Package 'sampleSelection'

December 7, 2017

Version 1.2-0

Date 2017-12-07

Title Sample Selection Models

Author Arne Henningsen [aut, cre],
Ott Toomet [aut],
Sebastian Petersen [ctb]

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

Depends R (>= 2.10), maxLik (>= 0.7-3), stats

Imports miscTools (>= 0.6-3), systemfit (>= 1.0-0), Formula (>= 1.1-1), VGAM (>= 0.9-4), mvtnorm (>= 0.9-9994)

Suggests lmtest, Ecdat

Description Two-step
and maximum likelihood estimation
of Heckman-type sample selection models:
standard sample selection models (Tobit-2),
endogenous switching regression models (Tobit-5),
sample selection models with binary dependent outcome variable,
interval regression with sample selection (only ML estimation),
and endogenous treatment effects models.

License GPL (>= 2)

URL http://www.sampleSelection.org

NeedsCompilation no

Repository CRAN

Date/Publication 2017-12-07 15:15:06 UTC

R topics documented:

coef.selection 2
fitted.selection 3
heckitVcov 4
invMillsRatio 5

linearPredictors 9
model.frame.binaryChoice 10
model.frame.selection 11
model.matrix.binaryChoice 12
model.matrix.selection 12
Mroz87 13
nlswork 15
predict.probit 16
predict.selection 17
probit 19
probit-methods 22
RandHIE 23
residuals.probit 25
residuals.selection 26
selection 27
selection-methods 34
Smoke 35
summary.probit 36
summary.selection 37
vcov.selection 38

Index **40**

`coef.selection` *Extract Coefficients from Selection Models*

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 indicating which parts of the coefficients to extract: `"full"` for all parameters (selection estimates, outcome estimates, error variance and correlation, including parameters that were calculated based on estimated parameters), `"outcome"` for the outcome estimates only (including the coefficient of the inverse Mill's ratio in case of a two-step estimation), or `"est"` for all estimated parameters.

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

coef, selection, vcov.selection, and selection-methods.

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.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 the outcome equation are calculated using all regressors of this equation including the inverse Mill's ratios, i.e. the fitted values correspond to the conditional expectations (see predict.selection).

If the model was estimated by the maximum likelihood method, the fitted values of the outcome equation are calculated by disregarding the selection equation, i.e. the fitted values correspond to the unconditional expectations (see predict.selection).

The fitted values of the selection equation are probabilities. If the dependent variable of the outcome equation is binary, also the fitted values of the outcome equation are probabilities.

Value

A numeric vector of the fitted values.

Author(s)

Arne Henningsen

See Also

fitted, selection, residuals.selection, and selection-methods.

---

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 δs, 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) Asymetric 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$nolfp <- 1 - Mroz87$lfp
myProbit2 <- glm( nolfp ~ 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
library( "mvtnorm" )
library( "VGAM" )

nObs <- 1000

# error terms (trivariate normal)
sigma <- symMatrix( c( 2, 0.7, 1.2, 1, 0.5, 1 ) )
myData <- as.data.frame( rmvnorm( 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 rhobit transformation)
bProbit2 <- vglm( cbind( y1, y2 ) ~ x1 + x2, family = binom2.rho(
   lrho = "identitylink" ), 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* <= 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* <= 0 and y2* <= 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* <= 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 )
```

`linearPredictors` *Calculates linear predictors for different models*

Description

Calculates the (unobservable) linear predictors for probability models.

Usage

```
linearPredictors(x, ...)

## S3 method for class 'probit'
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>, Arne Henningsen

See Also

probit and probit-methods.

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

binaryChoice, probit, model.frame, model.matrix.binaryChoice, and probit-methods.

---

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

model.frame, selection, model.matrix.selection, and selection-methods.

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, probit, model.matrix, model.frame.binaryChoice, and probit-methods.

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 (tobit-2 or treatment model) or a list of two outcome matrices (tobit-5 model) is returned. All unobserved outcomes, including the corresponding explanatory variables are set to NA, even in case where valid values were supplied for estimation.

