercept; <code>N0</code> , <code>N1</code> , integer, number of observations with unobserved and observed outcomes; <code>nObs</code> , integer, number of valid observations; <code>nParam</code> , integer, number of the parameters in the model (not all are independent); <code>df</code> , integer, degrees of freedom. Note this is not equal to <code>nObs - nParam</code> because of the parameters are not independent in all the cases; <code>levels</code> levels for the response of the selection equation. <code>levels[1]</code> corresponds to the outcome 1, <code>levels[2]</code> to the outcome 2.

lm, lm1, lm2	objects of class 'lm' that contain the results of the 2nd step estimation(s) of the outcome equation(s). Note: the standard errors of this estimation are biased, because they do not account for the estimation of γ in the 1st step estimation (the correct standard errors are returned by <code>summary</code> and they are contained in <code>vcov</code> component).
sigma, sigma1, sigma2	the standard error(s) of the error terms of the outcome equation(s).
rho, rho1, rho2	the estimated correlation coefficient(s) between the error term of the selection equation and the outcome equation(s).
invMillsRatio	the inverse Mills Ratios calculated from the results of the 1st step probit estimation.
imrDelta	the δ s calculated from the inverse Mills Ratios and the results of the 1st step probit estimation.

Note

The 2-step estimate of 'rho' may be outside of the $[-1, 1]$ interval. In that case the standard errors of `invMillsRatio` may be meaningless.

Author(s)

Arne Henningsen, Ott Toomet <otoomet@ut.ee>

References

- Cameron, A. C. and Trivedi, P. K. (2005) *Microeconometrics: Methods and Applications*, Cambridge University Press.
- Greene, W. H. (2003) *Econometric Analysis, Fifth Edition*, Prentice Hall.
- Heckman, J. (1976) The common structure of statistical models of truncation, sample selection and limited dependent variables and a simple estimator for such models, *Annals of Economic and Social Measurement*, 5(4), p. 475-492.
- Johnston, J. and J. DiNardo (1997) *Econometric Methods, Fourth Edition*, McGraw-Hill.
- Lee, L., G. Maddala and R. Trost (1980) Asymmetric covariance matrices of two-stage probit and two-stage tobit methods for simultaneous equations models with selectivity. *Econometrica*, 48, p. 491-503.
- Toomet, O. and A. Henningsen, (2008) Sample Selection Models in R: Package `sampleSelection`. *Journal of Statistical Software* 27(7), <http://www.jstatsoft.org/v27/i07/>
- Wooldridge, J. M. (2003) *Introductory Econometrics: A Modern Approach, 2e*, Thomson South-Western.

See Also

[lm](#), [glm](#), [binomial](#)

Examples

```

## Greene( 2003 ): example 22.8, page 786
data( Mroz87 )
Mroz87$kids <- ( Mroz87$kids5 + Mroz87$kids618 > 0 )
# Two-step estimation
summary( heckit( lfp ~ age + I( age^2 ) + faminc + kids + educ,
  wage ~ exper + I( exper^2 ) + educ + city, Mroz87 ) )
# ML estimation
summary( selection( lfp ~ age + I( age^2 ) + faminc + kids + educ,
  wage ~ exper + I( exper^2 ) + educ + city, Mroz87 ) )

## Wooldridge( 2003 ): example 17.5, page 590
data( Mroz87 )
# Two-step estimation
summary( heckit( lfp ~ nwifeinc + educ + exper + I( exper^2 ) + age +
  kids5 + kids618, log( wage ) ~ educ + exper + I( exper^2 ), Mroz87,
  method = "2step" ) )

## Example using binary outcome for selection model.
## We estimate the probability of womens' education on their
## chances to get high wage (> $5/hr in 1975 USD), using PSID data
## We use education, age, experience as explanatory variables
## and add kids, other family income, marginal tax rate, and parents
## education as exclusion restrictions.
## Note: this is slow
data(Mroz87)
m <- selection(lfp~educ + poly(age,3) + kids5 + kids618
  + huseduc + mtr + motheduc + fatheduc
  + poly(exper,2) + nwifeinc,
  wage >= 5 ~ educ + poly(age,2)
  + poly(exper,2),
  data=Mroz87)
print(summary(m))

## Cameron and Trivedi (2005): Section 16.6, page 553ff
data( RandHIE )
subsample <- RandHIE$year == 2 & !is.na( RandHIE$educdec )
selectEq <- binexp ~ logc + idp + lpi + fmde + physlm + disea +
  hlthg + hlthf + hlthp + linc + lfam + educdec + xage + female +
  child + fchild + black
outcomeEq <- lnmeddol ~ logc + idp + lpi + fmde + physlm + disea +
  hlthg + hlthf + hlthp + linc + lfam + educdec + xage + female +
  child + fchild + black
# ML estimation
cameron <- selection( selectEq, outcomeEq, data = RandHIE[ subsample, ] )
summary( cameron )

## example using random numbers
library( MASS )
nObs <- 1000
sigma <- matrix( c( 1, -0.7, -0.7, 1 ), ncol = 2 )

