Package 'REndo'

August 5, 2019

Title Fitting Linear Models with Endogenous Regressors using Latent Instrumental Variables

Version 2.3.0

Date 2019-08-04

Description

Fits linear models with endogenous regressor using latent instrumental variable approaches. The methods included in the package are Lewbel's (1997) <doi:10.2307/2171884> higher moments approach as well as
Lewbel's (2012) <doi:10.1080/07350015.2012.643126> heteroscedasticity approach, Park and Gupta's (2012) <doi:10.1287/mksc.1120.0718> joint estimation method that uses Gaussian copula and Kim and Frees's (2007) <doi:10.1007/s11336-007-9008-1> multilevel generalized
method of moment approach that deals with endogeneity in a multilevel setting.
These are statistical techniques to address the endogeneity problem where no external instrumental variables are needed.
Note that with version 2.0.0 sweeping changes were introduced which greatly improve functionality and usability but break backwards compatibility.

License GPL-3

URL https://github.com/mmeierer/REndo

BugReports https://github.com/mmeierer/REndo/issues

NeedsCompilation no

Depends R (>= 3.4)

Imports methods (>= 3.4), stats (>= 3.4), utils (>= 3.4), Formula (>= 1.2), optimx (>= 2013.8.7), mvtnorm (>= 1.0-8), AER (>= 1.2-5), Matrix (>= 1.2-14), lme4 (>= 1.1-18-1), data.table (>= 1.11.8), corpcor (>= 1.6.9)

Suggests testthat, covr, R.rsp

Encoding UTF-8

RoxygenNote 6.1.1

VignetteBuilder R.rsp

Author Raluca Gui [cre, aut],
Markus Meierer [aut],
Rene Algesheimer [aut],
Patrik Schilter [aut]

Maintainer Raluca Gui <raluca.gui@business.uzh.ch>

Repository CRAN

Date/Publication 2019-08-05 16:00:02 UTC

R topics documented:

`confint.rendo.boots` *Confidence Intervals for Bootstrapped Model Parameters*

Description

Confidence Intervals for Bootstrapped Model Parameters

Usage

```
## S3 method for class 'rendo.boots'
confint(object, parm, level = 0.95, ...)
```

Arguments

object	a fitted model object with bootstrapped parameters. Typically from copulaCorrection
parm	a specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered.
level	the confidence level required.
...	ignored, for consistency with the generic function.

Details

Computes the two-sided percentile confidence intervals from the bootstrapped parameter estimates. The intervals are obtained by selecting the quantile of the bootstrapped parameter estimates corresponding to the given alpha level.

A minimum of $\frac{1}{min(level, 1-level)}$ parameters estimates are needed to derive the confidence interval. The reason for this is that there is otherwise no natural way to derive the percentiles (ie one cannot reasonably estimate the 95% quantile of only 7 values).

---

copulaCorrection *Fitting Linear Models Endogenous Regressors using Gaussian Copula*

Description

Fits linear models with continuous or discrete endogenous regressors (or a mixture of both) using Gaussian copulas, as presented in Park and Gupta (2012). This is a statistical technique to address the endogeneity problem where no external instrumental variables are needed. The important assumption of the model is that the endogenous variables should NOT be normally distributed, if continuous, preferably with a skewed distribution.

Usage

```
copulaCorrection(formula, data, num.boots = 1000, verbose = TRUE, ...)
```

Arguments

formula	A symbolic description of the model to be fitted. See the "Details" section for the exact notation.
data	A data.frame containing the data of all parts specified in the formula parameter.
num.boots	Number of bootstrapping iterations. Defaults to 1000.
verbose	Show details about the running of the function.

... Arguments for the log-likelihood optimization function in the case of a single continuous endogenous regressor. Ignored with a warning otherwise.

start.params A named vector containing a set of parameters to use in the first optimization iteration. The names have to correspond exactly to the names of the components specified in the `formula` parameter. If not provided, a linear model is fitted to derive them.

optimx.args A named list of arguments which are passed to `optimx`. This allows users to tweak optimization settings to their liking.

Details

Method:

The underlying idea of the joint estimation method is that using information contained in the observed data, one selects marginal distributions for the endogenous regressor and the structural error term, respectively. Then, the copula model enables the construction of a flexible multivariate joint distribution allowing a wide range of correlations between the two marginals.

Consider the model:

$$Y_t = \beta_0 + \beta_1 P_t + \beta_2 X_t + \epsilon_t$$

where $t = 1, .., T$ indexes either time or cross-sectional units, Y_t is a $1x1$ response variable, X_t is a kxn exogenous regressor, P_t is a $kx1$ continuous endogenous regressor, ϵ_t is a normally distributed structural error term with mean zero and $E(\epsilon^2) = \sigma_\epsilon^2$, α and β are model parameters.

The marginal distribution of the endogenous regressor P_t is obtained using the Epanechnikov kernel density estimator (Epanechnikov, 1969), as below:

$$\hat{h}(p) = \frac{1}{T \cdot b} \sum_{t=1}^{T} K\left(\frac{p - P_t}{b}\right)$$

where P_t is the endogenous regressor, $K(x) = 0.75 \cdot (1 - x^2) \cdot I(\|x\| <= 1)$ and the bandwidth b is the one proposed by Silverman (1986), and is equal to $b = 0.9 \cdot T^{-1/5} \cdot min(s, IQR/1.34)$. IQR is the interquartile range while s is the data sample standard deviation and T is the number of time periods observed in the data. After obtaining the joint distribution of the error term and the continuous endogenous regressor, the model parameters are estimated using maximum likelihood estimation.