Author(s)

Arne Henningsen

See Also

model.matrix, selection, model.frame.selection, and selection-methods.

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 ResearchPanel Study of Income Dynamics, University of Michigan, http://psidonline.isr.umich.edu.

Examples

```
## 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" ) )
```

`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.

ttl_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 slighly 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" )
summary( nlswork )

## Not run:
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" )
summary( plmResult )

## End(Not run)
```

`predict.probit` *Predict method for fitted probit models*

Description

Calculate predicted values for fitted `probit` models.

Usage

```
   ## S3 method for class 'probit'
predict( object, newdata = NULL, type = "link", ... )
```

Arguments

object	a fitted object of class probit.
newdata	optionally, a data frame in which to look for variables with which to predict. If omitted, the fitted linear predictors or the fitted response values are returned.
type	the type of prediction. If this argument is "link" (the default), the predicted linear predictors are returned. If this argument is "response", the predicted probabilities are returned.
...	further arguments (currently ignored).

Value

A numeric vector of the predicted values.

Author(s)

Arne Henningsen and the R Core Team (the code of predict.probit is partly based on the code of predict.lm and predict.glm).

See Also

probit, predict, predict.glm, residuals.probit, and probit-methods.

Examples

```
## female labour force participation probability
data( "Mroz87" )
m <- probit( lfp ~ kids5 + kids618 + educ + hushrs +
   huseduc + huswage + mtr + motheduc, data=Mroz87 )
predict( m )                            # equal to linearPredictors(m)
predict( m, type = "response" )         # equal to fitted(m)
predict( m, newdata = Mroz87[ 3:9, ] )  # equal to linearPredictors(m)[3:9]
predict( m, newdata = Mroz87[ 3:9, ], type = "response" ) # equal to fitted(m)[3:9]
```

---

predict.selection *Predict method for fitted sample selection models*

Description

Calculate predicted values for sample selection models fitted with function selection.

Usage

```
   ## S3 method for class 'selection'
predict( object, newdata = NULL,
   part = ifelse( type %in% c( "unconditional", "conditional" ),
      "outcome", "selection" ),
   type = "unconditional", ... )
```

Arguments

object	a fitted object of class selection.
newdata	optionally, a data frame in which to look for variables with which to predict. If omitted, the fitted linear predictors or the fitted response values are returned.
part	character string indicating for which equation the predicted variables should be calculated: "selection" for the predicted values of the selection equation and "outcome" for the predicted values of the outcome equation; this argument is automatically chosen depending on the value of argument type.
type	if argument type is "link" and argument part is "selection", the linear predictors of the selection equation are returned; if argument type is "response" and argument part is "selection", the predicted probabilities of the selection equation are returned; if argument type is "unconditional" and argument part is "outcome", the unconditional expectations are returned, i.e. E[y|X] = X %*% beta; if argument type is "conditional" and argument part is "outcome", the conditional expectations are returned, i.e. E[y|X,Z,w=1] = X %*% beta + rho * sigma * dnorm(Z %*% gamma) / pnorm(Z %*% gamma).
...	further arguments (currently ignored).

Value

In most cases, a numeric vector of the predicted values is returned. However, there are three exceptions: (i) when predicting the unconditional expectations of a Tobit-5 model, a matrix with two columns is returned, where the two columns correspond to the two outcome equations (E[yo1] and E[yo2]); (ii) when predicting the conditional expectations of a Tobit-2 model, a matrix with two columns is returned, where the first column returns the expectations conditional on the observation being not selected (E[yo|ys=0]), while the second column returns the expectations conditional on the observation being selected (E[yo|ys=1]); (iii) when predicting the conditional expectations of a Tobit-5 model, a matrix with four columns is returned, where the first two columns return the conditional expectations of the first outcome equation (E[yo1|ys=0] and E[yo1|ys=1]) and the last two columns return the conditional expectations of the second outcome equation (E[yo2|ys=0] and E[yo2|ys=1]).

Author(s)

Arne Henningsen and 'fg nu' (the code is partly based on the code posted by 'fg nu' at http://stackoverflow.com/questions/14005788/predict-function-for-heckman-model)

See Also

selection, predict, predict.probit, residuals.selection, and selection-methods.

Examples

