

Package ‘mlogit’

July 22, 2019

Version 1.0-1

Date 2019-07-21

Title Multinomial Logit Models

Depends R (>= 2.10), Formula, zoo, lmtest

Imports statmod, MASS, Rdpack

Suggests knitr, car, nnet, lattice, AER, ggplot2, texreg, rmarkdown

Description Maximum Likelihood estimation of random utility discrete choice models, as described in Kenneth Train (2009) Discrete Choice Methods with Simulations <doi:10.1017/CBO9780511805271>.

VignetteBuilder knitr

Encoding UTF-8

License GPL (>= 2)

URL <https://cran.r-project.org/package=mlogit>,
<https://r-forge.r-project.org/projects/mlogit/>

RoxygenNote 6.1.1

RdMacros Rdpack

NeedsCompilation no

Author Yves Croissant [aut, cre]

Maintainer Yves Croissant <yves.croissant@univ-reunion.fr>

Repository CRAN

Date/Publication 2019-07-22 11:00:02 UTC

R topics documented:

mlogit-package	2
Car	2
Catsup	3
cor.mlogit	4
Cracker	4

distribution	5
effects.mlogit	7
Electricity	8
Fishing	9
Game	10
HC	11
Heating	12
hmftest	12
JapaneseFDI	14
logsum	15
mFormula	16
miscmethods.mlogit	18
mlogit	19
mlogit.data	24
mlogit.optim	27
Mode	29
ModeCanada	29
NOx	30
plot.mlogit	31
RiskyTransport	32
rpar	33
scoretest	34
Train	36
vcov.mlogit	37

mlogit-package

*mlogit package: estimation of random utility discrete choice models
by maximum likelihood*

Description

mlogit provides a model description interface (enhanced formula-data), a very versatile estimation function and a testing infrastructure to deal with random utility models.

Details

For a gentle and comprehensive introduction to the package, see the package's vignettes.

Car

Stated Preferences for Car Choice

Description

a sample of 4654 individuals

Format

A dataframe containing :

- choice: choice of a vehicle among 6 propositions,
- college: college education?,
- hsg2: size of household greater than 2?
- com15: commute lower than 5 miles a day?,
- typez: body type, one of regcar (regular car), sportuv (sport utility vehicle), sportcar, stwagon (station wagon), truck, van, for each proposition z from 1 to 6,
- fuelz: fuel for proposition z, one of gasoline, methanol, cng (compressed natural gas), electric.,
- pricez: price of vehicle divided by the logarithm of income,
- rangez: hundreds of miles vehicle can travel between refuelings/rechargings,
- accz: acceleration, tens of seconds required to reach 30 mph from stop,
- speedz: highest attainable speed in hundreds of mph,
- pollutionz: tailpipe emissions as fraction of those for new gas vehicle,
- sizez: 0 for a mini, 1 for a subcompact, 2 for a compact and 3 for a mid-size or large vehicle,
- spacez: fraction of luggage space in comparable new gas vehicle,
- costz: cost per mile of travel (tens of cents) : home recharging for electric vehicle, station refueling otherwise,
- stationz: fraction of stations that can refuel/recharge vehicle.

Source

Journal of Applied Econometrics data archive¹.

References

McFadden D, Train K (2000). "Mixed MNL Models for Discrete Response." *Journal of Applied Econometrics*, **15**(5), 447–470. ISSN 08837252, 10991255.

¹<http://jae.wiley.com/jae/>

 Catsup

Choice of Brand for Catsup

Description

a sample of 2798 individuals

Format

A dataframe containing :

- id: individuals identifiers,
- choice: one of heinz41, heinz32, heinz28, hunts32,
- disp.z: is there a display for brand z ?
- feat.z: is there a newspaper feature advertisement for brand z?
- price.z: price of brand z.

Source

Journal of Business Economics and Statistics web site².

References

Jain DC, Vilcassim NJ, Chintagunta PK (1994). "A Random-Coefficients Logit Brand-Choice Model Applied to Panel Data." *Journal of Business & Economic Statistics*, **12**(3), 317-328.

 cor.mlogit

Correlation structure of the random parameters

Description

Functions that extract the correlation structure of a mlogit object

Usage

```
cor.mlogit(x)
```

```
cov.mlogit(x)
```

Arguments

x an mlogit object with random parameters and correlation = TRUE.

²<https://www.amstat.org>

Details

These functions are deprecated, use `vcov` instead.

Value

A numerical matrix which returns either the correlation or the covariance matrix of the random parameters.

Author(s)

Yves Croissant

Cracker

Choice of Brand for Crakers

Description

a sample of 3292 individuals cross-section

Format

A dataframe containing :

- `id`: individuals identifiers,
- `choice`: one of sunshine, keebler, nabisco, private,
- `disp.z`: is there a display for brand z?
- `feat.z`: is there a newspaper feature advertisement for brand z?
- `price.z`: price of brand z.

Source

Journal of Business Economics and Statistics web site³.

References

Jain DC, Vilcassim NJ, Chintagunta PK (1994). "A Random-Coefficients Logit Brand-Choice Model Applied to Panel Data." *Journal of Business & Economic Statistics*, **12**(3), 317-328.

Paap R, Franses PH (2000). "A dynamic multinomial probit model for brand choice with different long-run and short-run effects of marketing-mix variables." *Journal of Applied Econometrics*, **15**(6), 717-744.

³<https://www.amstat.org>

distribution	<i>Functions used to describe the characteristics of estimated random parameters</i>
--------------	--

Description

Functions used to describe the characteristics of estimated random parameters

Usage

```
stdev(x, ...)  
  
rg(x, ...)  
  
med(x, ...)  
  
## S3 method for class 'rpar'  
mean(x, norm = NULL, ...)  
  
## S3 method for class 'rpar'  
med(x, norm = NULL, ...)  
  
## S3 method for class 'rpar'  
stdev(x, norm = NULL, ...)  
  
## S3 method for class 'rpar'  
rg(x, norm = NULL, ...)  
  
## S3 method for class 'mlogit'  
mean(x, par = NULL, norm = NULL, ...)  
  
## S3 method for class 'mlogit'  
med(x, par = NULL, norm = NULL, ...)  
  
## S3 method for class 'mlogit'  
stdev(x, par = NULL, norm = NULL, ...)  
  
## S3 method for class 'mlogit'  
rg(x, par = NULL, norm = NULL, ...)  
  
qrpar(x, ...)  
  
prpar(x, ...)  
  
drpar(x, ...)  
  
## S3 method for class 'rpar'
```

```

qrpar(x, norm = NULL, ...)

## S3 method for class 'rpar'
prpar(x, norm = NULL, ...)

## S3 method for class 'rpar'
drpar(x, norm = NULL, ...)

## S3 method for class 'mlogit'
qrpar(x, par = 1, y = NULL, norm = NULL, ...)

## S3 method for class 'mlogit'
prpar(x, par = 1, y = NULL, norm = NULL, ...)

## S3 method for class 'mlogit'
drpar(x, par = 1, y = NULL, norm = NULL, ...)

