Package ‘Rchoice’

March 10, 2023

Type Package

Title Discrete Choice (Binary, Poisson and Ordered) Models with Random Parameters

Version 0.3-6

Date 2023-03-10

Description An implementation of simulated maximum likelihood method for the estimation of Binary (Probit and Logit), Ordered (Probit and Logit) and Poisson models with random parameters for cross-sectional and longitudinal data as presented in Sarrias (2016) <doi:10.18637/jss.v074.i10>.

Depends R (>= 4.0), Formula, maxLik

Imports sandwich, miscTools, numDeriv, memisc, msm, plm, plotrix, stats, graphics

Suggests car, lmtest, pglm, AER

License GPL (>= 2)


LazyData no

RoxygenNote 7.2.3

Encoding UTF-8

NeedsCompilation no

Author Mauricio Sarrias [aut, cre] (<https://orcid.org/0000-0001-5932-4817>), Yves Croissant [ctb], Achim Zeileis [ctb]

Maintainer Mauricio Sarrias <msarrias86@gmail.com>

Repository CRAN

Date/Publication 2023-03-10 12:30:06 UTC
Description

Data from research by Long (1990) that analyzes the scientist's level of publications.

Usage

data(Articles)

Format

A data frame with 915 observations on the following 6 variables:

art Articles during last 3 years of Ph.D.,
fem 1 if female scientist; else 0,
mar 1 if married; else 0,
kid5 Number of children 5 or younger,
phd Prestige of Ph.D. department,
ment Articles by mentor during last 3 years,
Source


Examples

```r
data(Articles)
```

### Attitudes

**Attitudes towards working mothers**

Description

In 1997 and 1989, the General Social Survey asked respondents to evaluate the following statement: "A working mother can establish just as warm and secure a relationship with her children as a mother who does not work".

Usage

```r
data(Attitudes)
```

Format

A data frame with 2293 observations on the following 10 variables:

- `warm`: 1 = Strongly disagree, 2 = disagree, 3 = agree, 4 = strongly agree,
- `yr89`: survey year: 1 = 1989; 0 = 1977,
- `male`: 1 = male; 0 = female,
- `white`: 1 = white; 0 = nonwhite,
- `age`: age in years,
- `ed`: years of education,
- `prst`: occupational prestige,

Source

Examples

```r
data(Attitudes)

library("sandwich")
bread(probit)
```

Description

Computes the “bread” of the sandwich covariance matrix for a model of class `Rchoice`

Usage

```r
## S3 method for class 'Rchoice'
bread(x, ...)
```

Arguments

- `x` a fitted model of class `Rchoice`
- `...` Other arguments when `bread` is applied to another class object.

Details

For more information see `bread` from the package `sandwich`.

Value

the covariance matrix times observations

References


Examples

```r
## Probit model
data("Workmroz")
probit <- Rchoice(lfp ~ k5 + k618 + age + wc + hc + lwg + inc, 
                   data = Workmroz , family = binomial('probit'))
summary(probit)

library("sandwich")
bread(probit)
```
effect

Get average marginal effects for heterokedastic binary models and IV probit models

Description

Obtain the average marginal effects from hetprob or ivpml class models.

Usage

effect(object, ...)

Arguments

object an object of class hetprob or ivpml.
...
Additional arguments to be passed.

Value

Estimates of the average marginal effects computed as the average for each individual.

Examples

# Data
library("AER")
data("PSID1976")
PSID1976$lfp <- as.numeric(PSID1976$participation == "yes")
PSID1976$kids <- with(PSID1976, factor((youngkids + oldkids) > 0,
  levels = c(FALSE, TRUE),
  labels = c("no", "yes")))
PSID1976$finc <- PSID1976$fincome / 10000

# Average marginal effects for heteroskedastic Probit model
labor_het <- hetprob(lfp ~ age + I(age^2) + finc + education + factor(kids) |
  factor(kids) + finc,
  data = PSID1976,
  link = "probit")

eff_labor_het <- effect(labor_het)
summary(eff_labor_het)

# Average marginal effects for IV probit model
# (nwincome is endogenous and heducation is the additional instrument)
PSID1976$nwincome <- with(PSID1976, (fincome - hours * wage)/1000)

fiml.probit <- ivpml(lfp ~ education + experience + I(experience^2) + age +
  youngkids + oldkids + nwincome |
  education + experience + I(experience^2) + age +
  youngkids + oldkids + heducation,
  data = PSID1976)
summary(effect(fiml.probit))
summary(effect(fiml.probit, asf = FALSE))

**effect.hetprob**  
*Get average marginal effects for heterokedastic binary models*

**Description**

Obtain the average marginal effects from hetprob class model.