The additional parameters used during model fitting and printed in `summary` hence are:

`rho` The correlation between the endogenous regressor and the error.

`sigma` The variance of the model's error.

With more than one continuous endogenous regressor or an endogenous discrete regressor, an alternative approach to the estimation using Gaussian copula should be applied. This approach is similar to the control function approach (Petrin and Train, 2010). The core idea is to apply OLS estimation on the original set of explanatory variables in the model equation above, plus an additional regressor $P_t^* = \Phi^{-1}(H(P_t))$. Here, $H(P_t)$ is the marginal distribution of the endogenous regressor P. Including this regressor solves the correlation between the endogenous regressor and the structural error, ϵ, OLS providing consistent parameter estimates. Due to identification problems, the discrete endogenous regressor cannot have a binomial distribution.

Hence, only in the case of a single continuous endogenous regressor maximum likelihood estimation is used. In all other cases, augmented OLS based on Gaussian copula is applied. This includes cases of multiple endogenous regressors of both discrete and continuous distributions.

In the case of discrete endogenous regressors, a random seed needs to be assigned because the marginal distribution function of the endogenous regressor is a step function in this case. This means that the value of P^* lies between 2 values, $\Phi^{-1}(H(P_t - 1))$ and $\Phi^{-1}(H(P_t))$. However, the reported upper and lower bounds of the 95% bootstrapped confidence interval gives indication of the variance of the estimates.

Since the inference procedure in both cases, augmented OLS and maximum likelihood, occurs in two stages (first the empirical distribution of the endogenous regressor is computed and then used in constructing the likelihood function), the standard errors are not correct. Therefore, in both cases, the standard errors and the confidence intervals are obtained based on the sampling distributions resulted from bootstrapping. Since the distribution of the bootstrapped parameters is highly skewed, we report the percentile confidence intervals. Moreover, the variance-covariance matrix is also computed based on the bootstrapped parameters, and not based on the Hessian.

Formula parameter: The `formula` argument follows a two part notation:

A two-sided formula describing the model (e.g. `y ~ X1 + X2 + P`) to be estimated and a second right-hand side part in which the endogenous regressors and their distributional assumptions are indicated (e.g. `continuous(P)`). These two parts are separated by a single vertical bar (`|`). In the second part, the special functions `continuous`, `discrete`, or a combination of both, are used to indicate the endogenous regressors and their respective distribution. Both functions use the `...` parameter in which the respective endogenous regressors is specified.

Note that no argument to `continuous` or `discrete` is to be supplied as character but as symbols without quotation marks.

See the example section for illustrations on how to specify the `formula` parameter.

Value

For all cases, an object of classes `rendo.copula.correction`, `rendo.boots`, and `rendo.base` is returned which is a list and contains the following components:

`formula`	The formula given to specify the fitted model.
`terms`	The terms object used for model fitting.
`model`	The model.frame used for model fitting.
`coefficients`	A named vector of all coefficients resulting from model fitting.
`names.main.coefs`	a vector specifying which coefficients are from the model. For internal usage.
`names.vars.continuous`	The names of the continuous endogenous regressors.
`names.vars.discrete`	The names of the discrete endogenous regressors.
`fitted.values`	Fitted values at the found solution.
`residuals`	The residuals at the found solution.
`boots.params`	The bootstrapped coefficients.

For the case of a single continuous endogenous regressor, the returned object further contains the following components:

`start.params` A named vector with the initial set of parameters used to optimize the log-likelihood function.

`res.optimx` The result object returned by the function `optimx` after optimizing the log-likelihood function.

For all other cases, the returned object further contains the following component:

`res.lm.real.data`
The linear model fitted on the original data together with generated p.star data.

The function `summary` can be used to obtain and print a summary of the results. Depending on the returned object, the generic accessor functions `coefficients`, `fitted.values`, `residuals`, `vcov`, `logLik`, `AIC`, `BIC`, and `nobs` are available.

References

Park, S. and Gupta, S., (2012), "Handling Endogenous Regressors by Joint Estimation Using Copulas", Marketing Science, 31(4), 567-86.

Epanechnikov V (1969). "Nonparametric Estimation of a Multidimensional Probability Density." Teoriya veroyatnostei i ee primeneniya, 14(1), 156–161. Silverman B (1986). "Density Estimation for Statistics and Data Analysis". CRC Monographs on Statistics and Applied Probability. London: Chapman & Hall.

Petrin A, Train K (2010). "A Control Function Approach to Endogeneity in Consumer Choice Models." Journal of Marketing Research, 47(1), 3–13.