```
## Greene( 2003 ): example 22.8, page 786
data( Mroz87 )
Mroz87$kids  <- ( Mroz87$kids5 + Mroz87$kids618 > 0 )

# ML estimation
```

```
m <- selection( lfp ~ age + I( age^2 ) + faminc + kids + educ,
   wage ~ exper + I( exper^2 ) + educ + city, Mroz87 )

predict( m )
predict( m, type = "conditional" )
predict( m, type = "link" )
predict( m, type = "response" )
predict( m, newdata = Mroz87[ 3:9, ] )
```

`probit` *Binary choice models.*

Description

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

Usage

```
probit(formula, weights = NULL, ...)
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 `response ~ explanatory variables` (see also details).
`weights`	an optional vector of 'prior weights' to be used in the fitting process. Should be NULL or a numeric vector.
`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`	inital value of parameters.
`data`	an optional 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 `probit` 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 method = "ML" (Maximum Likelihood) is supported; method = "model.frame" returns the model frame (the same as with model = TRUE, 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 `cdfLower` 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`.

`cdfLower, cdfUpper, pdf, gradPdf` 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.

`logCdfLower, logCdfUpper, logPdf` logs of the corresponding functions. Providing these may improve precision in extreme tail. If not provided, simply logs are takes of the corresponding non-log values.

`maxMethod` character, a maximisation method supported by `maxLik`. This is only useful if using a user-supplied likelihood function.

`...` further arguments for `binaryChoice` 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' nad '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

- LRTThe LRT value
- dfDegrees of freedom for LRT (= df of the model - 1)

LRT is distributed by chi2(df) under H0.

`param` A list with following background information:

- 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.residual` degrees of freedom of the residuals.

`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 `model = TRUE` or `method = "model.frame"`.

`na.action` information returned by `model.frame` 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 (`binomial` with `link="probit"`)

Author(s)

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

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`, and `probit-methods`.

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))
```

probit-methods *probit-methods*

Description

Methods for probit models

Usage

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

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

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

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

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

Arguments

object,x	object of class probit.
digits	the minimum number of significant digits of the coefficients to be printed.
...	further arguments (currently ignored).

Details

The fitted method returns a vector of fitted values (probabilities). The logLik method returns the log likelihood value of the model. The nobs and nObs methods return the number of observations. The print method prints the call and the estimated coefficients.

Furthermore, some standard methods can be applied to probit models: the coef method returns the vector of the estimated parameters. The df.residual method returns the degrees of freedom of the residuals. The lrtest method can be used to perform likelihood-ratio tests. The stdEr method returns the vector of the standard errors of the estimated parameters. The vcov method returns the variance covariance matrix of the estimated coefficients.

The methods linearPredictors.probit, model.frame.binaryChoice, model.matrix.binaryChoice, residuals.probit, and summary.probit are described at seperate help pages.

Author(s)

Arne Henningsen

See Also

probit, summary.probit, and selection-methods.

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(coins+1)`.

fmde 0 if `idp=1`, `ln(max(1,mdeoff/(0.01*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 healt 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.

Examples

```
## 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 )
```

`residuals.probit` *Residuals of probit models*

Description

Calculate residuals of `probit` models.

Usage

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

Arguments

`object`	an object of class `probit`.
`type`	the type of residuals which should be returned. The alternatives are: "deviance" (default), "pearson", and "response" (see details).
`...`	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 ith residual, y_i is the ith 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 ith 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`, and `probit-methods`.

`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 (see section 'Details'). 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.

Argument type is only used for binary dependent variables, i.e. if argument part is equal "selection" or the dependent variable of the outcome model is binary. Hence, argument type is ignored if argument part is equal "outcome" and the dependent variable of the outcome model is numeric.

Value

A numeric vector of the residuals.

Author(s)

Arne Henningsen

See Also

residuals, selection, fitted.selection, residuals.probit, and selection-methods.

---

selection *Heckman-style selection and treatment effect 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 and interval outcomes in the single-outcome case. It also supports normal-distribution based treatment effect models.

For model specification and more details, see Toomet and Henningsen (2008) and the included vignettes "Sample Selection Models", "Interval Regression with Sample Selection", and "All-Normal Treatment Effects".

Usage

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

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

treatReg(selection, outcome,
   data=sys.frame(sys.parent()),
   mfs=TRUE, mfo=TRUE,
   ...)
```