**Usage**

```r
## S3 method for class 'hetprob'
effect(object, vcov = NULL, digits = max(3, getOption("digits") - 2), ...)

## S3 method for class 'effect.hetprob'
summary(object, ...)

## S3 method for class 'effect.hetprob'
print(x, ...)

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

**Arguments**

- `object`  
an object of class hetprob and effect.hetprob for summary and print method.
- `vcov`  
an estimate of the asymptotic variance-covariance matrix of the parameters for a hetprob object.
- `digits`  
the number of digits.
- `...`  
further arguments. Ignored.
- `x`  
an object of class effect.hetprob.

**Details**

This function allows to obtain the average marginal effects (not the marginal effects at the mean). The standard errors are computed using Delta Method.

**Value**

An object of class effect.heprob.

**Author(s)**

Mauricio Sarrias.
Examples

# Data
library("AER")
data("PSID1976")
PSID1976$lfp <- as.numeric(PSID1976$participation == "yes")
PSID1976$kids <- with(PSID1976, factor((youngkids + oldkids) > 0,
  levels = c(FALSE, TRUE),
  labels = c("no", "yes"))
PSID1976$finc <- PSID1976$fincome / 10000

# Average marginal effects for heteroskedastic Probit model
labor_het <- hetprob(lfp ~ age + I(age^2) + finc + education + factor(kids) |
  factor(kids) + finc,
  data = PSID1976,
  link = "probit")

eff_labor_het <- effect(labor_het)
summary(eff_labor_het)

---

**effect.ivpml**

Get average marginal effects for IV Probit model.

**Description**

Obtain the average marginal effects from ivpml class model.

**Usage**

```r
## S3 method for class 'ivpml'
effect(
  object,
  vcov = NULL,
  asf = TRUE,
  digits = max(3, getOption("digits") - 2),
  ...)

## S3 method for class 'effect.ivpml'
summary(object, ...)

## S3 method for class 'effect.ivpml'
print(x, ...)

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

object  an object of class ivpml and effect.ivpml for summary and print method.
vcov  an estimate of the asymptotic variance-covariance matrix of the parameters for a ivpml object.
asf  if TRUE, the average structural function is used.
digits  the number of digits.
...  further arguments. Ignored.
x  an object of class effect.ivpml.

Details

This function allows to obtain the average marginal effects (not the marginal effects at the mean). The standard errors are computed using Delta Method.

Value

An object of class effect.ivpml.

Author(s)

Mauricio Sarrias.

Examples

# Data
library("AER")
data("PSID1976")
PSID1976$lfp <- as.numeric(PSID1976$participation == "yes")
PSID1976$kids <- with(PSID1976, factor((youngkids + oldkids) > 0,
  levels = c(FALSE, TRUE),
  labels = c("no", "yes")))

# Average marginal effects for IV probit model
# (nwincome is endogenous and heducation is the additional instrument)
PSID1976$nwincome <- with(PSID1976, (fincome - hours * wage)/1000)
fiml.probit <- ivpml(lfp ~ education + experience + I(experience^2) + age +
youngkids + oldkids + nwincome |
  education + experience + I(experience^2) + age +
youngkids + oldkids + heducation,
  data = PSID1976)
summary(effect(fiml.probit))
summary(effect(fiml.probit, asf = FALSE))
Description

This a helper function to obtain the individuals’ conditional estimate of the random parameters or compensating variations.

Usage

```r
## S3 method for class 'Rchoice'
effect(object, par = NULL, effect = c("cv", "ce"), wrt = NULL, ...)
```

Arguments

- `object`: an object of class `Rchoice`,
- `par`: a string giving the name of the variable with random parameter,
- `effect`: a string indicating what should be computed: the conditional expectation of the individual coefficients "ce", or the conditional expectation of the individual compensating variations "cv",
- `wrt`: a string indicating respect to which variable the compensating variation should be computed,
- `...`: further arguments. Ignored.

Value

A named list where “mean” contains the individuals’ conditional mean for the random parameter or compensating variation, and where ‘sd.est’ contains their standard errors.

References


See Also

`Rchoice` for the estimation of different discrete choice models with individual parameters.

Examples

```r
# Poisson with random parameters
data(“Articles”)
poisson.ran <- Rchoice(art ~ fem + mar + kid5 + phd + ment, 
                       data = Articles, family = poisson, 
                       ranp = c(kid5 = "n", phd = "n", ment = "n"), 
                       R = 10)
```
## Get the individuals' conditional mean and their standard errors for ment

```r
bi.ment <- effect(poisson.ran, par = "ment", effect = "ce")
summary(bi.ment$mean)
summary(bi.ment$sd.est)
```

---

**estfun.Rchoice**

*Gradient for observations*

**Description**

It extracts the gradient for each observations evaluated at the estimated parameters for a model of class `Rchoice`.

**Usage**

```r
## S3 method for class 'Rchoice'
estfun(x, ...)
```

**Arguments**

- `x`  
a fitted model of class `Rchoice`,
- `...`  
Other arguments when `estfun` is applied to another class object

**Details**

For more information see `estfun` from package `sandwich`.