```

Arguments

<code>x</code>	a <code>mlogit</code> or a <code>rpar</code> object,
<code>...</code>	further arguments.
<code>norm</code>	the variable used for normalization if any : for the <code>mlogit</code> method, this should be the name of the parameter, for the <code>rpar</code> method the absolute value of the parameter,
<code>par</code>	the required parameter(s) for the <code>mlogit</code> methods (either the name or the position of the parameter(s)). If <code>NULL</code> , all the random parameters are used.
<code>y</code>	values for which the function has to be evaluated,

Details

`rpar` objects contain all the relevant information about the distribution of random parameters. These functions enables to obtain easily descriptive statistics, density, probability and quantiles of the distribution.

`mean`, `med`, `stdev` and `rg` compute respectively the mean, the median, the standard deviation and the range of the random parameter. `qrpar`, `prpar`, `drpar` return functions that compute the quantiles, the probability and the density of the random parameters (note that `sd` and `range` are not generic function in R and that `median` is, but without `...`).

Value

a numeric vector for `qrpar`, `drpar` and `prpar`, a numeric vector for `mean`, `stdev` and `med` and a numeric matrix for `rg`.

Author(s)

Yves Croissant

See Also

`mlogit()` for the estimation of random parameters logit models and `rpar()` for the description of `rpar` objects.

`effects.mlogit` *Marginal effects of the covariates*

Description

The `effects` method for `mlogit` objects computes the marginal effects of the selected covariate on the probabilities of choosing the alternatives

Usage

```
## S3 method for class 'mlogit'
effects(object, covariate = NULL, type = c("aa", "ar",
    "rr", "ra"), data = NULL, ...)
```

Arguments

<code>object</code>	a <code>mlogit</code> object,
<code>covariate</code>	the name of the covariate for which the effect should be computed,
<code>type</code>	the effect is a ratio of two marginal variations of the probability and of the covariate ; these variations can be absolute "a" or relative "r". This argument is a string that contains two letters, the first refers to the probability, the second to the covariate,
<code>data</code>	a <code>data.frame</code> containing the values for which the effects should be calculated. The number of lines of this <code>data.frame</code> should be equal to the number of alternatives,
<code>...</code>	further arguments.

Value

If the covariate is alternative specific, a $J \times J$ matrix is returned, J being the number of alternatives. Each line contains the marginal effects of the covariate of one alternative on the probability to choose any alternative. If the covariate is individual specific, a vector of length J is returned.

Author(s)

Yves Croissant

See Also

`mlogit()` for the estimation of multinomial logit models.

Examples

```

data("Fishing", package = "mlogit")
library("zoo")
Fish <- mlogit.data(Fishing, varying = c(2:9), shape = "wide", choice = "mode")
m <- mlogit(mode ~ price | income | catch, data = Fish)
# compute a data.frame containing the mean value of the covariates in
# the sample
z <- with(Fish, data.frame(price = tapply(price, index(m)$alt, mean),
                           catch = tapply(catch, index(m)$alt, mean),
                           income = mean(income)))
# compute the marginal effects (the second one is an elasticity
effects(m, covariate = "income", data = z)
effects(m, covariate = "price", type = "rr", data = z)
effects(m, covariate = "catch", type = "ar", data = z)

```

Electricity

Stated preference data for the choice of electricity suppliers

Description

A sample of 2308 households in the United States

Format

A dataframe containing :

- choice: the choice of the individual, one of 1, 2, 3, 4,
- id: the individual index,
- pfi: fixed price at a stated cents per kWh, with the price varying over suppliers and experiments, for scenario $i=(1, 2, 3, 4)$,
- cli: the length of contract that the supplier offered, in years (such as 1 year or 5 years.) During this contract period, the supplier guaranteed the prices and the buyer would have to pay a penalty if he/she switched to another supplier. The supplier could offer no contract in which case either side could stop the agreement at any time. This is recorded as a contract length of 0,
- loci: is the supplier a local company,
- wki: is the supplier a well-known company,
- todi: a time-of-day rate under which the price is 11 cents per kWh from 8am to 8pm and 5 cents per kWh from 8pm to 8am. These TOD prices did not vary over suppliers or experiments: whenever the supplier was said to offer TOD, the prices were stated as above.
- seasi: a seasonal rate under which the price is 10 cents per kWh in the summer, 8 cents per kWh in the winter, and 6 cents per kWh in the spring and fall. Like TOD rates, these prices did not vary. Note that the price is for the electricity only, not transmission and distribution, which is supplied by the local regulated utility.

Source

Kenneth Train's home page⁴.

References

Huber J, Train K (2000). "On the Similarity of Classical and Bayesian Estimates of Individual Mean Partworths." *Marketing Letters*, **12**, 259–269.

Revelt D, Train K (2001). "Customer-Specific Taste Parameters and Mixed Logit: Households' Choice of Electricity Supplier." *Econometrics 0012001*, University Library of Munich, Germany. <https://ideas.repec.org/p/wpa/wuwpem/0012001.html>.

Fishing

Choice of Fishing Mode

Description

A sample of 1182 individuals in the United-States for the choice of 4 alternative fishing modes.

Format

A dataframe containing :

- mode: recreation mode choice, one of : beach, pier, boat and charter,
- price.beach: price for beach mode
- price.pier: price for pier mode,
- price.boat: price for private boat mode,
- price.charter: price for charter boat mode,
- catch.beach: catch rate for beach mode,
- catch.pier: catch rate for pier mode,
- catch.boat: catch rate for private boat mode,
- catch.charter: catch rate for charter boat mode,
- income: monthly income,

Source

Cameron A, Trivedi P (2005). *Microeconometrics*. Cambridge University Press. <https://EconPapers.repec.org/RePEc:cup:cbooks:9780521848053>.

References

Herriges JA, Kling CL (1999). "Nonlinear Income Effects in Random Utility Models." *The Review of Economics and Statistics*, **81**(1), 62-72. doi: 10.1162/003465399767923827⁵, <https://doi.org/10.1162/003465399767923827>, <https://doi.org/10.1162/003465399767923827>.

⁴<http://elsa.berkeley.edu/~train/>

⁵<https://doi.org/10.1162/003465399767923827>

Game

Ranked data for gaming platforms

Description

A sample of 91 Dutch individuals

Format

A dataframe containing :

- `ch.Platform`: where `platform` is one of Xbox, PlayStation, PSPortable, GameCube, GameBoy and PC. This variables contain the ranking of the platforms from 1 to 6,
- `own.Platform`: these 6 variables are dummies which indicate whether the given platform is already owned by the respondent,
- `age`: the age of the respondent,
- `hours`: hours per week spent on gaming.,

Details

The data are also provided in long format (use in this case `data(Game2)`). In this case, the alternative and the choice situation are respectively indicated in the `platform` and `chid` variables.

Source

Journal of Applied Econometrics data archive⁶.

References

Fok D, Paap R, Van Dijk B (2012). "A Rank-Ordered Logit Model With Unobserved Heterogeneity In Ranking Capabilities." *Journal of Applied Econometrics*, **27**(5), 831-846. doi: 10.1002/jae.1223⁷, <https://onlinelibrary.wiley.com/doi/pdf/10.1002/jae.1223>, <https://onlinelibrary.wiley.com/doi/abs/10.1002/jae.1223>.