```

```

errorTerms <- mvrnorm( nObs, c( 0, 0 ), sigma )
myData <- data.frame( no = c( 1:nObs ), x1 = rnorm( nObs ), x2 = rnorm( nObs ),
  u1 = errorTerms[ , 1 ], u2 = errorTerms[ , 2 ] )
myData$y <- 2 + myData$x1 + myData$u1
myData$s <- ( 2 * myData$x1 + myData$x2 + myData$u2 - 0.2 ) > 0
myData$y[ !myData$s ] <- NA
myOls <- lm( y ~ x1, data = myData )
summary( myOls )
myHeckit <- heckit( s ~ x1 + x2, y ~ x1, myData, print.level = 1 )
summary( myHeckit )

## example using random numbers with IV/2SLS estimation
library( MASS )
nObs <- 1000
sigma <- matrix( c( 1, 0.5, 0.1, 0.5, 1, -0.3, 0.1, -0.3, 1 ), ncol = 3 )
errorTerms <- mvrnorm( nObs, c( 0, 0, 0 ), sigma )
myData <- data.frame( no = c( 1:nObs ), x1 = rnorm( nObs ), x2 = rnorm( nObs ),
  u1 = errorTerms[ , 1 ], u2 = errorTerms[ , 2 ], u3 = errorTerms[ , 3 ] )
myData$w <- 1 + myData$x1 + myData$u1
myData$y <- 2 + myData$w + myData$u2
myData$s <- ( 2 * myData$x1 + myData$x2 + myData$u3 - 0.2 ) > 0
myData$y[ !myData$s ] <- NA
myHeckit <- heckit( s ~ x1 + x2, y ~ w, data = myData )
summary( myHeckit ) # biased!
myHeckitIv <- heckit( s ~ x1 + x2, y ~ w, data = myData, inst = ~ x1 )
summary( myHeckitIv ) # unbiased

## tobit-5 example
N <- 500
library(mvtnorm)
vc <- diag(3)
vc[lower.tri(vc)] <- c(0.9, 0.5, 0.6)
vc[upper.tri(vc)] <- vc[lower.tri(vc)]
eps <- rmvnorm(N, rep(0, 3), vc)
xs <- runif(N)
ys <- xs + eps[,1] > 0
xo1 <- runif(N)
yo1 <- xo1 + eps[,2]
xo2 <- runif(N)
yo2 <- xo2 + eps[,3]
a <- selection(ys~xs, list(yo1 ~ xo1, yo2 ~ xo2))
summary(a)

## tobit2 example
vc <- diag(2)
vc[2,1] <- vc[1,2] <- -0.7
eps <- rmvnorm(N, rep(0, 2), vc)
xs <- runif(N)
ys <- xs + eps[,1] > 0
xo <- runif(N)
yo <- (xo + eps[,2])*(ys > 0)
a <- selection(ys~xs, yo ~xo)
summary(a)

```

summary.probit	<i>Summarizing Probit Estimations</i>
----------------	---------------------------------------

Description

Print or return a summary of a probit estimation.

Usage

```
## S3 method for class 'probit'  
summary( object, ... )  
## S3 method for class 'summary.probit'  
print( x, ... )
```

Arguments

object	an object of class probit.
x	an object of class summary.probit.
...	currently not used.

Value

The summary method returns an object of class `summary.probit`; the print method prints summary results and returns the argument invisibly.

Author(s)

Arne Henningsen

See Also

[probit](#)

summary.selection	<i>Summarizing Selection Estimations</i>
-------------------	------------------------------------------

Description

Print or return a summary of a selection estimation.

Usage

```
## S3 method for class 'selection'
summary(object, ...)
## S3 method for class 'summary.selection'
print(x,
      digits = max(3, getOption("digits") - 3),
      part = "full", ...)
```

Arguments

object	an object of class 'selection'.
x	an object of class 'summary.selection'.
part	character string: which parts of the summary to print: "full" for all the estimated parameters (probit selection, outcome estimates, correlation and residual variance), or "outcome" for the outcome results only.
digits	numeric, (suggested) number of significant digits.
...	currently not used.

Details

The variance-covariance matrix of the two-step estimator is currently implemented only for tobit-2 (sample selection) models, but not for the tobit-5 (switching regression) model.

Value

Summary methods return an object of class `summary.selection`. Print methods return the argument invisibly.