**Value**

the gradient matrix of dimension n times k

**References**

**getSummary.effect.hetprob**

*Get Model Summaries for use with "mtable" for objects of class effect.hetprob*

**Description**

A generic function to collect coefficients and summary statistics from a `effect.hetprob` object. It is used in `mtable`.

**Usage**

```r
## S3 method for class 'effect.hetprob'
getSummary(obj, alpha = 0.05, ...)
```

**Arguments**

- `obj` an `effect.hetprob` object.
- `alpha` level of the confidence intervals,
- `...` further arguments,

**Details**

For more details see package `memisc`.

---

**getSummary.effect.ivpml**

*Get Model Summaries for use with "mtable" for objects of class effect.ivpml*

**Description**

A generic function to collect coefficients and summary statistics from a `effect.ivpml` object. It is used in `mtable`.

**Usage**

```r
## S3 method for class 'effect.ivpml'
getSummary(obj, alpha = 0.05, ...)
```

**Arguments**

- `obj` an `effect.ivpml` object.
- `alpha` level of the confidence intervals,
- `...` further arguments,
Details

For more details see package \texttt{memisc}.

\begin{knitrout}
\small
\begin{verbatim}
getSummary.hetprob Get Model Summaries for use with "mtable" for objects of class hetprob
\end{verbatim}
\end{knitrout}

Description

A generic function to collect coefficients and summary statistics from a \texttt{hetprob} object. It is used in \texttt{mtable}.

Usage

\begin{verbatim}
## S3 method for class 'hetprob'
getSummary(obj, alpha = 0.05, ...)
\end{verbatim}

Arguments

- \texttt{obj} a \texttt{hetprob} object,
- \texttt{alpha} level of the confidence intervals,
- \texttt{...} further arguments,

Details

For more details see package \texttt{memisc}.

\begin{knitrout}
\small
\begin{verbatim}
getSummary.ivpml Get Model Summaries for use with "mtable" for objects of class ivpml
\end{verbatim}
\end{knitrout}

Description

A generic function to collect coefficients and summary statistics from a \texttt{ivpml} object. It is used in \texttt{mtable}.

Usage

\begin{verbatim}
## S3 method for class 'ivpml'
getSummary(obj, alpha = 0.05, ...)
\end{verbatim}

Arguments

- \texttt{obj} a \texttt{ivpml} object,
- \texttt{alpha} level of the confidence intervals,
- \texttt{...} further arguments,
Details

For more details see package **memisc**.

### Description

A generic function to collect coefficients and summary statistics from a Rchoice object. It is used in `mtable`.

### Usage

```r
## S3 method for class 'Rchoice'
getSummary(obj, alpha = 0.05, ...)
```

### Arguments

- `obj`: a Rchoice object,
- `alpha`: level of the confidence intervals,
- `...`: further arguments,

### Details

For more details see package **memisc**.

---

### Health

*German Health Care Data*

### Description

German Health Care Data, unbalanced panel.

### Usage

```r
data(Health)
```
Format

A data frame with 27326 observations on the following 27 variables:

- **id**: person identification number
- **female**: female =1, male =0
- **year**: calendar year of the observation
- **age**: age in years
- **hsat**: health satisfaction, 0 (low),...,10 (high)
- **handdum**: handicapped = 1, 0 otherwise
- **handper**: degree of handicap in percent; 0,100
- **hhinc**: household nominal monthly net income in German marks
- **hhkids**: children under age 16 in the household = 1; otherwise = 0
- **educ**: years of schooling
- **married**: married =1, otherwise = 0
- **haupts**: highest schooling degree is Hauptschul degree = 1; otherwise = 0
- **reals**: highest schooling degree is Realschul degree = 1, otherwise = 0
- **fachhs**: highest schooling degree is Polytechnical degree = 1; otherwise = 0
- **abitur**: highest schooling degree is Abitur = 1; otherwise = 0
- **univ**: highest schooling degree is university degree =1; otherwise = 0
- **working**: employed =1; otherwise = 0
- **bluec**: blue-collar employee = 1; otherwise = 0
- **whitec**: white-collar employeee =1; otherwise = 0
- **self**: self-employed = 1; otherwise = 0
- **beamt**: civil servant = 1; otherwise = 0
- **docvis**: number of doctor visits in last three months
- **hospvis**: number of hospital visits in last calendar year
- **public**: insured in public health =1; otherwise = 0
- **addon**: insured by add-on insurance =1; otherwise = 0
- **hsat2**: 40 observations on hsat recorded between 6 and 7 were changed to 7
- **newhsat**: recording of hsat, (0-2) = 0, (3-5)=1, (6-8)=2, (9)=3 (10)=4

Source


References


Examples

```r
data(Health)
```
hetprob

Estimate heteroskedastic binary (Probit or Logit) model.

Description

Estimation of binary dependent variables, either probit or logit, with heteroskedastic error terms for cross-sectional dataset.

Usage

hetprob(formula, data, link = c("probit", "logit"), ...)