⁶<http://jae.wiley.com/jae/>

⁷<https://doi.org/10.1002/jae.1223>

HC	<i>Heating and Cooling System Choice in Newly Built Houses in California</i>
----	--

Description

A sample of 250 Californian households

Format

A dataframe containing :

- depvar: heating system, one of gcc (gas central heat with cooling), ecc (electric central resistance heat with cooling), erc (electric room resistance heat with cooling), hpc (electric heat pump which provides cooling also), gc (gas central heat without cooling), ec (electric central resistance heat without cooling), er (electric room resistance heat without cooling),
- ich.z: installation cost of the heating portion of the system,
- icca: installation cost for cooling,
- och.z: operating cost for the heating portion of the system,
- occa: operating cost for cooling,
- income: annual income of the household.

Source

Kenneth Train's home page⁸.

Heating	<i>Heating System Choice in California Houses</i>
---------	---

Description

A sample of 900 Californian households#'

Format

A dataframe containing:

- idcase: id,
- depvar: heating system, one of gc (gas central), gr (gas room), ec (electric central), er (electric room), hp (heat pump),
- ic.z: installation cost for heating system z (defined for the 5 heating systems),
- oc.z: annual operating cost for heating system z (defined for the 5 heating systems),

⁸<http://elsa.berkeley.edu/~train/>

- pb.z: ratio $oc.z/ic.z$,
- income: annual income of the household,
- agehed: age of the household head
- rooms: numbers of rooms in the house,

Source

Kenneth Train's home page⁹.

hmftest

Hausman-McFadden Test

Description

Test the IIA hypothesis (independence of irrelevant alternatives) for a multinomial logit model.

Usage

```
hmftest(x, ...)

## S3 method for class 'formula'
hmftest(x, alt.subset, ...)

## S3 method for class 'mlogit'
hmftest(x, z, ...)
```

Arguments

x	an object of class <code>mlogit</code> or a formula,
...	further arguments passed to <code>mlogit</code> for the <code>formula</code> method.
alt.subset	a subset of alternatives,
z	an object of class <code>mlogit</code> or a subset of alternatives for the <code>mlogit</code> method. This should be the same model as <code>x</code> estimated on a subset of alternatives,

Details

This is an implementation of the Hausman's consistency test for multinomial logit models. If the independence of irrelevant alternatives applies, the probability ratio of every two alternatives depends only on the characteristics of these alternatives. Consequently, the results obtained on the estimation with all the alternatives or only on a subset of them are consistent, but more efficient in the first case. On the contrary, only the results obtained from the estimation on a relevant subset are consistent. To compute this test, one needs a model estimated with all the alternatives and one model estimated on a subset of alternatives. This can be done by providing two objects of class `mlogit`, one object of class `mlogit` and a character vector indicating the subset of alternatives, or a formula and a subset of alternatives.

⁹<http://elsa.berkeley.edu/~train/>

Value

an object of class "htest".

Author(s)

Yves Croissant

References

Hausman, J.A. and D. McFadden (1984), A Specification Test for the Multinomial Logit Model, *Econometrica*, **52**, pp.1219–1240.

Examples

```
## from Greene's Econometric Analysis p. 731

data("TravelMode", package="AER")
TravelMode <- mlogit.data(TravelMode, choice="choice", shape="long",
                        alt.var="mode", chid.var="individual")

## Create a variable of income only for the air mode

TravelMode$avinc <- with(TravelMode, (mode=='air')*income)

## Estimate the model on all alternatives, with car as the base level
## like in Greene's book.

#x <- mlogit(choice~wait+gcost+avinc, TravelMode, refllevel="car")
x <- mlogit(choice~wait+gcost+avinc, TravelMode)

## Estimate the same model for ground modes only (the variable avinc
## must be dropped because it is 0 for every observation

g <- mlogit(choice~wait+gcost, TravelMode, refllevel="car",
           alt.subset=c("car", "bus", "train"))

## Compute the test

hmftest(x, g)
```

Description

A sample of 452 Japanese production units in Europe #'

Format

A dataframe containing :

- firm: the investment id,
- country: the country,
- region: the region (nuts1 nomenclature),
- choice: a dummy indicating the chosen region ,
- choice.c: the chosen country,
- wage: wage rate in the region,
- unemp: unemployment rate in the region,
- elig: is the country eligible to european subsidies,
- area: the area of the region,
- scrate: social charge rate (country level),
- ctaxrate: corporate tax rate (country level),
- gdp: regional gdp,
- harris: harris' market potential,
- krugman: krugman's market potential,
- domind: domestic industry count,
- japind: japan industry count,
- network: network count.

Source

kindly provided by Thierry Mayer

References

Head K, Mayer T (2004). "Market Potential and the Location of Japanese Investment in the European Union." *The Review of Economics and Statistics*, **86**(4), 959-972. doi: 10.1162/0034653043125257¹⁰, <https://doi.org/10.1162/0034653043125257>, <https://doi.org/10.1162/0034653043125257>.

¹⁰<https://doi.org/10.1162/0034653043125257>

logsum

*Compute the log-sum or inclusive value/utility***Description**

The `logsum` function computes the inclusive value, or inclusive utility, which is used to compute the surplus and to estimate the two steps nested logit model.

Usage

```
logsum(coef, X = NULL, formula = NULL, data = NULL, type = NULL,
       output = c("chid", "obs"))
```

Arguments

<code>coef</code>	a numerical vector or a <code>mlogit</code> object, from which the <code>coef</code> vector is extracted,
<code>X</code>	a matrix or a <code>mlogit</code> object from which the <code>model.matrix</code> is extracted,
<code>formula</code>	a formula or a <code>mlogit</code> object from which the <code>formula</code> is extracted,
<code>data</code>	a <code>data.frame</code> or a <code>mlogit</code> object from which the <code>model.frame</code> is extracted,
<code>type</code>	either "group" or "global" : if a <code>group</code> argument has been provided in the <code>mlogit.data</code> , the inclusive values are by default computed for every group, otherwise, a unique global inclusive value is computed for each choice situation,
<code>output</code>	the shape of the results: if "chid", the results is a vector (if <code>type = "global"</code>) or a matrix (if <code>type = "region"</code>) with row number equal to the number of choice situation, if "obs" a vector of length equal to the number of lines of the data in long format is returned.

Details

The inclusive value, or inclusive utility, or log-sum is the log of the denominator of the probabilities of the multinomial logit model. If a "group" variable is provided in the "mlogit.data" function, the denominator can either be the one of the multinomial model or those of the lower model of the nested logit model.

If only one argument (`coef`) is provided, it should a `mlogit` object and in this case, the coefficients and the `model.matrix` are extracted from this model.

In order to provide a different `model.matrix`, further arguments could be used. `X` is a matrix or a `mlogit` from which the `model.matrix` is extracted. The `formula-data` interface can also be used to construct the relevant `model.matrix`.

Value

either a vector or a matrix.

Author(s)

Yves Croissant

See Also

mlogit() for the estimation of a multinomial logit model.

mFormula*Model formula for logit models*

Description

Two kinds of variables are used in logit models: alternative specific and individual specific variables. mFormula provides a relevant class to deal with this specificity and suitable methods to extract the elements of the model.