Author(s)

Arne Henningsen, Ott Toomet <otoomet@ut.ee>

See Also

[selection](#)

Examples

```
## Wooldridge( 2003 ): example 17.5, page 590
data( Mroz87 )
wooldridge <- selection( lfp ~ nwifeinc + educ + exper + I( exper^2 ) +
  age + kids5 + kids618, log( wage ) ~ educ + exper + I( exper^2 ),
  data = Mroz87, method = "2step" )

# summary of the 1st step probit estimation (Example 17.1, p. 562f)
# and the 2nd step OLS regression (example 17.5, page 590)
summary( wooldridge )

# summary of the outcome equation only
print(summary(wooldridge), part="outcome")
```

vcov.selection	<i>Extract Variance Covariance Matrix</i>
----------------	-------------------------------------------

Description

This function extracts the coefficient variance-covariance matrix from sample selection models.

Usage

```
## S3 method for class 'selection'
vcov(object, part = "full", ...)
```

Arguments

object	object of class "selection".
part	character string indication which parts of the variance-covariance matrix to extract: "full" for all estimated parameters (selection estimates, outcome estimates, error variance and correlation) or "outcome" for the outcome estimates only (including the coefficient of the inverse Mill's ratio in case of a two-step estimation).
...	currently not used.

Details

The variance-covariance matrix of a two-step estimate is currently only partly implemented. The unimplemented part of the matrix is filled with NAs.

Value

the estimated variance covariance matrix of the coefficients.

Author(s)

Arne Henningsen, Ott Toomet <otoomet@ut.ee>

See Also

[vcov](#), [selection](#), [coef.selection](#).

Examples

```
## Estimate a simple female wage model taking into account the labour
## force participation
data(Mroz87)
a <- heckit(lfp ~ huswage + kids5 + mtr + fatheduc + educ + city,
           log(wage) ~ educ + city, data=Mroz87)
## extract the full variance-covariance matrix:
vcov( a )
```

```
## now extract the variance-covariance matrix of the outcome model only:  
vcov( a, part = "outcome" )
```

Index

- *Topic **datasets**
 - Mroz87, [14](#)
 - nlswork, [15](#)
 - RandHIE, [20](#)
- *Topic **methods**
 - coef.selection, [2](#)
 - fitted.probit, [3](#)
 - fitted.selection, [4](#)
 - linearPredictors, [10](#)
 - model.frame.binaryChoice, [11](#)
 - model.frame.selection, [12](#)
 - model.matrix.binaryChoice, [12](#)
 - model.matrix.selection, [13](#)
 - nObs.probit, [17](#)
 - residuals.probit, [22](#)
 - residuals.selection, [23](#)
 - vcov.selection, [33](#)
- *Topic **misc**
 - sampleSelection-deprecated, [24](#)
- *Topic **models**
 - heckitVcov, [5](#)
 - invMillsRatio, [6](#)
 - linearPredictors, [10](#)
 - probit, [17](#)
 - selection, [25](#)
 - summary.probit, [31](#)
 - summary.selection, [31](#)
- *Topic **nonlinear**
 - probit, [17](#)
- *Topic **regression**
 - probit, [17](#)
 - selection, [25](#)
- binaryChoice, [13](#)
- binaryChoice (probit), [17](#)
- binomial, [19, 28](#)
- coef.selection, [2, 27, 33](#)
- coef.summary.selection
 - (coef.selection), [2](#)
- coefficients, [3](#)
- Deprecated, [25](#)
- fitted, [4, 5](#)
- fitted.probit, [3, 4, 5](#)
- fitted.selection, [4, 24](#)
- glm, [6, 19, 28](#)
- heckit, [6](#)
- heckit (selection), [25](#)
- heckit2fit, [26](#)
- heckit5fit, [26](#)
- heckitVcov, [5](#)
- invMillsRatio, [6](#)
- linearPredictors, [10](#)
- lm, [28](#)
- maxLik, [18, 19](#)
- maxNR, [18, 25](#)
- model.frame, [11, 12, 19](#)
- model.frame.binaryChoice, [11, 12, 13](#)
- model.frame.selection, [12](#)
- model.matrix, [13, 14](#)
- model.matrix.binaryChoice, [12, 13, 14](#)
- model.matrix.selection, [13](#)
- Mroz87, [14](#)
- nlswork, [15](#)
- nObs, [17](#)
- nObs.probit, [17](#)
- print.coef.selection (coef.selection), [2](#)
- print.selection (selection), [25](#)
- print.summary.probit (summary.probit),
[31](#)
- print.summary.selection
(summary.selection), [31](#)

probit, 3, 4, 6, 11, 17, 17, 22, 23, 31

RandHIE, 20

residuals, 23, 24

residuals.glm, 23

residuals.probit, 22, 24

residuals.selection, 5, 23

sampleSelection-deprecated, 24

selection, 3, 5, 12, 14, 24, 25, 25, 32, 33

summary.probit, 31

summary.selection, 3, 27, 31

tobit2 (sampleSelection-deprecated), 24

tobit2fit, 26

tobit5fit, 26

vcov, 33

vcov.selection, 3, 27, 33

vglm, 6