## S3 method for class 'hetprob'
terms(x, ...)

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

## S3 method for class 'hetprob'
estfun(x, ...)

## S3 method for class 'hetprob'
bread(x, ...)

## S3 method for class 'hetprob'
vcov(object, eigentol = 1e-12, ...)

## S3 method for class 'hetprob'
df.residual(object, ...)

## S3 method for class 'hetprob'
coeff(object, ...)

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

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

## S3 method for class 'hetprob'
summary(object, eigentol = 1e-12, ...)

## S3 method for class 'summary.hetprob'
print(x, digits = max(3, getOption("digits") - 2), ...)

## S3 method for class 'hetprob'
predict(object, newdata = NULL, type = c("xb", "pr", "sigma"), ...)
Arguments

- **formula**: a symbolic description of the model of the form $y \sim x \mid z$ where $y$ is the binary dependent variable and $x$ and $z$ are regressors variables for the mean of the model and Insignia.
- **data**: the data of class `data.frame`.
- **link**: the assumption of the distribution of the error term. It could be either `link = "probit"` or `link = "logit"`.
- **...**: arguments passed to `maxLik`.
- **x, object**: an object of class `hetprob`.
- **eigentol**: the standard errors are only calculated if the ratio of the smallest and largest eigenvalue of the Hessian matrix is less than `eigentol`. Otherwise the Hessian is treated as singular.
- **digits**: the number of digits.
- **newdata**: optionally, a data frame in which to look for variables with which to predict.
- **type**: the type of prediction required. The default, `type = xb`, is on the linear prediction without the variance. If `type = pr`, the predicted probabilities of a positive outcome is returned. Finally, if `type = sigma` the predictions of $\sigma$ for each individual is returned.

Details

The heterokedastic binary model for cross-sectional data has the following structure:

$$y_i^* = x_i^\top \beta + \epsilon_i,$$

with

$$\text{var}(\epsilon_i|x_i, z_i) = \sigma_i^2 = \left[\exp(z_i^\top \delta)\right]^2,$$

where $y_i^*$ is the latent (unobserved) dependent variable for individual $i = 1, \ldots, N$; $x_i$ is a $K \times 1$ vector of independent variables determining the latent variable $y_i^*$ ($x$ variables in `formula`); and $\epsilon_i$ is the error term distributed either normally or logistically with $E(\epsilon_i|z_i, x_i) = 0$ and heterokedastic variance $\text{var}(\epsilon_i|x_i, z_i) = \sigma_i^2, \forall i = 1, \ldots, N$. The variance for each individual is modeled parametrically assuming that it depends on a $P \times 1$ vector observed variables $z_i$ ($z$ in `formula`), whereas $\delta$ is the vector of parameters associated with each variable. It is important to emphasize that $z_i$ does not include a constant, otherwise the parameters are not identified.

The models are estimated using the `maxLik` function from `maxLik` package using both analytic gradient and hessian (if `Hess = TRUE`). In particular, the log-likelihood function is:

$$\log L(\theta) = \sum_{i=1}^{n} \log \left\{ \left[ 1 - F\left( \frac{x_i^\top \beta}{\exp(z_i^\top \delta)} \right) \right]^{1-y_i} \left[ F\left( \frac{x_i^\top \beta}{\exp(z_i^\top \delta)} \right) \right]^{y_i} \right\}.$$
ivpml

the formula,
the model framed used,
the matched call.

**Author(s)**

Mauricio Sarrias.

**References**


**Examples**

```r
# Estimate a heteroskedastic probit and logit model
data("Health")
het.probit <- hetprob(working ~ factor(female) + factor(year) + educ + age + I(age^2) |
factor(female) + age + I(age^2),
data = Health,
link = "probit")
summary(het.probit)
het.logit <- hetprob(working ~ factor(female) + factor(year) + educ + age + I(age^2) |
factor(female) + age + I(age^2),
data = Health,
link = "logit")
summary(het.logit)
```

---

**ivpml**

Estimate Instrumental Variable Probit model by Maximum Likelihood.

**Description**

Estimation of Probit model with one endogenous and continuous variable by Maximum Likelihood.

**Usage**

```r
ivpml(formula, data, messages = TRUE, ...)
```

### S3 method for class 'ivpml'
```
terms(x, ...)
```

### S3 method for class 'ivpml'
```
terms(x, ...)
```

```r
```
## S3 method for class 'ivpml'
estfun(x, ...)

## S3 method for class 'ivpml'
bread(x, ...)

## S3 method for class 'ivpml'
vcov(object, ...)

df.residual(object, ...)