See Also

`summary` for how fitted models are summarized

`vcov` for how the variance-covariance matrix is derived

`confint` for how confidence intervals are derived

`optimx` for possible elements of parameter `optimx.arg`

Examples

```
data("dataCopCont")
data("dataCopCont2")
data("dataCopDis")
data("dataCopDis2")
data("dataCopDisCont")


## Not run:
# Single continuous: log-likelihood optimization
c1 <- copulaCorrection(y~X1+X2+P|continuous(P), num.boots=10, data=dataCopCont)
# same as above, with start.parameters and number of bootstrappings
c1 <- copulaCorrection(y~X1+X2+P|continuous(P), num.boots=10, data=dataCopCont,
                       start.params = c("(Intercept)"=1, X1=1, X2=-2, P=-1))

# All following examples fit linear model with Gaussian copulas
```

```
# 2 continuous endogenous regressors
c2 <- copulaCorrection(y~X1+X2+P1+P2|continuous(P1, P2),
                       num.boots=10, data=dataCopCont2)
# same as above
c2 <- copulaCorrection(y~X1+X2+P1+P2|continuous(P1)+continuous(P2),
                       num.boots=10, data=dataCopCont2)

# single discrete endogenous regressor
d1 <- copulaCorrection(y~X1+X2+P|discrete(P), num.boots=10, data=dataCopDis)

# two discrete endogenous regressor
d2 <- copulaCorrection(y~X1+X2+P1+P2|discrete(P1)+discrete(P2),
                       num.boots=10, data=dataCopDis2)
# same as above but less bootstrap runs
d2 <- copulaCorrection(y~X1+X2+P1+P2|discrete(P1, P2), num.boots = 10,
                      data=dataCopDis2)

# single discrete, single continuous
cd <- copulaCorrection(y~X1+X2+P1+P2|discrete(P1)+continuous(P2),
                       num.boots=10, data=dataCopDisCont)

# For single continuous only: use own optimization settings (see optimx())
# set maximum number of iterations to 50'000
res.c1 <- copulaCorrection(y~X1+X2+P|continuous(P),
                           optimx.args = list(itnmax = 50000),
                            num.boots=10, data=dataCopCont)

# print detailed tracing information on progress
 res.c1 <- copulaCorrection(y~X1+X2+P|continuous(P),
                            optimx.args = list(control = list(trace = 6)),
                             num.boots=10, data=dataCopCont)

# use method L-BFGS-B instead of Nelder-Mead and print report every 50 iterations
 res.c1 <- copulaCorrection(y~X1+X2+P|continuous(P),
                            optimx.args = list(method = "L-BFGS-B",
                                               control=list(trace = 2, REPORT=50)),
                             num.boots=10, data=dataCopCont)

# For coef(), the parameter "complete" determines if only the
# main model parameters or also the auxiliary coefficients are returned

c1.all.coefs <- coef(res.c1) # also returns rho and sigma
# same as above
c1.all.coefs <- coef(res.c1, complete = TRUE)

# only main model coefs
c1.main.coefs <- coef(res.c1, complete = FALSE)


## End(Not run)
```

dataCopCont *Simulated Dataset with One Endogenous Continuous Regressor*

Description

A dataset with two exogenous regressors, X1,X2, and one endogenous, continuous regressor, P, having a T-distribution with 3 degrees of freedom. An intercept and a dependent variable, y, are also included. The true parameter values for the coefficients are: b0 = 2, b1 = 1.5, b2 = -3 and the coefficient of the endogenous regressor, P, is equal to a1 = -1.

Usage

```
data("dataCopCont")
```

Format

A data frame with 2500 observations on 4 variables:

y a numeric vector representing the dependent variable.

X1 a numeric vector, normally distributed and exogenous.

X2 a numeric vector, normally distributed and exogenous.

P a numeric vector, continuous and endogenous having T-distribution with 3 degrees of freedom.

Author(s)

Raluca Gui <raluca.gui@business.uzh.ch>

dataCopCont2 *Simulated Dataset with Two Endogenous Continuous Regressor*

Description

A dataset with two exogenous regressors, X1,X2, and two endogenous, continuous regressors, P1 and P2, having a T-distribution with 3 degrees of freedom. An intercept and a dependent variable, y, are also included. The true parameter values for the intercept and the exogenous regressors' coefficients are: b0 = 2, b1 = 1.5, b2 = -3. The coefficient of the endogenous regressor P1 is equal to a1 = -1 and of P2 is equal to a2 = 0.8.

Usage

```
data("dataCopCont2")
```

Format

A data frame with 2500 observations on 5 variables:

`y` a numeric vector representing the dependent variable.

`X1` a numeric vector, normally distributed and exogenous.

`X2` a numeric vector, normally distributed and exogenous.

`P1` a numeric vector, continuous and endogenous having T-distribution with 3 degrees of freedom.

`P2` a numeric vector, continuous and endogenous having T-distribution with 3 degrees of freedom.

Author(s)

Raluca Gui <raluca.gui@business.uzh.ch>

---

dataCopDis *Simulated Dataset with One Endogenous Discrete Regressor*

Description

A dataset with two exogenous regressors, `X1`,`X2`, and one endogenous, discrete (Poisson distributed) regressor, `P`. An intercept and a dependent variable, `y`, are also included. The true parameter values for the coefficients are: `b0 = 2`, `b1 = 1.5`, `b2 = -3` and the coefficient of the endogenous regressor, P, is equal to `a1 = -1`.

Usage

```
data("dataCopDis")
```

Format

A data frame with 2500 observations on 4 variables:

`y` a numeric vector representing the dependent variable.

`X1` a numeric vector, normally distributed and exogenous.

`X2` a numeric vector, normally distributed and exogenous.

`P` a numeric vector, continuous and endogenous having T-distribution with 3 degrees of freedom.

Author(s)

Raluca Gui <raluca.gui@business.uzh.ch>

dataCopDis2 *Simulated Dataset with Two Endogenous Discrete Regressors*

Description

A dataset with two exogenous regressors, X1,X2, and two endogenous, discrete (Poisson distributed) regressors, P1 and P2. An intercept and a dependent variable, y, are also included. The true parameter values for the coefficients of the intercept and the exogenous variables are: b0 = 2, b1 = 1.5, b2 = -3. The true parameter values for the coefficients of the endogenous regressors are a1 = -1 for P1 and a2 = 0.8 for P2.

Usage

```
data("dataCopDis2")
```

Format

A data frame with 2500 observations on 5 variables:

- y a numeric vector representing the dependent variable.
- X1 a numeric vector, normally distributed and exogenous.
- X2 a numeric vector, normally distributed and exogenous.
- P1 a numeric vector, having a Poisson distribution with parameter lambda equal to 3, and endogenous.
- P2 a numeric vector, having a Poisson distribution with parameter lambda equal to 3, and endogenous.