Usage

```
mFormula(object)

## S3 method for class 'formula'
mFormula(object)

is.mFormula(object)

## S3 method for class 'mFormula'
model.frame(formula, data, ..., lhs = NULL,
             rhs = NULL, alt.subset = NULL, refllevel = NULL)

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

Arguments

object	for the mFormula function, a formula, for the update and model.matrix methods, a mFormula object,
formula	a mFormula object,
data	a data.frame,
...	further arguments.
lhs	see Formula,
rhs	see Formula,
alt.subset	a vector of subset of alternatives one want to select,
reflevel	the alternative selected to be the reference alternative,

Details

Let J being the number of alternatives. The formula may include alternative-specific and individual specific variables. For the latter, $J - 1$ coefficients are estimated for each variable. For the former, only one (generic) coefficient or J different coefficient may be estimated.

A `mFormula` is a formula for which the right hand side may contain three parts: the first one contains the alternative specific variables with generic coefficient, *i.e.* a unique coefficient for all the alternatives ; the second one contains the individual specific variables for which one coefficient is estimated for all the alternatives except one of them ; the third one contains the alternative specific variables with alternative specific coefficients. The different parts are separated by a `|` sign. If a standard formula is written, it is assumed that there are only alternative specific variables with generic coefficients.

The intercept is necessarily alternative specific (a generic intercept is not identified because only utility differences are relevant). Therefore, it deals with the second part of the formula. As it is usual in R , the default behaviour is to include an intercept. A model without an intercept may be specified by including `+ 0` or `-1` in the second right-hand side part of the formula. `+ 0` or `-1` in the first and in the third part of the formula are simply ignored.

Specific methods are provided to build correctly the model matrix and to update the formula. The `mFormula` function is not intended to be use directly. While using the `mlogit()` function, the first argument is automatically coerced to a `mFormula` object.

Value

an object of class `mFormula`.

Author(s)

Yves Croissant

Examples

```
data("Fishing", package = "mlogit")
Fish <- mlogit.data(Fishing, varying = c(2:9), shape = "wide", choice =
"mode")

# a formula with to alternative specific variables (price and
# catch) and an intercept
f1 <- mFormula(mode ~ price + catch)
head(model.matrix(f1, Fish), 2)

# same, with an individual specific variable (income)
f2 <- mFormula(mode ~ price + catch | income)
head(model.matrix(f2, Fish), 2)

# same, without an intercept
f3 <- mFormula(mode ~ price + catch | income + 0)
head(model.matrix(f3, Fish), 2)

# same as f2, but now, coefficients of catch are alternative
```

```
# specific
f4 <- mFormula(mode ~ price | income | catch)
head(model.matrix(f4, Fish), 2)
```

miscmethods.mlogit *Methods for mlogit objects*

Description

Miscellaneous methods for mlogit objects.

Usage

```
## S3 method for class 'mlogit'
residuals(object, outcome = TRUE, ...)

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

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

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

model.response.mlogit(object, ...)

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

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

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

## S3 method for class 'mlogit'
summary(object, ..., type = c("chol", "cov", "cor"))

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

## S3 method for class 'mlogit'
index(x, ...)
```

```
## S3 method for class 'mlogit'
predict(object, newdata = NULL, returnData = FALSE,
  ...)

## S3 method for class 'mlogit'
fitted(object, type = c("outcome", "probabilities",
  "linpred", "parameters"), outcome = NULL, ...)

## S3 method for class 'mlogit'
coef(object, subset = c("all", "iv", "sig", "sd", "sp",
  "chol"), fixed = FALSE, ...)

## S3 method for class 'summary.mlogit'
coef(object, ...)
```

Arguments

<code>outcome</code>	a boolean which indicates, for the <code>fitted</code> and the <code>residuals</code> methods whether a matrix (for each choice, one value for each alternative) or a vector (for each choice, only a value for the alternative chosen) should be returned,
<code>...</code>	further arguments.
<code>x, object</code>	an object of class <code>mlogit</code>
<code>new</code>	an updated formula for the <code>update</code> method,
<code>digits</code>	the number of digits,
<code>width</code>	the width of the printing,
<code>type</code>	one of <code>outcome</code> (probability of the chosen alternative), <code>probabilities</code> (probabilities for all the alternatives), <code>parameters</code> for individual-level random parameters for the <code>fitted</code> method, how the correlated random parameters should be displayed : <code>"chol"</code> for the estimated parameters (the elements of the Cholesky decomposition matrix), <code>"cov"</code> for the covariance matrix and <code>"cor"</code> for the correlation matrix and the standard deviations,
<code>newdata</code>	a <code>data.frame</code> for the <code>predict</code> method,
<code>returnData</code>	for the <code>predict</code> method, if <code>TRUE</code> , the data is returned as an attribute,
<code>subset</code>	an optional vector of coefficients to extract for the <code>coef</code> method,
<code>fixed</code>	if <code>FALSE</code> (the default), constant coefficients are not returned,

mlogit

Multinomial logit model

Description

Estimation by maximum likelihood of the multinomial logit model, with alternative-specific and/or individual specific variables.

Usage