## S3 method for class 'ivpml'
coef(object, ...)

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

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

## S3 method for class 'ivpml'
summary(object, eigentol = 1e-12, ...)

## S3 method for class 'summary.ivpml'
print(x, digits = max(3, getOption("digits") - 2), ...)

## S3 method for class 'ivpml'
predict(object, newdata = NULL, type = c("xb", "pr", "stdp"), asf = TRUE, ...)

### Arguments

- **formula**: a symbolic description of the model of the form \( y \sim x | z \) where \( y \) is the binary dependent variable, \( x \) includes the exogenous and the endogenous continuous variable, and \( z \) is the complete set of instruments.
- **data**: the data of class data.frame.
- **messages**: if TRUE, then additional messages for the estimation procedure are displayed.
- **...**: arguments passed to maxLik.
- **x, object**: an object of class ivpml.
- **eigentol**: the standard errors are only calculated if the ratio of the smallest and largest eigenvalue of the Hessian matrix is less than \( \text{eigentol} \). Otherwise the Hessian is treated as singular.
- **digits**: the number of digits.
- **newdata**: optionally, a data frame in which to look for variables with which to predict.
- **type**: the type of prediction required. The default, type = xb, is on the linear prediction. If type = pr, the predicted probabilities of a positive outcome is returned.
Finally, if type = stdp the standard errors of the linear predictions for each individual is returned.

if TRUE, the average structural function is used. This option is not allowed with xb or stdp.

Details

The IV probit for cross-sectional data has the following structure:

\[ y_1^* = x_i^\top \beta + \gamma y_2 + \epsilon_i, \]

with

\[ y_2 = z_i^\top \delta + \upsilon_i, \]

where \( y_1^* \) is the latent (unobserved) dependent variable for individual \( i = 1, \ldots, N \); \( y_2 \) is the endogenous continuous variable; \( z_i \) is the vector of exogenous variables which also includes the instruments for \( y_2 \); and \((\epsilon, \upsilon)\) are normal jointly distributed.

The model is estimated using the maxLik function from maxLik package using analytic gradient.

Author(s)

Mauricio Sarrias.

References


Examples

# Data
library("AER")
data("PSID1976")
PSID1976$lfp <- as.numeric(PSID1976$participation == "yes")
PSID1976$kids <- with(PSID1976, factor((youngkids + oldkids) > 0,
levels = c(FALSE, TRUE),
labels = c("no", "yes")))

# IV probit model by MLE
# (nwincome is endogenous and heducation is the additional instrument)
PSID1976$nwincome <- with(PSID1976, (fincome - hours * wage)/1000)
firml.probit <- ivpml(lfp ~ education + experience + I(experience^2) + age +
youngkids + oldkids + nwincome |
education + experience + I(experience^2) + age +
youngkids + oldkids + heducation,
data = PSID1976)
supply(firml.probit)
plot.Rchoice

Plot the distribution of conditional expectation for random parameters.

Description

Plot the distribution of the conditional expectation of the random parameters or compensating variations for objects of class Rchoice.

Usage

```r
## S3 method for class 'Rchoice'
plot(
x, par = NULL,
effect = c("ce", "cv"),
wrt = NULL,
type = c("density", "histogram"),
adjust = 1,
main = NULL,
col = "indianred1",
breaks = 10,
ylab = NULL,
xlab = NULL,
ind = FALSE,
id = NULL,
...)
```

Arguments

- `x`: a object of class Rchoice,
- `par`: a string giving the name of the variable with random parameter,
- `effect`: a string indicating what should be plotted: the conditional expectation of the individual coefficients "ce", or the conditional expectation of the individual compensating variations "cv",
- `wrt`: a string indicating respect to which variable should be computed the compensating variation,
- `type`: a string indicating the type of distribution: it can be a histogram or a density of the conditional expectation,
- `adjust`: bandwidth for the kernel density,
- `main`: an overall title for the plot,
- `col`: color for the graph,
- `breaks`: number of breaks for the histogram if `type = "histogram"`. 
ylab a title for the y axis,
xlab a title for the x axis,
ind a boolean. If TRUE, a 95% As default, the conditional expectation of par for the first 10 individual is plotted,
id only relevant if ind is not NULL. This is a vector indicating the individuals for which the confidence intervals are plotted,
... further arguments. Ignored.

Author(s)
Mauricio Sarrias

References


See Also

Rchoice for the estimation of different discrete choice models with individual parameters.

Examples

# Poisson with random parameters
data("Articles")
poisson.ran <- Rchoice(art ~ fem + mar + kid5 + phd + ment, 
data = Articles, family = poisson, 
ranp = c(kid5 = "n", phd = "n", ment = "n"), 
R = 10)

## Plot the distribution of the conditional mean for ment
plot(poisson.ran, par = "ment", type = "density")

## Plot the conditional mean for the first 20 individuals
plot(poisson.ran, par = "ment", ind = TRUE, id = 1:20, col = "blue")

## Plot the compensating variation with respect to fem
plot(poisson.ran, par = "ment", effect = "cv", wrt = "fem", type = "histogram")
Rchoice

Estimate discrete choice model with random parameters

Description

Estimation of discrete choice models such as Binary (logit and probit), Poisson and Ordered (logit and probit) model with random coefficients for cross-sectional and panel data using simulated maximum likelihood.