Author(s)

Raluca Gui <raluca.gui@business.uzh.ch>

dataCopDisCont *Simulated Dataset with Two Endogenous Regressors*

Description

A dataset with two exogenous regressors, X1,X2, and two endogenous regressors, P1, having a Poisson distribution with lambda parameter equal to 3, and P2, having a T-distribution with 3 degrees of freedom. An intercept and a dependent variable, y, are also included. The true parameter values for the coefficients are: b0 = 2, b1 = 1.5, b2 = -3 and the coefficient of the endogenous regressor P1 is set to a1 = -1 and of P2 is set to a2=0.8.

Usage

```
data("dataCopDisCont")
```

Format

A data frame with 2500 observations on 5 variables:

- `y` a numeric vector representing the dependent variable.
- `X1` a numeric vector, normally distributed and exogenous.
- `X2` a numeric vector, normally distributed and exogenous.
- `P1` a numeric vector, continuous and endogenous having Poisson distribution with parameter lambda equal to 3.
- `P2` a numeric vector, continuous and endogenous having T-distribution with 3 degrees of freedom.

Author(s)

Raluca Gui <raluca.gui@business.uzh.ch>

---

dataHetIV *Simulated Dataset with One Endogenous Continuous Regressor*

Description

A dataset with two exogenous regressors, `X1`,`X2`, one endogenous, continuous regressor `P`, and the dependent variable `y`. The true parameter values for the coefficients are: `b0 = 2`, `b1 = 1.5`, `b2 = 3` and the coefficient of the endogenous regressor, `P`, is equal to `a1 = -1`.

Usage

```
data("dataHetIV")
```

Format

A data frame with 2500 observations on 4 variables:

- `y` a numeric vector representing the dependent variable.
- `X1` a numeric vector, normally distributed and exogenous.
- `X2` a numeric vector, normally distributed and exogenous.
- `P` a numeric vector, continuous and endogenous regressor, normally distributed.

Author(s)

Raluca Gui <raluca.gui@business.uzh.ch>

dataHigherMoments *Simulated Dataset with One Endogenous Regressor*

Description

A dataset with two exogenous regressors, X1,X2, and one endogenous, continuous regressor P. An intercept and a dependent variable, y, are also included. The true parameter values for the coefficients are: b0 = 2, b1 = 1.5, b2 = 3 and the coefficient of the endogenous regressor, P, is equal to a1 = -1.

Usage

```
data("dataHigherMoments")
```

Format

A data frame with 2500 observations on 4 variables:

y a numeric vector representing the dependent variable.

X1 a numeric vector, normally distributed and exogenous.

X2 a numeric vector, normally distributed and exogenous.

P a numeric vector, continuous and endogenous regressor, normally distributed.

Author(s)

Raluca Gui <raluca.gui@business.uzh.ch>

Examples

```
data("dataHigherMoments")
# to recover the parameters,
#   on average over many simulations
higherMomentsIV(formula = y ~ X1 + X2 + P|P|IIV(iiv=yp),
               data=dataHigherMoments)
```

dataLatentIV *Simulated Dataset with One Endogenous Continuous Regressor*

Description

A dataset with one endogenous regressor P, an instrument Z used to build P, an intercept and a dependent variable, y. The true parameter values for the coefficients are: b0 = 3 for the intercept and a1 = -1 for P.

Usage

```
data("dataLatentIV")
```

Format

A data frame with 2500 observations on 3 variables:

`y` a numeric vector representing the dependent variable.

`P` a numeric vector representing the endogenous variable.

`Z` a numeric vector used in the construction of the endogenous variable, P.

Author(s)

Raluca Gui <raluca.gui@business.uzh.ch>

---

dataMultilevelIV *Multilevel Simulated Dataset - Three Levels*

Description

A dataset simulated to exemplify the use of the `multilevelIV()` function. It has 2645 observations, clustered into 40 level-three variables and 1312 observations at level two. The endogenous regressor is `X15` with a true coefficient value of -1.

Usage

```
data("dataMultilevelIV")
```

Format

A data frame with 2645 observations clustered into 40 level-three variables and 1312 level-two variables.

`y` a numeric vector representing the dependent variable.

`X11` a level-one numeric vector representing a categorical exogenous variable with true parameter value equal to 3.

`X12` a level-one numeric vector representing a binomial distributed exogenous variable with true parameter value equal to 9.

`X13` a level-one numeric vector representing a binomial distributed exogenous variable with true parameter value equal to -2.

`X14` a level-two numeric vector representing a normally distributed exogenous variable with true parameter value equal to 2.

`X15` a level-two numeric vector representing a normally distributed endogenous variable, correlated with the level-two errors. It true parameter value equals to -1 and it has a correlation with the level two errors equal to 0.7.

`X21` a level-two numeric vector representing a binomial distributed exogenous variable with true parameter value equal to -1.5.

`X22` a level-two numeric vector representing a binomial distributed exogenous variable with true parameter value equal to -4.

`X23` a level-two numeric vector representing a binomial distributed exogenous variable with true parameter value equal to -3.

`X24` a level-teo numeric vector representing a normally distributed exogenous variable with true parameter value equal to 6.

`X31` a level-three numeric vector representing a normally distributed exogenous variable with true parameter value equal to 0.5.

`X32` a level-three numeric vector representing a truncated normally distributed exogenous variable with true parameter value equal to 0.1.

`X33` a level-three numeric vector representing a truncated normally distributed exogenous variable with true parameter value equal to -0.5.

`SID` a numeric vector identifying each level-three observations.

`CID` a numeric vector identifying each level-two observations.