Arguments

selection	formula, the selection equation.
outcome	the outcome equation(s). Either a single equation (for tobit 2 models), or a list of two equations (tobit 5 models).
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 selection is called.
weights	an optional vector of 'prior weights' to be used in the fitting process. Should be NULL or a numeric vector. Weights are currently only supported in type-2 models.
subset	an optional index vector specifying a subset of observations to be used in the fitting process.
method	how to estimate the model. Either "ml" for Maximum Likelihood, "2step" for 2-step estimation, or "model.frame" for returning the model frame (only).
type	model type. NULL will automatically detect the type, but one may also manually determine it. Numeric '2' and '5' are tobit-2 and tobit-5 models respectively, and character "treatment" is treatment effect model.
start	vector, initial values for the ML estimation. If start does not have names, names are constructed based on the model frame.
boundaries	an optional vector of boundaries of the intervals of the dependent variable of the outcome equation for sample selection models with interval regression of the outcome equation.
ys, yo, xs, xo, mfs, mfo	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.

printLevel, print.level
: integer. Various debugging information, higher value gives more information. The preferred option is 'printLevel'.

...
: additional parameters for the corresponding fitting functions tobit2fit, tobit5fit, heckit2fit, heckit5fit, and tobit2Bfit.

Details

The dependent variable of of the selection equation (specified by 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'). If the dependent variable of the outcome equation (specified by argument outcome) has exactly two levels, this variable is modelled as a binary variable. If argument boundaries is specified, the outcome equation is estimated as interval regression model and the dependent variable of the outcome equation must be a categorical (factor) variable or a variable of strictly positive integer values, whereas the vector specified by argument boundaries must have one more element than the number of levels or the largest integer, respectively. In all other cases, it is assumed that the dependent variable of the outcome equation is continuous and an ordinary sample selection model is estimated.

For tobit-2 (sample selection) models, only those observations are included in the second step estimation (argument 'outcome'), where the dependent variable variable of the selection equation 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 the dependent variable of the selection 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').

Treatment effect models are a version of tobit-5 models where the two outcomes are "participation" and "non-participation". treatReg takes an equal set of explanatory variables for both of these choices and assumes that the corresponding parameters are equal. In typical treatment effect model the selection outcome variable (participation decision) is included as an explanatory variable for the outcome. If this is not done, treatReg amounts to estimating two equations with correlated error structure.

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. Note also that standard errors tend to be large without a strong exclusion restriction.

If argument method is equal to "ml" (the default), the estimation is done by the maximum likelihood method, where the Newton-Raphson algorithm is used by default. Argument maxMethod (see tobit2fit) can be used to chose other algorithms for the maximisation of the (log) likelihood function.

If argument method is equal to "ml" (the default) and argument start is NULL (the default), the starting values for the maximum-likelihood (ML) estimation of a tobit-2 or tobit-5 model are obtained by an initial two-step estimation of this model.

The two-step estimation of interval-regression models with sample-selection has not yet been implemented. If no starting values for a maximum-likelihood (ML) estimation of an interval-regression model with sample-selection are specified (i.e., argument `start` is `NULL`, the default), starting values are obtained by an initial estimation of a tobit-2 model, where the dependent variable of the outcome equation is set to the mid points of the boudaries of the intervals. By default, the starting values are obtained by a maximum-likelihood (ML) estimation of the tobit-2 model, whereas the starting values for the maximum-likelihood (ML) estimation of the tobit-2 model are obtained by a 2-step estimation of the tobit-2 model. If argument `start` is set to `"2step"`, the starting values for the maximum-likelihood (ML) estimation of an interval-regression model with sample-selection are directly obtained by a 2-step estimation of the tobit-2 model (i.e., without a subsequent ML estimation of the tobit-2 model).