Usage

Rchoice(
  formula,
  data,
  subset,
  weights,
  na.action,
  family,
  start = NULL,
  ranp = NULL,
  R = 40,
  haltons = NA,
  seed = NULL,
  correlation = FALSE,
  panel = FALSE,
  index = NULL,
  mvar = NULL,
  print.init = FALSE,
  init.ran = 0.1,
  gradient = TRUE,
  ...
)

## S3 method for class 'Rchoice'
terms(x, ...)

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

## S3 method for class 'Rchoice'
coef(object, ...)

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

## S3 method for class 'Rchoice'
residuals(object, ...)
## S3 method for class 'Rchoice'
df.residual(object, ...)

## S3 method for class 'Rchoice'
update(object, new, ...)

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

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

## S3 method for class 'Rchoice'
summary(object, ...)

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

### Arguments

**formula**  
a symbolic description of the model to be estimated. The `formula` consists in two parts. The first one is reserved for standard variables with fixed and random parameters. The second one is reserved for variables that enter in the mean of the random parameters. See for example `rFormula`,

**data**  
the data. It may be a `pdata.frame` object or an ordinary `data.frame`,

**subset**  
an optional vector specifying a subset of observations,

**weights**  
an optional vector of weights,

**na.action**  
a function which indicates what should happen when the data contains NA's,

**family**  
the distribution to be used. It might be `family = binomial("probit")` for a Probit Model, `family = binomial("logit")` for a Logit model, `family = ordinal("probit")` for an Ordered Probit Model, `family = ordinal("logit")` for a Ordered Logit Model for an Ordered Logit Model, and `family = "poisson"` for a Poisson Model,

**start**  
a vector of starting values,

**ranp**  
a named vector whose names are the random parameters and values the distribution: "n" for normal, "ln" for log-normal, "cn" for truncated normal, "u" for...
uniform, "t" for triangular, "sb" for Johnson Sb,

R the number of draws if \( \text{ranp} \) is not NULL,

haltons only relevant if \( \text{ranp} \) is not NULL. If not NULL, halton sequence is used instead of pseudo-random numbers. If \( \text{haltons} = \text{NA} \), some default values are used for the prime of the sequence and for the number of element dropped. Otherwise, haltons should be a list with elements prime and drop,

seed the seed for the pseudo-random draws. This is only relevant if \( \text{haltons} = \text{NA} \),

correlation only relevant if \( \text{ranp} \) is not NULL. If TRUE, the correlation between random parameters is taken into account,

panel if TRUE a panel data model is estimated,

index a string indicating the ‘id’ for individuals in the data. This argument is not required if data is a \( \text{pdata.frame} \) object,

mvar only valid if \( \text{ranp} \) is not NULL. This is a named list, where the names correspond to the variables with random parameters, and the values correspond to the variables that enter in the mean of each random parameters,

print.init if TRUE, the initial values for the optimization procedure are printed,

init.ran initial values for standard deviation of random parameters. Default is 0.1,

gradient if FALSE, numerical gradients are used for the optimization procedure of models with random parameters,

... further arguments passed to \texttt{maxLik},

x, object and object of class \texttt{Rchoice},

new an updated formula for the update method,

digits number of digits,

width width,

Details

The models are estimated using the \texttt{maxLik} function from \texttt{maxLik} package.

If \( \text{ranp} \) is not NULL, the random parameter model is estimated. A random parameter model or random coefficient models permits regression parameter to vary across individuals according to some distribution. A fully parametric random parameter model specifies the latent variable \( y^* \) conditional on regressors \( x \) and given parameters \( \beta_i \) to have conditional density \( f(y|x, \beta_i) \) where \( \beta_i \) are iid with density \( g(\beta_i|\theta_i) \). The density is assumed a priori by the user by the argument \( \text{ranp} \).

If the parameters are assumed to be normally distributed \( \beta_i \sim \mathcal{N}(\beta, \Sigma) \), then the random parameter are constructed as:

\[
\beta_{ir} = \beta + L \omega_{ir}
\]

where \( LL' = \Sigma \) and \( \omega_{ir} \) is the \( r \)-th draw from standard normal distribution for individual \( i \).

Once the model is specified by the argument \texttt{family}, the model is estimated using Simulated Maximum Likelihood (SML). The probabilities, given by \( f(y|x, \beta_i) \), are simulated using \( R \) pseudo-draws if \( \text{halton} = \text{NULL} \) or \( R \) halton draws if \( \text{halton} = \text{NA} \). The user can also specified the primes and the number of dropped elements for the halton draws. For example, if the model consists of two random parameters, the user can specify \( \text{haltons} = \text{list}("\text{prime}" = \text{c}(2, 3), "\text{drop}" = \text{c}(11, 11)) \).
A random parameter hierarchical model can be estimated by including heterogeneity in the mean of the random parameters:

\[ \beta_{ir} = \beta + \pi_s + \mu_{ir} \]

**Rchoice** manages the variables in the hierarchical model by the formula object: all the hierarchical variables \((s_i)\) are included after the \(|\) symbol. The argument \texttt{mvar} indicate which variables enter in each random parameter. See examples below.