Author(s)

Raluca Gui <raluca.gui@business.uzh.ch>

`hetErrorsIV` *Fitting Linear Models with Endogenous Regressors using Heteroskedastic Covariance Restrictions*

Description

This function estimates the model parameters and associated standard errors for a linear regression model with one or more endogenous regressors. Identification is achieved through heteroscedastic covariance restrictions within the triangular system as proposed in Lewbel(2012).

Usage

```
hetErrorsIV(formula, data, verbose = TRUE)
```

Arguments

`formula`	A symbolic description of the model to be fitted. See the "Details" section for the exact notation.
`data`	A data.frame containing the data of all parts specified in the formula parameter.
`verbose`	Show details about the running of the function.

Details

Method: The method proposed in Lewbel(2012) identifies structural parameters in regression models with endogenous regressors by means of variables that are uncorrelated with the product of heteroskedastic errors. The instruments are constructed as simple functions of the model's data. The method can be applied when no external instruments are available or to supplement external instruments to improve the efficiency of the IV estimator. Consider the model in the equation:

$$y_t = \beta_0 + \beta_1 P_t + \beta_2 X_t + \epsilon_t$$

where $t = 1, .., T$ indexes either time or cross-sectional units.The endogeneity problem arises from the correlation of P_t and ϵ_t. As such:

$$P_t = \gamma Z_t + \nu_t,$$

where Z_t is a subset of variables in X_t.

The errors, ϵ and ν, may be correlated with each other. Structural parameters are identified by an ordinary two-stage least squares regression of Y on X and P, using X and $[Z - E(Z)]\nu$ as instruments. A vital assumption for identification is that $cov(Z, \nu^2) \neq 0$. The strength of the instrument is proportional to the covariance of $(Z - \bar{Z})\nu$ with ν, which corresponds to the degree of heteroskedasticity of ν with respect to Z (Lewbel 2012).

The assumption that the covariance between Z and the squared error is different from zero can be empirically tested (this is checked in the background when calling the function). If it is zero or close to zero, the instrument is weak, producing imprecise estimates, with large standard errors.

Formula parameter: The `formula` argument follows a four part notation:

A two-sided formula describing the model (e.g. `y ~ X1 + X2 + P`), a single endogenous regressor (e.g. `P`), and the exogenous variables from which the internal instrumental variables should be build (e.g. `IIV(X1) + IIV(X2)`), each part separated by a single vertical bar (`|`).

The instrumental variables that should be built are specified as (multiple) functions, one for each instrument. This function is `IIV` and uses the following arguments:

`...` The exogenous regressors to build the internal instruments from. If more than one is given, separate instruments are built for each.

Note that no argument to `IIV` is to be supplied as character but as symbols without quotation marks.

Optionally, additional external instrumental variables to also include in the instrumental variable regression can be specified. These external instruments have to be already present in the data and are provided as the fourth right-hand side part of the formula, again separated by a vertical bar.

See the example section for illustrations on how to specify the `formula` parameter.

Value

Returns an object of classes `rendo.ivreg` and `ivreg`, It extends the object returned from function `ivreg` of package `AER` and slightly modifies it by adapting the `call` and `formula` components. The `summary` function prints additional diagnostic information as described in documentation for `summary.ivreg`.

All generic accessor functions for `ivreg` such as `anova`, `hatalues`, or `vcov` are available.

References

Lewbel, A. (2012). Using Heteroskedasticity to Identify and Estimate Mismeasured and Endogenous Regressor Models, Journal of Business & Economic Statistics, 30(1), 67-80.

Angrist, J. and Pischke, J.S. (2009). Mostly Harmless Econometrics: An Empiricists Companion, Princeton University Press.

See Also

ivreg

Examples

```
data("dataHetIV")
# P is the endogenous regressor in all examples

# 2 IVs, one from X1, one from X2
het <- hetErrorsIV(y~X1+X2+P|P|IIV(X1)+IIV(X2), data=dataHetIV)
# same as above
het <- hetErrorsIV(y~X1+X2+P|P|IIV(X1,X2), data=dataHetIV)

# use X2 as an external IV
het <- hetErrorsIV(y~X1+P|P|IIV(X1)|X2, data=dataHetIV)

summary(het)
```

---

higherMomentsIV *Fitting Linear Models with Endogenous Regressors using Lewbel's Higher Moments Approach*

Description

Fits linear models with one endogenous regressor using internal instruments built using the approach described in Lewbel A. (1997). This is a statistical technique to address the endogeneity problem where no external instrumental variables are needed. The implementation allows the incorporation of external instruments if available. An important assumption for identification is that the endogenous variable has a skewed distribution.

Usage

```
higherMomentsIV(formula, data, verbose = TRUE)
```

Arguments

formula	A symbolic description of the model to be fitted. See the "Details" section for the exact notation.
data	A data.frame containing the data of all parts specified in the formula parameter.
verbose	Show details about the running of the function.

Details

Method:

Consider the model:

$$Y_t = \beta_0 + \beta_1 X_t + \alpha P_t + \epsilon_t \quad (1)$$

$$P_t = \gamma Z_t + \nu_t \quad (2)$$

The observed data consist of Y_t, X_t and P_t, while Z_t, ϵ_t, and ν_t are unobserved. The endogeneity problem arises from the correlation of P_t with the structural error ϵ_t, since $E(\epsilon\nu) \neq 0$. The requirement for the structural and measurement error is to have mean zero, but no restriction is imposed on their distribution.