Methods that can be applied to objects returned by `selection()` are described on the help page `selection-methods`.

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 a 'maxLik' object, and in addition the elements 'twoStep', 'start, 'param', 'termS', 'termO', 'outcomeVar', 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 a list with components 'probit', 'coefficients', 'param', 'vcov', 'lm', 'sigma', 'rho', 'invMillsRatio', and 'imrDelta'. If a tobit-5 (switching regression) model is estimated by the two-step method (argument method = "2step"), the returned object is a 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'.

`probit` object of class 'probit' that contains the results of the 1st step (probit estimation) (only for two-step estimations).

`twoStep` (only if initial values not given) results of the 2-step estimation, used for initial values

`start` initial values for ML estimation

`termsS, termsO` terms for the selection and outcome equation

`ys, xs, yo, xo, mfs, mfo`
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).

`coefficients` estimated coefficients, the complete model. coefficient for the Inverse Mills ratio is treated as a parameter ($= \varrho\sigma$).

`vcov` variance covariance matrix of the estimated coefficients.

`param` a list with following components: `index`, a list of numeric vectors: where in the `coef` the component are located; `oIntercept`, a logical: whether the outcome equation includes intercept; `N0, N1`, integer, number of observations with unobserved and observed outcomes; `nObs`, integer, number of valid observations; `nParam`, integer, number of the parameters in the model (not all are independent); `df`, integer, degrees of freedom. Note this is not equal to `nObs - nParam`

because of the parameters are not independent in all the cases; `levels` levels for the response of the selection equation. `levels[1]` corresponds to the outcome 1, `levels[2]` 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 `summary` and they are contained in `vcov` 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 δs calculated from the inverse Mills Ratios and the results of the 1st step probit estimation.

`outcomeVar` character string indicating whether the dependent variable of the outcome equation is `"continuous"` or `"binary"`.

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) Asymetric covariance matrices of two-stage probit and two-stage tobit methods for simultaneous equations models with selectivity. *Econometrica*, 48, p. 491-503.

Petersen, S., G. Henningsen and A. Henningsen (2017) *Which Households Invest in Energy-Saving Home Improvements? Evidence From a Danish Policy Intervention*. Unpublished Manuscript. Department of Management Engineering, Technical University of Denmark.

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

summary.selection, selection-methods, probit, lm, and Mroz87 and RandHIE for further examples.

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 ) )

## 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 as explanatory variable
## and add age, kids, and non-work income as exclusion restrictions.
data(Mroz87)
m <- selection(lfp ~ educ + age + kids5 + kids618 + nwifeinc,
   wage >= 5 ~ educ, data = Mroz87 )
summary(m)


## example using random numbers
library( "mvtnorm" )
nObs <- 1000
sigma <- matrix( c( 1, -0.7, -0.7, 1 ), ncol = 2 )
errorTerms <- rmvnorm( 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( "mvtnorm" )
nObs <- 1000
sigma <- matrix( c( 1, 0.5, 0.1, 0.5, 1, -0.3, 0.1, -0.3, 1 ), ncol = 3 )
errorTerms <- rmvnorm( 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)

## Example for treatment regressions
## Estimate the effect of treatment on income
## selection outcome: treatment participation, logical (treatment)
## selection explanatory variables: age, education (years)
##   unemployment in 1974, 1975, race
## outcome: log real income 1978
## outcome explanatory variables: treatment, age, education, race.
## unemployment variables are treated as exclusion restriction
data(Treatment, package="Ecdat")
a <- treatReg(treat~poly(age,2) + educ + u74 + u75 + ethn,
              log(re78)~treat + poly(age,2) + educ + ethn,
              data=Treatment)
print(summary(a))
```

selection-methods *selection-methods*

Description

Methods for selection models

Usage

```
## S3 method for class 'selection'
logLik(object, ... )

## S3 method for class 'selection'
nobs(object, ... )

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

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

Arguments

object,x	object of class selection.
digits	the minimum number of significant digits of the coefficients to be printed.
...	further arguments (currently ignored).