```
mlogit(formula, data, subset, weights, na.action, start = NULL,
        alt.subset = NULL, refllevel = NULL, nests = NULL,
        un.nest.el = FALSE, unscaled = FALSE, heterosc = FALSE,
        rpar = NULL, probit = FALSE, R = 40, correlation = FALSE,
        halton = NULL, random.nb = NULL, panel = FALSE, estimate = TRUE,
        seed = 10, ...)
```

Arguments

<code>formula</code>	a symbolic description of the model to be estimated,
<code>data</code>	the data: an <code>mlogit.data</code> object or an ordinary <code>data.frame</code> ,
<code>subset</code>	an optional vector specifying a subset of observations for <code>mlogit</code> ,
<code>weights</code>	an optional vector of weights,
<code>na.action</code>	a function which indicates what should happen when the data contains NAs,
<code>start</code>	a vector of starting values,
<code>alt.subset</code>	a vector of character strings containing the subset of alternative on which the model should be estimated,
<code>reflevel</code>	the base alternative (the one for which the coefficients of individual-specific variables are normalized to 0),
<code>nests</code>	a named list of characters vectors, each names being a nest, the corresponding vector being the set of alternatives that belong to this nest,
<code>un.nest.el</code>	a boolean, if <code>TRUE</code> , the hypothesis of unique elasticity is imposed for nested logit models,
<code>unscaled</code>	a boolean, if <code>TRUE</code> , the unscaled version of the nested logit model is estimated,
<code>heterosc</code>	a boolean, if <code>TRUE</code> , the heteroscedastic logit model is estimated,
<code>rpar</code>	a named vector whose names are the random parameters and values the distribution: 'n' for normal, 'l' for log-normal, 't' for truncated normal, 'u' for uniform,
<code>probit</code>	if <code>TRUE</code> , a multinomial porbit model is estimated,
<code>R</code>	the number of function evaluation for the gaussian quadrature method used if <code>heterosc = TRUE</code> , the number of draws of pseudo-random numbers if <code>rpar</code> is not <code>NULL</code> ,
<code>correlation</code>	only relevant if <code>rpar</code> is not <code>NULL</code> , if <code>true</code> , the correlation between random parameters is taken into account,
<code>halton</code>	only relevant if <code>rpar</code> is not <code>NULL</code> , if not <code>NULL</code> , halton sequence is used instead of pseudo-random numbers. If <code>halton = NA</code> , some default values are used for the prime of the sequence (actually, the primes are used in order) and for the number of elements dropped. Otherwise, <code>halton</code> should be a list with elements <code>prime</code> (the primes used) and <code>drop</code> (the number of elements dropped).
<code>random.nb</code>	only relevant if <code>rpar</code> is not <code>NULL</code> , a user-supplied matrix of random,
<code>panel</code>	only relevant if <code>rpar</code> is not <code>NULL</code> and if the data are repeated observations of the same unit ; if <code>TRUE</code> , the mixed-logit model is estimated using panel techniques,

<code>estimate</code>	a boolean indicating whether the model should be estimated or not: if not, the <code>model.frame</code> is returned,
<code>seed</code>	the seed to use for random numbers (for mixed logit and probit models),
<code>...</code>	further arguments passed to <code>mlogit.data</code> or <code>mlogit.optim</code> .

Details

For how to use the formula argument, see `mFormula()`.

The data argument may be an ordinary `data.frame`. In this case, some supplementary arguments should be provided and are passed to `mlogit.data()`. Note that it is not necessary to indicate the choice argument as it is deduced from the formula.

The model is estimated using the `mlogit.optim()` function.

The basic multinomial logit model and three important extensions of this model may be estimated.

If `heterosc=TRUE`, the heteroscedastic logit model is estimated. $J-1$ extra coefficients are estimated that represent the scale parameter for $J-1$ alternatives, the scale parameter for the reference alternative being normalized to 1. The probabilities don't have a closed form, they are estimated using a gaussian quadrature method.

If `nests` is not `NULL`, the nested logit model is estimated.

If `rpar` is not `NULL`, the random parameter model is estimated. The probabilities are approximated using simulations with `R` draws and halton sequences are used if `halton` is not `NULL`. Pseudo-random numbers are drawn from a standard normal and the relevant transformations are performed to obtain numbers drawn from a normal, log-normal, censored-normal or uniform distribution. If `correlation=TRUE`, the correlation between the random parameters are taken into account by estimating the components of the cholesky decomposition of the covariance matrix. With `G` random parameters, without correlation `G` standard deviations are estimated, with correlation $G * (G + 1) / 2$ coefficients are estimated.

Value

An object of class "mlogit", a list with elements:

- `coefficients`: the named vector of coefficients,
- `logLik`: the value of the log-likelihood,
- `hessian`: the hessian of the log-likelihood at convergence,
- `gradient`: the gradient of the log-likelihood at convergence,
- `call`: the matched call,
- `est.stat`: some information about the estimation (time used, optimisation method),
- `freq`: the frequency of choice,
- `residuals`: the residuals,
- `fitted.values`: the fitted values,
- `formula`: the formula (a `mFormula` object),
- `expanded.formula`: the formula (a `formula` object),
- `model`: the model frame used,
- `index`: the index of the choice and of the alternatives.

Author(s)

Yves Croissant

References

McFadden D (1973). "Conditional Logit Analysis of Qualitative Choice Behaviour." In Zarembka P (ed.), *Frontiers in Econometrics*, 105-142. Academic Press New York, New York, NY, USA.

McFadden D (1974). "The measurement of urban travel demand." *Journal of Public Economics*, 3(4), 303 - 328. ISSN 0047-2727, <http://www.sciencedirect.com/science/article/pii/0047272774900036>.

Train K (2009). *Discrete Choice Methods with Simulation*. Cambridge University Press. <https://EconPapers.repec.org/RePEc:cup:cbooks:9780521766555>.

See Also

`mlogit.data()` to shape the data. `nnet::multinom()` from package `nnet` performs the estimation of the multinomial logit model with individual specific variables. `mlogit.optim()` details about the optimization function.

Examples

```
## Cameron and Trivedi's Microeconometrics p.493 There are two
## alternative specific variables : price and catch one individual
## specific variable (income) and four fishing mode : beach, pier, boat,
## charter

data("Fishing", package = "mlogit")
Fish <- mlogit.data(Fishing, varying = c(2:9), shape = "wide", choice = "mode")

## a pure "conditional" model
summary(mlogit(mode ~ price + catch, data = Fish))

## a pure "multinomial model"
summary(mlogit(mode ~ 0 | income, data = Fish))

## which can also be estimated using multinom (package nnet)
library("nnet")
summary(multinom(mode ~ income, data = Fishing))

## a "mixed" model
m <- mlogit(mode ~ price+ catch | income, data = Fish)
summary(m)

## same model with charter as the reference level
m <- mlogit(mode ~ price+ catch | income, data = Fish, relevel = "charter")

## same model with a subset of alternatives : charter, pier, beach
m <- mlogit(mode ~ price+ catch | income, data = Fish,
            alt.subset = c("charter", "pier", "beach"))
```

```

## model on unbalanced data i.e. for some observations, some
## alternatives are missing
# a data.frame in wide format with two missing prices
Fishing2 <- Fishing
Fishing2[1, "price.pier"] <- Fishing2[3, "price.beach"] <- NA
mlogit(mode~price+catch|income, Fishing2, shape="wide", choice="mode", varying = 2:9)

# a data.frame in long format with three missing lines
data("TravelMode", package = "AER")
Tr2 <- TravelMode[-c(2, 7, 9),]
mlogit(choice~wait+gcost|income+size, Tr2, shape = "long",
       chid.var = "individual", alt.var="mode", choice = "choice")

## An heteroscedastic logit model
data("TravelMode", package = "AER")
hl <- mlogit(choice ~ wait + travel + vcost, TravelMode,
            shape = "long", chid.var = "individual", alt.var = "mode",
            method = "bfgs", heterosc = TRUE, tol = 10)

## A nested logit model
TravelMode$avincome <- with(TravelMode, income * (mode == "air"))
TravelMode$time <- with(TravelMode, travel + wait)/60
TravelMode$timeair <- with(TravelMode, time * I(mode == "air"))
TravelMode$income <- with(TravelMode, income / 10)
# Hensher and Greene (2002), table 1 p.8-9 model 5
TravelMode$incomeother <- with(TravelMode, ifelse(mode %in% c('air', 'car'), income, 0))
nl <- mlogit(choice~gcost+wait+incomeother, TravelMode,
            shape='long', alt.var='mode',
            nests=list(public=c('train', 'bus'), other=c('car','air')))
# same with a comon nest elasticity (model 1)
nl2 <- update(nl, un.nest.el = TRUE)

## a probit model
## Not run:
pr <- mlogit(choice ~ wait + travel + vcost, TravelMode,
            shape = "long", chid.var = "individual", alt.var = "mode",
            probit = TRUE)

## End(Not run)

## a mixed logit model
## Not run:
rpl <- mlogit(mode ~ price+ catch | income, Fishing, varying = 2:9,
            shape = 'wide', rpar = c(price= 'n', catch = 'n'),
            correlation = TRUE, halton = NA,
            R = 10, tol = 10, print.level = 0)
summary(rpl)
rpar(rpl)
cor.mlogit(rpl)
cov.mlogit(rpl)
rpar(rpl, "catch")
summary(rpar(rpl, "catch"))

```

```
## End(Not run)

# a ranked ordered model
data("Game", package = "mlogit")
g <- mlogit(ch~own|hours, Game, choice='ch', varying = 1:12,
            ranked=TRUE, shape="wide", refllevel="PC")
```

mlogit.data	<i>data.frame for logit model</i>
-------------	-----------------------------------

Description

shape a `data.frame` in a suitable form for the use of the `mlogit` function.

Usage