Let $\bar{S}$ be the sample mean of a variable S_t and $G_t = G(X_t)$ for any given function G that has finite third own and cross moments. Lewbel(1997) proves that the following instruments can be constructed and used with two-stage least squares to obtain consistent estimates:

$$q_{1t} = (G_t - \bar{G}) \quad (3a)$$

$$q_{2t} = (G_t - \bar{G})(P_t - \bar{P}) \quad (3b)$$

$$q_{3t} = (G_t - \bar{G})(Y_t - \bar{Y}) \quad (3c)$$

$$q_{4t} = (Y_t - \bar{Y})(P_t - \bar{P}) \quad (3d)$$

$$q_{5t} = (P_t - \bar{P})^2 \quad (3e)$$

$$q_{6t} = (Y_t - \bar{Y})^2 \quad (3f)$$

Instruments in equations $3e$ and $3f$ can be used only when the measurement and the structural errors are symmetrically distributed. Otherwise, the use of the instruments does not require any distributional assumptions for the errors. Given that the regressors $G(X) = X$ are included as instruments, $G(X)$ should not be linear in X in equation $3a$.

Let small letter denote deviation from the sample mean: $s_i = S_i - \bar{S}$. Then, using as instruments the variables presented in equations 3 together with 1 and X_t, the two-stage-least-squares estimation will provide consistent estimates for the parameters in equation 1 under the assumptions exposed in Lewbel(1997).

Formula parameter:

The `formula` argument follows a four part notation:

A two-sided formula describing the model (e.g. `y ~ X1 + X2 + P`), a single endogenous regressor (e.g. `P`), and the exogenous variables from which the internal instrumental variables should be build (e.g. `IIV(iiv=y2)`), each part separated by a single vertical bar (`|`).

The instrumental variables that should be built are specified as (multiple) functions, one for each instrument. This function is `IIV` and uses the following arguments:

`iiv` Which internal instrument to build. One of `g,gp,gy,yp,p2,y2` can be chosen.

`g` Which function `g` represents in `iiv`. One of `x2,x3,lnx,1/x` can be chosen. Only required if the type of internal instrument demands it.

`...` The exogenous regressors to build the internal instrument. If more than one is given, separate instruments are built for each. Only required if the type of internal instrument demands it.

Note that no argument to IIV is to be supplied as character but as symbols without quotation marks.

Optionally, additional external instrumental variables to also include in the instrumental variable regression can be specified. These external instruments have to be already present in the data and are provided as the fourth right-hand side part of the formula, again separated by a vertical bar.

See the example section for illustrations on how to specify the formula parameter.

Value

Returns an object of classes rendo.ivreg and ivreg, It extends the object returned from function ivreg of package AER and slightly modifies it by adapting the call and formula components. The summary function prints additional diagnostic information as described in documentation for summary.ivreg.

All generic accessor functions for ivreg such as anova, hatalues, or vcov are available.

References

Lewbel A (1997). "Constructing Instruments for Regressions with Measurement Error When No Additional Data are Available, With an Application to Patents and R&D." Econometrica, 65(5), 1201–1213.

See Also

ivreg

Examples

```
data("dataHigherMoments")
# P is the endogenous regressor in all examples

# 2 IVs with g*p, g=x^2, separately for each regressor X1 and X2.
hm <- higherMomentsIV(y~X1+X2+P|P|IIV(iiv=gp, g=x2, X1, X2),
                      data = dataHigherMoments)
# same as above
hm <- higherMomentsIV(y~X1+X2+P|P|IIV(iiv=gp, g=x2, X1) +
                                  IIV(iiv=gp, g=x2, X2),
                      data = dataHigherMoments)

# 3 different IVs
hm <- higherMomentsIV(y~X1+X2+P|P|IIV(iiv=y2) + IIV(iiv=yp) +
                                  IIV(iiv=g,g=x3,X1),
                      data = dataHigherMoments)

# use X2 as external IV
hm <- higherMomentsIV(y~X1+P|P|IIV(iiv=y2)+IIV(iiv=g,g=lnx,X1)| X2,
                      data = dataHigherMoments)
summary(hm)
```

`latentIV` *Fitting Linear Models with one Endogenous Regressor using Latent Instrumental Variables*

Description

Fits linear models with one endogenous regressor and no additional explanatory variables using the latent instrumental variable approach presented in Ebbes, P., Wedel, M., Böckenholt, U., and Steerneman, A. G. M. (2005). This is a statistical technique to address the endogeneity problem where no external instrumental variables are needed. The important assumption of the model is that the latent variables are discrete with at least two groups with different means and the structural error is normally distributed.

Usage

```
latentIV(formula, data, start.params = c(), optimx.args = list(),
  verbose = TRUE)
```

Arguments

`formula`	A symbolic description of the model to be fitted. Of class "formula".
`data`	A data.frame containing the data of all parts specified in the formula parameter.
`start.params`	A named vector containing a set of parameters to use in the first optimization iteration. The names have to correspond exactly to the names of the components specified in the formula parameter. If not provided, a linear model is fitted to derive them.
`optimx.args`	A named list of arguments which are passed to `optimx`. This allows users to tweak optimization settings to their liking.
`verbose`	Show details about the running of the function.