### Value

An object of class “Rchoice”, a list elements:

- **coefficients**: the named vector of coefficients,
- **family**: type of model,
- **link**: distribution of the errors,
- **logLik**: a set of values of the maximum likelihood procedure,
- **mf**: the model framed used,
- **formula**: the formula (a Formula object),
- **time**: proc.time() minus the start time,
- **freq**: frequency of dependent variable,
- **draws**: type of draws used,
- **R.model**: TRUE if a random parameter model is fitted,
- **R**: number of draws used,
- **bi**: an array of dimension \(N \times R \times K\) with the individual parameters,
- **Qir**: matrix of dimension \(N \times R\) representing \(P_{ir}/\sum_r P_{ir}\),
- **ranp**: vector indicating the variables with random parameters and their distribution,
- **probabilities**: the fitted probabilities for each individuals,
- **residuals**: the residuals,
- **call**: the matched call.

### Author(s)

Mauricio Sarrias <msarrias86@gmail.com>

### References


### See Also

- plot.Rchoice
- effect.Rchoice
Examples

```r
## Probit model
data("Workmroz")
probit <- Rchoice(lfp ~ k5 + k618 + age + wc + hc + lwg + inc,
                 data = Workmroz, family = binomial('probit'))
summary(probit)

## Poisson model
data("Articles")
poisson <- Rchoice(art ~ fem + mar + kid5 + phd + ment, data = Articles, family = poisson)
summary(poisson)

## Ordered probit model
data("Health")
oprobit <- Rchoice(newhsat ~ age + educ + hhinc + married + hhkids,
                   data = Health, family = ordinal('probit'), subset = year == 1988)
summary(oprobit)

## Poisson Model with Random Parameters
poisson.ran <- Rchoice(art ~ fem + mar + kid5 + phd + ment,
                        data = Articles, family = poisson,
                        ranp = c(kid5 = "n", phd = "n", ment = "n"))
summary(poisson.ran)

## Poisson Model with Correlated Random Parameters
poissonc.ran <- Rchoice(art ~ fem + mar + kid5 + phd + ment,
                         data = Articles,
                         ranp = c(kid5 = "n", phd = "n", ment = "n"),
                         family = poisson,
                         correlation = TRUE,
                         R = 20)
summary(poissonc.ran)

## Hierarchical Poisson Model
poissonH.ran <- Rchoice(art ~ fem + mar + kid5 + phd + ment | fem + phd,
                        data = Articles,
                        ranp = c(kid5 = "n", phd = "n", ment = "n"),
                        mvar = list(phd = c("fem"), ment = c("fem", "phd")),
                        family = poisson,
                        R = 10)
summary(poissonH.ran)

## Ordered Probit Model with Random Effects and Random Parameters
Health$linc <- log(Health$hhinc)
oprobit.ran <- Rchoice(newhsat ~ age + educ + married + hhkids + linc,
                       data = Health[1:2000, ],
                       family = ordinal('probit'),
                       ranp = c(constant = "n", hhkids = "n", linc = "n"),
                       panel = TRUE,
                       index = "id",
                       R = 10,
```
**Model formula for Rchoice models**

**Description**

Two kinds of variables are used in models with individual heterogeneity: the typical variables that enter in the latent process and those variables that enter in the random parameter (Hierarchical Model). `rFormula` deal with this type of models using suitable methods to extract the elements of the model.

**Usage**

```r
rFormula(object)

is.rFormula(object)
```

### S3 methods for class \texttt{rFormula}

```r
model.frame(formula, data, ..., lhs = NULL, rhs = NULL)

model.matrix(object, data, rhs = NULL, ...)
```

**Arguments**

- `object`: a formula form the `rFormula` function, for the `model.matrix` method, a `rFormula` object.
- `formula`: a `rFormula` object.
- `data`: a `data.frame`.
- `...`: further arguments.
- `lhs`: see `Formula`.
- `rhs`: see `Formula`.

```r
print.init = TRUE)
summary(oprobit.ran)
```
The `vcov` method for `Rchoice` objects extracts the covariance matrix of the coefficients or the random parameters. It also allows to get the standard errors for the variance-covariance matrix of the random parameters.