```
mlogit.data(data, choice = NULL, shape = c("long", "wide"),
            varying = NULL, sep = ".", alt.var = NULL, chid.var = NULL,
            alt.levels = NULL, id.var = NULL, group.var = NULL,
            opposite = NULL, drop.index = FALSE, ranked = FALSE,
            subset = NULL, ...)
```

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

```
## S3 method for class 'mlogit.data'
index(x, ...)
```

```
## S3 method for class 'mlogit.data'
x[i, j, drop = TRUE]
```

```
## S3 method for class 'mlogit.data'
x[[y]]
```

```
## S3 method for class 'mlogit.data'
x$y
```

```
## S3 replacement method for class 'mlogit.data'
object$y <- value
```

```
## S3 replacement method for class 'mlogit.data'
object[[y]] <- value
```

```
## S3 method for class 'mlogit.data'
mean(x, ...)
```

Arguments

<code>data</code>	a <code>data.frame</code> ,
<code>choice</code>	the variable indicating the choice made: it can be either a logical vector, a numerical vector with 0 where the alternative is not chosen, a factor with level 'yes' when the alternative is chosen
<code>shape</code>	the shape of the <code>data.frame</code> : whether <code>long</code> if each row is an alternative or <code>wide</code> if each row is an observation,
<code>varying</code>	the indexes of the variables that are alternative specific,
<code>sep</code>	the separator of the variable name and the alternative name (only relevant for a wide <code>data.frame</code>),
<code>alt.var</code>	the name of the variable that contains the alternative index (for a <code>long data.frame</code> only) or the name under which the alternative index will be stored (the default name is <code>alt</code>),
<code>chid.var</code>	the name of the variable that contains the choice index or the name under which the choice index will be stored,
<code>alt.levels</code>	the name of the alternatives: if null, for a <code>wide data.frame</code> , they are guessed from the variable names and the choice variable (both should be the same), for a <code>long data.frame</code> , they are guessed from the <code>alt.var</code> argument,
<code>id.var</code>	the name of the variable that contains the individual index if any,
<code>group.var</code>	the name of the variable that contains the group index if any,
<code>opposite</code>	returns the opposite of the specified variables,
<code>drop.index</code>	should the index variables be dropped from the <code>data.frame</code> ,
<code>ranked</code>	a logical value which is true if the response is a rank,
<code>subset</code>	a logical expression which defines the subset of observations to be selected,
<code>...</code>	further arguments passed to <code>reshape</code> .
<code>x, object</code>	a <code>mlogit.data</code> or a <code>pseries</code> object,
<code>i</code>	the rows to extract,
<code>j</code>	the columns to extract,
<code>drop</code>	a boolean, equal to <code>FALSE</code> if one wants that a <code>data.frame</code> is always returned,
<code>y</code>	the column of the <code>data.frame</code> to extract or to replace,
<code>value</code>	the replacement value,

Value

A `mlogit.data` object, which is a `data.frame` in `long` format, *i.e.* one line for each alternative. It has a `index` attribute, which is a `data.frame` that contains the index of the choice made (`chid`), the index of the alternative (`alt`) and, if any, the index of the individual (`id`) and of the alternative groups (`group`). The choice variable is a boolean which indicates the choice made. This function use `stats::reshape()` if the `data.frame` is in `wide` format.

Author(s)

Yves Croissant

See Also

```
stats::reshape()
```

Examples

```
# ModeChoice is a long data.frame

data("TravelMode", package = "AER")
TM <- mlogit.data(TravelMode, choice = "choice", shape = "long",
  alt.levels = c("air", "train", "bus", "car"))

# Same but the alt variable called mode is provided

TM <- mlogit.data(TravelMode, choice = "choice", shape = "long",
  alt.var = "mode")

# Same but the chid variable called individual is provided

TM <- mlogit.data(TravelMode, choice = "choice",
  shape = "long", id.var = "individual",
  alt.levels = c("air", "train", "bus", "car"))

# Same but with two own provided variables

TM <- mlogit.data(TravelMode, choice = "choice", shape = "long",
  id.var = "individual", alt.var = "mode")

# Same but with two own provided variables which are deleted from the
# data.frame

TM <- mlogit.data(TravelMode, choice = "choice", shape = "long",
  id.var = "individual", alt.var = "mode", drop.index = TRUE)

# Train is a wide data.frame with columns 'choiceid' is the choice
# index, the alternatives are named "ch1" and "ch2", the opposite of
# the variables is returned

data("Train", package = "mlogit")
Train <- mlogit.data(Train, choice = "choice", shape = "wide",
  varying = 4:11, alt.levels = c("A", "B"), sep = "_",
  opposite = c("price", "time", "change", "comfort"))

data("HC", package = "mlogit")
HC <- mlogit.data(HC, choice = "depvar", varying=c(2:8, 10:16), shape="wide")

# Game is a data.frame in wide format for which the response is a
# ranking variable

data("Game", package = "mlogit")
G <- mlogit.data(Game, shape="wide", varying = 1:12, alt.var = 'platform',
  drop.index = TRUE, choice="ch", ranked =TRUE)
```

```
# Game2 contains the same data, but in long format
data("Game2", package = "mlogit")
G2 <- mlogit.data(Game2, shape='long', choice="ch", alt.var = 'platform', ranked = TRUE)
```

mlogit.optim *Non-linear minimization routine*

Description

This function performs efficiently the optimization of the likelihood functions for multinomial logit models

Usage