Details

Let's consider the model:

$$Y_t = \beta_0 + \alpha P_t + \epsilon_t$$
$$P_t = \pi^{'} Z_t + \nu_t$$

where $t = 1, .., T$ indexes either time or cross-sectional units, Y_t is the dependent variable, P_t is a k x 1 continuous, endogenous regressor, ϵ_t is a structural error term with mean zero and $E(\epsilon^2) = \sigma_\epsilon^2$, α and β_0 are model parameters. Z_t is a l x 1 vector of instruments, and ν_t is a random error with mean zero and $E(\nu^2) = \sigma_\nu^2$. The endogeneity problem arises from the correlation of P and ϵ_t through $E(\epsilon\nu) = \sigma_{\epsilon\nu}$

`latentIV` considers $Z_t^{'}$ to be a latent, discrete, exogenous variable with an unknown number of groups m and π is a vector of group means. It is assumed that Z is independent of the error terms ϵ and ν and that it has at least two groups with different means. The structural and random errors are considered normally distributed with mean zero and variance-covariance matrix Σ:

$$\Sigma = \begin{pmatrix} \sigma_\epsilon^2 & \sigma_{\epsilon\nu} \\ \sigma_{\epsilon\nu} & \sigma_\nu^2 \end{pmatrix}$$

The identification of the model lies in the assumption of the non-normality of P_t, the discreteness of the unobserved instruments and the existence of at least two groups with different means.

The method has been implemented such that the latent variable has two groups. Ebbes et al.(2005) show in a Monte Carlo experiment that even if the true number of the categories of the instrument is larger than two, estimates are approximately consistent. Besides, overfitting in terms of the number of groups/categories reduces the degrees of freedom and leads to efficiency loss. For a model with additional explanatory variables a Bayesian approach is needed, since in a frequentist approach identification issues appear.

Identification of the parameters relies on the distributional assumptions of the latent instruments as well as that of the endogenous regressor P_t. Specifically, the endogenous regressor should have a non-normal distribution while the unobserved instruments, Z, should be discrete and have at least two groups with different means Ebbes, Wedel, and Böckenholt (2009). A continuous distribution for the instruments leads to an unidentified model, while a normal distribution of the endogenous regressor gives rise to inefficient estimates.

Additional parameters used during model fitting and printed in `summary` are:

pi1 The instrumental variables Z are assumed to be divided into two groups. `pi1` represents the estimated group mean of the first group.

pi2 The estimated group mean of the second group of the instrumental variables Z.

theta5 The probability of being in the first group of the instruments.

theta6 The variance, σ_ϵ^2

theta7 The covariance, $\sigma_{\epsilon\nu}$

theta8 The variance, σ_ν^2

Value

An object of classes `rendo.latent.IV` and `rendo.base` is returned which is a list and contains the following components:

`formula`	The formula given to specify the fitted model.
`terms`	The terms object used for model fitting.
`model`	The model.frame used for model fitting.
`coefficients`	A named vector of all coefficients resulting from model fitting.
`names.main.coefs`	a vector specifying which coefficients are from the model. For internal usage.
`start.params`	A named vector with the initial set of parameters used to optimize the log-likelihood function.
`res.optimx`	The result object returned by the function `optimx` after optimizing the log-likelihood function.
`hessian`	A named, symmetric matrix giving an estimate of the Hessian at the found solution.

m.delta.diag	A diagonal matrix needed when deriving the vcov to apply the delta method on theta5 which was transformed during the LL optimization.
fitted.values	Fitted values at the found optimal solution.
residuals	The residuals at the found optimal solution.

The function summary can be used to obtain and print a summary of the results. The generic accessor functions coefficients, fitted.values, residuals, vcov, confint, logLik, AIC, BIC, case.names, and nobs are available.

References

Ebbes, P., Wedel,M., Böckenholt, U., and Steerneman, A. G. M. (2005). 'Solving and Testing for Regressor-Error (in)Dependence When no Instrumental Variables are Available: With New Evidence for the Effect of Education on Income'. Quantitative Marketing and Economics, 3:365–392.

Ebbes P., Wedel M., Böckenholt U. (2009). "Frugal IV Alternatives to Identify the Parameter for an Endogenous Regressor." Journal of Applied Econometrics, 24(3), 446–468.

See Also

summary for how fitted models are summarized

optimx for possible elements of parameter optimx.arg

Examples

```
data("dataLatentIV")

# function call without any initial parameter values
l  <- latentIV(y ~ P, data = dataLatentIV)
summary(l)

# function call with initial parameter values given by the user
l1 <- latentIV(y ~ P, start.params = c("(Intercept)"=2.5, P=-0.5),
               data = dataLatentIV)
summary(l1)

# use own optimization settings (see optimx())
# set maximum number of iterations to 50'000
l2 <- latentIV(y ~ P, optimx.args = list(itnmax = 50000),
               data = dataLatentIV)

# print detailed tracing information on progress
l3 <- latentIV(y ~ P, optimx.args = list(control = list(trace = 6)),
               data = dataLatentIV)

# use method L-BFGS-B instead of Nelder-Mead and print report every 50 iterations
l4 <- latentIV(y ~ P, optimx.args = list(method = "L-BFGS-B", control=list(trace = 2, REPORT=50)),
               data = dataLatentIV)

# read out all coefficients, incl auxiliary coefs
```

```
lat.all.coefs <- coef(l4)
# same as above
lat.all.coefs <- coef(l4, complete = TRUE)
# only main model coefs
lat.main.coefs <- coef(l4, complete = FALSE)
```

`multilevelIV` *Fitting Multilevel GMM Estimation with Endogenous Regressors*

Description

Estimates multilevel models (max. 3 levels) employing the GMM approach presented in Kim and Frees (2007). One of the important features is that, using the hierarchical structure of the data, no external instrumental variables are needed, unlike traditional instrumental variable techniques. Specifically, the approach controls for endogeneity at higher levels in the data hierarchy. For example, for a three-level model, endogeneity can be handled either if present at level two, at level three or at both levels. Level one endogeneity, where the regressors are correlated with the structural errors (errors at level one), is not addressed. Moreover, if considered, random slopes cannot be endogenous. Also, the dependent variable has to have a continuous distribution. The function returns the coefficient estimates obtained with fixed effects, random effects and the GMM estimator proposed by Kim and Frees (2007), such that a comparison across models can be done. Asymptotically, the multilevel GMM estimators share the same properties of corresponding fixed effects estimators, but they allow the estimation of all the variables in the model, unlike the fixed effects counterpart.