### Usage

```r
## S3 method for class 'Rchoice'
vcov(
  object,
  what = c("coefficient", "ranp"),
  type = c("cov", "cor", "sd"),
  se = FALSE,
  digits = max(3, getOption("digits") - 2),
  ...
)

cov.Rchoice(x)
cor.Rchoice(x)
se.cov.Rchoice(x, sd = FALSE, digits = max(3, getOption("digits") - 2))
```

### Arguments

- **object**: a fitted model of class `Rchoice`.
- **what**: indicates which covariance matrix has to be extracted. The default is `coefficient`. In this case the `vcov` behaves as usual. If `what = "ranp"` the covariance matrix of the random parameters is returned as default.
- **type**: if the model is estimated with random parameters, then this argument indicates what matrix should be returned. If `type = "cov"`, then the covariance matrix of the random parameters is returned; if `type = "cor"` then the correlation matrix of the random parameters is returned; if `type = "sd"` then the standard deviation of the random parameters is returned.
- **se**: if `TRUE` and `type = "cov"` then the standard error of the covariance matrix of the random parameters is returned; if `TRUE` and `type = "sd"` the standard error of the standard deviation of the random parameter is returned. This argument is valid only if the model is estimated using correlated random parameters.
- **digits**: number of digits.
- **x**: a fitted model of class `Rchoice`.
- **sd**: if `TRUE`, then the standard deviation of the random parameters are returned.
Details

This new interface replaces the cor.Rchoice, cov.Rchoice and se.cov.Rchoice functions which are deprecated.

See Also

Rchoice for the estimation of discrete choice models with random parameters.

Description

Data extracted by Mroz(1987) from the 1976 Panel Study of Income Dynamics. The sample consists of 753 white, married women between the ages of 30 and 60.

Usage

data(Workmroz)

Format

A data frame with 753 observations on the following 9 variables:

lfp 1 if wife is in the paid labor force; else 0,
k5 Number of children ages 5 and younger,
k618 Number of children ages 6 to 18,
age Wife’s age in years,
wc 1 if wife attended college; else 0,
hc 1 if husband attended college; else 0,
lwg Log of wife’s estimated wage rate,
inc Family income excluding wife’s wage,
linc Log of Family income excluding wife’s wage,

Source


Examples

data(Workmroz)
Index

* datasets
  Articles, 2
  Attitudes, 3
  Health, 13
  Workmroz, 29

* models
  hetprob, 15
  ivpml, 17
  Rchoice, 22

  Articles, 2
  Attitudes, 3

  bread, 4
  bread.hetprob (hetprob), 15
  bread.ivpml (ivpml), 17
  bread.Rchoice, 4

  coef.hetprob (hetprob), 15
  coef.ivpml (ivpml), 17
  coef.Rchoice (Rchoice), 22
  cor.Rchoice (vcov.Rchoice), 28
  cov.Rchoice (vcov.Rchoice), 28

  df.residual.hetprob (hetprob), 15
  df.residual.ivpml (ivpml), 17
  df.residual.Rchoice (Rchoice), 22

  effect, 5
  effect.hetprob, 6
  effect.ivpml, 7
  effect.Rchoice (Rchoice), 22
  estfun, 10
  estfun.hetprob (hetprob), 15
  estfun.ivpml (ivpml), 17
  estfun.Rchoice, 10

  fitted.Rchoice (Rchoice), 22
  Formula, 27

  getSummary.effect.hetprob, 11
  getSummary.effect.ivpml, 11
  getSummary.hetprob, 12
  getSummary.ivpml, 12
  getSummary.Rchoice, 13

  Health, 13
  hetprob, 15

  is.rFormula (rFormula), 27
  ivpml, 17

  logLik.hetprob (hetprob), 15
  logLik.ivpml (ivpml), 17
  logLik.Rchoice (Rchoice), 22

  maxLik, 16, 19, 24
  model.frame.rFormula (rFormula), 27
  model.matrix.hetprob (hetprob), 15
  model.matrix.ivpml (ivpml), 17
  model.matrix.Rchoice (Rchoice), 22
  model.matrix.rFormula (rFormula), 27

  ordinal (Rchoice), 22

  plot.Rchoice, 20, 25
  predict.hetprob (hetprob), 15
  predict.ivpml (ivpml), 17
  print.effect.hetprob (effect.hetprob), 6
  print.effect.ivpml (effect.ivpml), 7
  print.hetprob (hetprob), 15
  print.ivpml (ivpml), 17
  print.Rchoice (Rchoice), 22
  print.summary.eff

  plot.Rchoice (Rchoice), 22
  print.summary.hetprob (effect.hetprob), 6
  print.summary.ivpml (effect.ivpml), 7
  print.summary.Rchoice (Rchoice), 22

  Rchoice, 9, 21, 22, 29
residuals.Rchoice (Rchoice), 22  
rFormula, 23, 27

se.cov.Rchoice (vcov.Rchoice), 28  
summary.effect.hetprob  
   (effect.hetprob), 6  
summary.effect.ivpml (effect.ivpml), 7  
summary.hetprob (hetprob), 15  
summary.ivpml (ivpml), 17  
summary.Rchoice (Rchoice), 22

terms.hetprob (hetprob), 15  
terms.ivpml (ivpml), 17  
terms.Rchoice (Rchoice), 22

update.Rchoice (Rchoice), 22

vcov.hetprob (hetprob), 15  
vcov.ivpml (ivpml), 17  
vcov.Rchoice, 28

Workmroz, 29