Details

The logLik method returns the log likelihood value of the model. The nobs and nObs methods return the number of observations. The print method prints the call and the estimated coefficients.

Furthermore, some standard methods can be applied to selection models: The lrtest method can be used to perform likelihood-ratio tests. The stdEr method returns the vector of the standard errors of the estimated parameters.

The methods coef.selection, fitted.selection model.frame.selection, model.matrix.selection, residuals.selection, summary.selection, and vcov.selection are described at seperate help pages.

Author(s)

Arne Henningsen

See Also

selection, summary.selection, and probit-methods.

`Smoke` *Survey Responses on Smoking Behaviour*

Description

'Instructional dataset, N=807, cross-sectional individual data on smoking accompanying Introductory Econometrics: A Modern Approach, Jeffrey M. Wooldridge, South-Western College Publishing, (c) 2000 and Jeffrey M. Wooldridge, Econometric Analysis of Cross Section and Panel Data, MIT Press, (c) 2001.' (https://ideas.repec.org/p/boc/bocins/smoke.html#biblio, accessed February 27, 2017). This dataset is a subset of data used in Mullahy (1997). Data was collected in 1979 and 1980 through the Smoking Supplement to the US National Health Interview Survey.

Usage

```
data(Smoke)
```

Format

This data frame contains the following columns:

educ Years of schooling.

age Respondents age in years.

cigpric State cigarette price, cents per pack.

income Annual income in USD.

restaurn Dummy variable indicating if state restaurant smoking restrictions are in place.

smoker Dummy variable indicating if person has smoked at least one cigarette.

cigs_intervals Number of cigarettes smoked per day, coded in intervals with intervals boundaries: (0,5,10,20,50)

cigs Number of cigarettes smoked per day.

Source

Wooldridge(2009)'s dataset also available in other formats at https://ideas.repec.org/p/boc/bocins/smoke.html#biblio.

Original data used in Mullahy (1985) and Mullahy (1997).

References

Jeffrey, M. Wooldridge (2009), *Introductory Econometrics: A modern approach*, Canada: South-Western Cengage Learning.

Mullahy, John (1997), *Instrumental-Variable Estimation of Count Data Models: Applications to Models of Cigarette Smoking Behavior*, Review of Economics and Statistics 79, 596-593.

Mullahy, John (1985) *Cigarette Smoking: Habits, Health Concerns, and Heterogeneous Unobservables in a Microeconometric Analysis of Consumer Demand*, Ph.D. dissertation, University of Virginia.

Examples

```
data( Smoke )
# boundaries of the intervals
bounds <- c(0,5,10,20,50,Inf)
## Not run:
# estimation with starting values obtained by a ML estimation
# of a standard tobit-2 model with the dependent variable
# of the outcome equation equal to the mid-points of the intervals
res <- selection( smoker ~ educ + age, cigs_intervals ~ educ,
  data = Smoke, boundaries = bounds )
summary( res )

# estimation with starting values obtained by a two-step estimation
# of a standard tobit-2 model with the dependent variable
# of the outcome equation equal to the mid-points of the intervals
res2 <- selection( smoker ~ educ + age, cigs_intervals ~ educ,
  data = Smoke, boundaries = bounds, start = "2step" )
summary( res2 )

## End(Not run)

# estimation with starting values that are very close to the estimates
# (in order to reduce the execution time of running this example)
resS <- selection( smoker ~ educ + age, cigs_intervals ~ educ,
  data = Smoke, boundaries = bounds,
  start = c( 0.527, -0.0482, -0.0057, 4.23, -0.319, 2.97, 2.245 ) )
summary( resS )
```

`summary.probit` *Summarizing Probit Estimations*

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 and probit-methods.