```
mlogit.optim(logLik, start, method = c("bfgs", "nr", "bhhh"),
  iterlim = 2000, tol = 1e-06, ftol = 1e-08, steptol = 1e-10,
  print.level = 0, constPar = NULL, ...)
```

Arguments

logLik	the likelihood function to be maximized,
start	the initial value of the vector of coefficients,
method	the method used, one of 'nr' for Newton-Ralphson, 'bhhh' for Berndt-Hausman-Hall-Hall and 'bfgs',
iterlim	the maximum number of iterations,
tol	the value of the criteria for the gradient,
ftol	the value of the criteria for the function,
steptol	the value of the criteria for the step,
print.level	one of (0, 1, 2), the details of the printing messages. If 'print.level = 0', no information about the optimization process is provided, if 'print.level = 1' the value of the likelihood, the step and the stoping criteria is printing, if 'print.level = 2' the vectors of the parameters and the gradient are also printed.
constPar	a numeric or a character vector which indicates that some parameters should be treated as constant,
...	further arguments passed to <i>f</i> .

Details

The optimization is performed by updating, at each iteration, the vector of parameters by the amount $\text{step} * \text{direction}$, where step is a positive scalar and $\text{direction} = H^{-1} * g$, where g is the gradient and H^{-1} is an estimation of the inverse of the hessian. The choice of H^{-1} depends on the method chosen :

if $\text{method} = \text{'nr'}$, H is the hessian (*i.e.* is the second derivates matrix of the likelihood function),

if `method = 'bhhh'`, H is the outer-product of the individual contributions of each individual to the gradient,

if `method = 'bfgs'`, H^{-1} is updated at each iteration using a formula that uses the variations of the vector of parameters and the gradient. The initial value of the matrix is the inverse of the outer-product of the gradient (i.e. the bhhh estimator of the hessian).

The initial step is 1 and, if the new value of the function is less than the previous value, it is divided by two, until a higher value is obtained.

The routine stops when the gradient is sufficiently close to 0. The criteria is $g * H^{-1} * g$ which is compared to the `tol` argument. It also may stop if the number of iterations equals `iterlim`.

The function f has a `initial.value` argument which is the initial value of the likelihood. The function is then evaluated a first time with a step equals to one. If the value is lower than the initial value, the step is divided by two until the likelihood increases. The gradient is then computed and the function returns as attributes the gradient is the step. This method is more efficient than other functions available for R:

For the `optim` and the `maxLik` functions, the function and the gradient should be provided as separate functions. But, for multinomial logit models, both depends on the probabilities which are the most time-consuming elements of the model to compute.

For the `nlm` function, the fonction returns the gradient as an attribute. The gradient is therefore computed at each iteration, even when the function is computed with a step that is unable to increase the value of the likelihood.

Previous versions of `mlogit` depended on the 'maxLik' package. We kept the same interface, namely the `start`, `method`, `iterlim`, `tol`, `print.level` and `constPar` arguments.

The default method is 'bfgs', which is known to perform well, even if the likelihood function is not well behaved and the default value for `print.level = 1`, which means moderate printing.

A special default behavior is performed if a simple multinomial logit model is estimated. Indeed, for this model, the likelihood function is concave, the analytical hessian is simple to write and the optimization is straightforward. Therefore, in this case, the default method is 'nr' and `print.level = 0`.

Value

a list that contains the followings elements :

- `optimum`: the value of the function at the optimum, with attributes: `gradi` a matrix that contains the contribution of each individual to the gradient, `gradient` the gradient and, if `method = 'nr'`, `hessian` the hessian,
- `coefficients`: the vector of the parameters at the optimum,
- `est.stat`: a list that contains some information about the optimization : `'nb.iter'` the number of iterations, `'eps'` the value of the stoping criteria, `'method'` the method of optimization method used, `'message'`

Author(s)

Yves Croissant

Mode

Mode Choice

Description

A sample of 453 individuals for 4 transport modes.

Format

A dataframe containing :

- choice: one of car, carpool, bus or rail,
- cost.z: cost of mode z,
- time.z: time of mode z.

Source

Kenneth Train's home page¹¹.

ModeCanada

Mode Choice for the Montreal-Toronto Corridor

Description

A sample of 3880 travellers for the Montreal-Toronto corridor

Format

A dataframe containing

- case: the individual index,
- alt: the alternative, one of train, car, bus and air,
- choice: one if the mode is chosen, zero otherwise,
- cost: monetary cost,
- ivt: in vehicule time,
- ovt: out vehicule time,
- frequency: frequency,
- income: income,
- urban: urban,
- noalt: the number of alternatives available.

¹¹<http://elsa.berkeley.edu/~train/>

Source

kindly provided by S. Koppelman

References

Bhat CR (1995). "A heteroscedastic extreme value model of intercity travel mode choice." *Transportation Research Part B: Methodological*, **29**(6), 471 - 483. ISSN 0191-2615, <http://www.sciencedirect.com/science/article/pii/0191261595000156>.

Koppelman FS, Wen C (2000). "The paired combinatorial logit model: properties, estimation and application." *Transportation Research Part B: Methodological*, **34**(2), 75 - 89. ISSN 0191-2615, <http://www.sciencedirect.com/science/article/pii/S0191261599000120>.

Wen C, Koppelman FS (2001). "The generalized nested logit model." *Transportation Research Part B: Methodological*, **35**(7), 627 - 641. ISSN 0191-2615, <http://www.sciencedirect.com/science/article/pii/S019126150000045X>.

Examples

```
data("ModeCanada", package = "mlogit")
bususers <- with(ModeCanada, case[choice == 1 & alt == "bus"])
ModeCanada <- subset(ModeCanada, ! case %in% bususers)
ModeCanada <- subset(ModeCanada, noalt == 4)
ModeCanada <- subset(ModeCanada, alt != "bus")
ModeCanada$alt <- ModeCanada$alt[drop = TRUE]
KoppWen00 <- mlogit.data(ModeCanada, shape='long', chid.var = 'case',
                        alt.var = 'alt', choice='choice',
                        drop.index=TRUE)
pcl <- mlogit(choice~freq+cost+ivt+ovt, KoppWen00, relevel='car',
             nests='pcl', constPar=c('iv:train.air'))
```

Description

A sample of 632 American production units

Format

A dataframe containing:

- chid: the plant id,
- alt: the alternative,
- id: the owner id,
- choice: the chosen alternative,
- available: a dummy indicating that the alternative is available,

- env: the regulatory environment, one of 'regulated', 'deregulated' and 'public',
- post: dummy for post-combustion pollution control technology,
- cm: dummy for combustion modification technology,
- lnbn: dummy for low NOx burners technology,
- age: age of the plant (in deviation from the mean age),
- vcost: variable cost,
- kcost: capital cost.

Source

American Economic Association data archive¹².

References

Fowlie M (2010). "Emissions Trading, Electricity Restructuring, and Investment in Pollution Abatement." *American Economic Review*, **100**(3), 837-69. doi: 10.1257/aer.100.3.837¹³, <http://www.aeaweb.org/articles?id=10.1257/aer.100.3.837>.

plot.mlogit

Plot of the distribution of estimated random parameters

Description

Methods for `rpar` and `mlogit` objects which provide a plot of the distribution of one or all of the estimated random parameters

Usage

```
## S3 method for class 'mlogit'
plot(x, par = NULL, norm = NULL, type = c("density",
  "probability"), ...)

## S3 method for class 'rpar'
plot(x, norm = NULL, type = c("density", "probability"),
  ...)
```

Arguments

<code>x</code>	a <code>mlogit</code> or a <code>rpar</code> object,
<code>par</code>	a subset of the random parameters ; if <code>NULL</code> , all the parameters are selected,
<code>norm</code>	the coefficient's name for the <code>mlogit</code> method or the coefficient's value for the <code>rpar</code> method used for normalization,

¹²<http://aeaweb.org/aer/>

¹³<https://doi.org/10.1257/aer.100.3.837>

`type` the function to be plotted, whether the density or the probability density function,
`...` further arguments, passed to `plot.rpar` for the `mlogit` method and to `plot` for the `rpar` method.

Details

For the `rpar` method, one plot is drawn. For the `mlogit` method, one plot for each selected random parameter is drawn.

Author(s)

Yves Croissant

See Also

`mlogit()` the estimation of random parameters logit models and `rpar()` for the description of `rpar` objects and distribution for functions which return informations about the distribution of random parameters.

RiskyTransport

Risky Transportation Choices

Description

1793 choices by 561 individuals of a transport mode at Freetwon airport

Format

A dataframe containing:

- `id`: individual id,
- `choice`: 1 for the chosen mode,
- `mode`: one of `Helicopter`, `WaterTaxi`, `Ferry`, and `Hovercraft`,
- `cost`: the generalised cost of the transport mode,
- `risk`: the fatality rate, numbers of death per 100,000 trips,
- `weight`: weights,
- `seats`: ,
- `noise`: ,
- `crowdness`: ,
- `convloc`: ,
- `clientele`: ,
- `chid`: choice situation id,
- `african`: `yes` if born in Africa, `no` otherwise,

- lifeExp: declared life expectancy,
- dwage: declared hourly wage,
- iwage: imputed hourly wage,
- educ: level of education, one of low and high,
- fatalism: self-ranking of the degree of fatalism,
- gender: gender, one of female and male,
- age: age,
- haveChildren: yes if the traveler has children, no otherwise,
- swim: yes if the traveler knows how to swim, no, otherwise.

Source

American Economic Association data archive¹⁴.

References

León G, Miguel E (2017). “Risky Transportation Choices and the Value of a Statistical Life.” *American Economic Journal: Applied Economics*, **9**(1), 202-28. doi: 10.1257/app.20160140¹⁵, <http://www.aeaweb.org/articles?id=10.1257/app.20160140>.

rpar

random parameter objects

Description

rpar objects contain the relevant information about estimated random parameters. The homonymous function extract on rpar object from a mlogit object.

Usage

```
rpar(x, par = NULL, norm = NULL, ...)

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

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

¹⁴<http://aeaweb.org/aer/>

¹⁵<https://doi.org/10.1257/app.20160140>

Arguments

<code>x, object</code>	a <code>mlogit</code> object,
<code>par</code>	the name or the index of the parameters to be extracted ; if <code>NULL</code> , all the parameters are selected,
<code>norm</code>	the coefficient used for normalization if any,
<code>...</code>	further arguments.
<code>digits</code>	the number of digits
<code>width</code>	the width of the printed output

Details

`mlogit` objects contain an element called `rpar` which contain a list of `rpar` objects, one for each estimated random parameter. The `print` method prints the name of the distribution and the parameter, the `summary` behave like the one for numeric vectors.

Value

a `rpar` object, which contains:

- `dist`: the name of the distribution,
- `mean`: the first parameter of the distribution,
- `sigma`: the second parameter of the distribution,
- `name`: the name of the parameter.

Author(s)

Yves Croissant

See Also

`mlogit()` for the estimation of a random parameters logit model.

Description

Three tests for `mlogit` models: specific methods for the Wald test and the likelihood ration test and a new function for the score test

Usage

```
scoretest(object, ...)  
  
## S3 method for class 'mlogit'  
scoretest(object, ...)  
  
## Default S3 method:  
scoretest(object, ...)  
  
## S3 method for class 'mlogit'  
waldtest(object, ...)  
  
## S3 method for class 'mlogit'  
lrtest(object, ...)
```

Arguments

<code>object</code>	an object of class <code>mlogit</code> or a formula,
<code>...</code>	two kinds of arguments can be used. If <code>mlogit</code> arguments are introduced, initial model is updated using these arguments. If formula or other <code>mlogit</code> models are introduced, the standard behavior of <code>lmtest::waldtest()</code> and <code>lmtest::lrtest()</code> is followed.

Details

The `scoretest` function and `mlogit` method for `waldtest` and `lrtest` from the `lmtest` package provides the infrastructure to compute the three tests of hypothesis for `mlogit` objects.

The first argument must be a `mlogit` object. If the second one is a fitted model or a formula, the behaviour of the three functions is the one of the default methods of `waldtest` and `lrtest`: the two models provided should be nested and the hypothesis tested is that the constrained model is the 'right' model.

If no second model is provided and if the model provided is the constrained model, some specific arguments of `mlogit` should be provided to describe how the initial model should be updated. If the first model is the unconstrained model, it is tested versus the 'natural' constrained model; for example, if the model is a heteroscedastic logit model, the constrained one is the multinomial logit model.

Value

an object of class `htest`.

Author(s)

Yves Croissant

Examples

```

library("mlogit")
library("lmtest")
data("TravelMode", package = "AER")
ml <- mlogit(choice ~ wait + travel + vcost, TravelMode,
            shape = "long", chid.var = "individual", alt.var = "mode")
hl <- mlogit(choice ~ wait + travel + vcost, TravelMode,
            shape = "long", chid.var = "individual", alt.var = "mode",
            method = "bfgs", heterosc = TRUE)
lrtest(ml, hl)
waldtest(hl)
scoretest(ml, heterosc = TRUE)

```

Train

*Stated Preferences for Train Traveling***Description**

A sample of 235 Dutch individuals facing 2929 choice situations

Format

A dataframe containing:

- id: individual identifiant,
- choiceid: choice identifiant,
- choice: one of 'A' or 'B',
- price_z: price of proposition z (z = 'A', 'B') in cents of guilders,
- time_z: travel time of proposition z (z = 'A', 'B') in minutes,
- comfort_z: comfort of proposition z (z = 'A', 'B'), 0, 1 or 2 in decreasing comfort order,
- change_z: number of changes for proposition z (z = 'A', 'B').

Source

Journal of Applied Econometrics data archive¹⁶.

References

Ben-Akiva M, Bolduc D, Bradley M (1993). "Estimation of Travel Choice Models with Randomly Distributed Values of Time." Papers 9303, Laval - Recherche en Energie. <https://ideas.repec.org/p/fth/lavaen/9303.html>.

Meijer E, Rouwendal J (2006). "Measuring welfare effects in models with random coefficients." *Journal of Applied Econometrics*, **21**(2), 227-244. doi: 10.1002/jae.841¹⁷, <https://onlinelibrary.wiley.com/doi/pdf/10.1002/jae.841>
<https://onlinelibrary.wiley.com/doi/abs/10.1002/jae.841>.

¹⁶<http://jae.wiley.com/jae/>

¹⁷<https://doi.org/10.1002/jae.841>

vcov.mlogit *vcov method for mlogit objects*

Description

The `vcov` method for `mlogit` objects extract the covariance matrix of the coefficients, the errors or the random parameters.

Usage

```
## S3 method for class 'mlogit'
vcov(object, what = c("coefficient", "errors", "rpar"),
      subset = c("all", "iv", "sig", "sd", "sp", "chol"), type = c("cov",
        "cor", "sd"), refllevel = NULL, ...)

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

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

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

Arguments

<code>object</code>	a <code>mlogit</code> object (and a <code>vcov.mlogit</code> for the summary method),
<code>what</code>	indicates which covariance matrix has to be extracted : the default value is <code>coefficients</code> , in this case, <code>vcov</code> behaves as usual. If <code>what</code> equals <code>errors</code> the covariance matrix of the errors of the model is returned. Finally, if <code>what</code> equals <code>rpar</code> , the covariance matrix of the random parameters are extracted,
<code>subset</code>	the subset of the coefficients that have to be extracted (only relevant if <code>what = "coefficients"</code>),
<code>type</code>	with this argument, the covariance matrix may be returned (the default) ; the correlation matrix with the standard deviation on the diagonal may also be extracted,
<code>refllevel</code>	relevant for the extraction of the errors of a multinomial probit model ; in this case the covariance matrix of error differences is returned and, with this argument, the alternative used for differentiation is indicated,
<code>...</code>	further arguments.
<code>x</code>	a <code>vcov.mlogit</code> or a <code>summary.vcov.mlogit</code> object,
<code>digits</code>	the number of digits,
<code>width</code>	the width of the printing,

Details

This new interface replaces the `cor.mlogit` and `cov.mlogit` functions which are deprecated.

Author(s)

Yves Croissant

See Also

`mlogit()` for the estimation of multinomial logit models.