---

summary.selection *Summarizing Selection Estimations*

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

summary, selection, and selection-methods.

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 *Extract Variance Covariance Matrix*

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 indicating which parts of the variance-covariance matrix to extract: "full" for all parameters (selection estimates, outcome estimates, error variance and correlation, including parameters that were calculated based on estimated parameters), "outcome" for the outcome estimates only (including the coefficient of the inverse Mill's ratio in case of a two-step estimation), or "est" for all estimated parameters.
...	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, and selection-methods.

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, 13
  nlswork, 15
  RandHIE, 23
  Smoke, 35
∗Topic **methods**
  coef.selection, 2
  fitted.selection, 3
  linearPredictors, 9
  model.frame.binaryChoice, 10
  model.frame.selection, 11
  model.matrix.binaryChoice, 12
  model.matrix.selection, 12
  predict.probit, 16
  predict.selection, 17
  probit-methods, 22
  residuals.probit, 25
  residuals.selection, 26
  selection-methods, 34
  vcov.selection, 38
∗Topic **models**
  heckitVcov, 4
  invMillsRatio, 5
  linearPredictors, 9
  probit, 19
  selection, 27
  summary.probit, 36
  summary.selection, 37
∗Topic **nonlinear**
  probit, 19
∗Topic **regression**
  probit, 19
  selection, 27

binaryChoice, *11, 12*
binaryChoice *(*probit*)*, 19
binomial, *21*

coef, *3*, *22*
coef.selection, 2, *34*, *39*
coef.summary.selection
        *(*coef.selection*)*, 2

df.residual, *22*

fitted, *4*
fitted.probit, *4*
fitted.probit *(*probit-methods*)*, 22
fitted.selection, 3, *27*, *34*

glm, *6*, *20, 21*

heckit, *5*
heckit *(*selection*)*, 27
heckit2fit, *29*
heckit5fit, *29*
heckitVcov, 4

invMillsRatio, 5

linearPredictors, 9
linearPredictors.probit, *22*
lm, *32*
logLik.probit *(*probit-methods*)*, 22
logLik.selection *(*selection-methods*)*, 34
lrtest, *22*, *34*

maxLik, *20, 21*
maxNR, *20*
model.frame, *10, 11*, *21*
model.frame.binaryChoice, 10, *11, 12*, *22*
model.frame.selection, 11, *13*, *34*
model.matrix, *12, 13*
model.matrix.binaryChoice, *11*, 12, *13*, *22*
model.matrix.selection, *11*, 12, *34*
Mroz87, 13, *32*

nlswork, 15
nObs.probit *(*probit-methods*)*, 22
nobs.probit *(*probit-methods*)*, 22
nObs.selection *(*selection-methods*)*, 34

nobs.selection *(selection-methods)*, 34

predict, *17, 18*
predict.glm, *17*
predict.lm, *17*
predict.probit, 16, *18*
predict.selection, *4*, 17
print.coef.selection *(coef.selection)*, 2
print.probit *(probit-methods)*, 22
print.selection *(selection-methods)*, 34
print.summary.probit *(summary.probit)*, 36
print.summary.selection *(summary.selection)*, 37
probit, *6*, *10–12*, *16, 17*, 19, *23*, *25, 26*, *32*, *37*
probit-methods, 22

RandHIE, 23, *32*
residuals, *26, 27*
residuals.glm, *26*
residuals.probit, *17*, *22*, 25, *27*
residuals.selection, *4*, *18*, 26, *34*

selection, *3, 4*, *11*, *13*, *17, 18*, *27*, 27, *34*, *38, 39*
selection-methods, 34
Smoke, 35
stdEr, *22*, *34*
summary, *38*
summary.probit, *22, 23*, 36
summary.selection, *32*, *34*, 37

tobit2Bfit, *29*
tobit2fit, *29*
tobit5fit, *29*
treatReg *(selection)*, 27

vcov, *22*, *39*
vcov.selection, *3*, *34*, 38
vglm, *6*