To facilitate the choice of the estimator to be used for the given data, the function also conducts omitted variable test based on the Hausman-test for panel data (Hausman, 1978). It allows to compare a robust estimator and an estimator that is efficient under the null hypothesis of no omitted variables, and to compare two robust estimators at different levels. The results of these tests are returned when calling `summary()` on a fitted model.

Usage

```
multilevelIV(formula, data, lmer.control = lmerControl(optimizer =
  "Nelder_Mead", optCtrl = list(maxfun = 1e+05)), verbose = TRUE)
```

Arguments

`formula`	A symbolic description of the model to be fitted. See the "Details" section for the exact notation.
`data`	A data.frame containing the data of all parts specified in the formula parameter.
`lmer.control`	An output from `lmerControl` that will be used to fit the `lmer` model from which the variance and correlation are obtained.
`verbose`	Show details about the running of the function.

Details

Method: Multilevel modeling is a generalization of regression methods that recognize the existence of such data hierarchies by allowing for residual components at each level in the hierarchy. For example, a three-level multilevel model which allows for grouping of students within classrooms, over time, would include time, student and classroom residuals (see equation below). Thus, the residual variance is partitioned into four components: between-classroom (the variance of the classroom-level residuals), within-classroom (the variance of the student-level residuals), between student (the variance of the student-level residuals) and within-student (the variance of the time-level residuals). The classroom residuals represent the unobserved classroom characteristics that affect student's outcomes. These unobserved variables lead to correlation between outcomes for students from the same classroom. Similarly, the unobserved time residuals lead to correlation between a student's outcomes over time. A three-level model can be described as follows:

$$y_{cst} = Z^1_{cst}\beta^1_{cs} + X^1_{cst}\beta_1 + \epsilon^1_{cst}$$

$$\beta^1_{cs} = Z^2_{cs}\beta^2_c + X^2_{cs}\beta_2 + \epsilon^2_{cs}$$

$$\beta^2_c = X^3_c\beta_3 + \epsilon^3_c$$

.

Like in single-level regression, in multilevel models endogeneity is also a concern. The additional problem is that in multilevel models there are multiple independent assumptions involving various random components at different levels. Any moderate correlation between some predictors and a random component or error term, can result in a significant bias of the coefficients and of the variance components. The multilevel GMM approach for addressing endogeneity uses both the between and within variations of the exogenous variables, but only the within variation of the variables assumed endogenous. The assumptions in the multilevel generalized moment of moments model is that the errors at each level are normally distributed and independent of each other. Moreover, the slope variables are assumed exogenous. Since the model does not handle "level 1 dependencies", an additional assumption is that the level 1 structural error is uncorrelated with any of the regressors. If this assumption is not met, additional, external instruments are necessary. The coefficients of the explanatory variables appear in the vectors β_1, β_2 and β_3. The term β^1_{cs} captures latent, unobserved characteristics that are classroom and student specific while β^2_c captures latent, unobserved characteristics that are classroom specific. For identification, the disturbance term ϵ_{cst} is assumed independent of the other variables, Z^1_{cst} and X^1_{cst}. When all model variables are assumed exogenous, the GMM estimator is the usual GLS estimator, denoted as REF. When all variables (except the variables used as slope) are assumed endogenous, the fixed-effects estimator is used, FE. While REF assumes all explanatory variables are uncorrelated with the random intercepts and slopes in the model, FE allows for endogeneity of all effects but sweeps out the random components as well as the explanatory variables at the same levels. The more general estimator GMM proposed by Kim and Frees (2007) allows for some of the explanatory variables to be endogenous and uses this information to build instrumental variables. The multilevel GMM estimator uses both the between and within variations of the exogenous variables, but only the within variation of the variables assumed endogenous. When all variables are assumed exogenous, GMM estimator equals REF. When all covariates are assume endogenous, GMM equals FE.

Formula parameter:

The formula argument follows a two part notation:

In the first part, the model is specified while in the second part, the endogenous regressors are indicated. These two parts are separated by a single vertical bar (|).

The first RHS follows the exact same model specification as required by the lmer function of package lme4 and internally will be used to fit a lmer model. In the second part, one or multiple endogenous regressors are indicated by passing them to the special function endo (e.g. endo(X1,X2)). Note that no argument to endo() is to be supplied as character but as symbols without quotation marks.

See the example section for illustrations on how to specify the formula parameter.

Value

multilevelIV returns an object of class "rendo.multilevel".

The generic accessor functions coef, fitted, residuals, vcov, confint, and nobs, are available. Note that an additional argument model with possible values "REF","FE_L2","FE_L3","GMM_L2", or "GMM_L3" is available for summary, fitted, residuals, confint, and vcov to extract the features for the specified model.

Note that the obtained coefficients are rounded with round(x,digits=getOption("digits")).

An object of class rendo.multilevel is returned that is a list and contains the following components:

formula	the formula given to specify the model to be fitted.
num.levels	the number of levels detected from the model.
dt.model.data	a data.table of model data including data for slopes and level group ids
coefficients	a matrix of rounded coefficients, one column per model.
coefficients.se	a matrix of coefficients' SE, one column per model.
l.fitted	a named list which contains the fitted values per model sorted as the input data
l.residuals	a named list which contains the residuals per model sorted as the input data
l.vcov	a list of variance-covariance matrix, named per model.
V	the variance–covariance matrix V of the disturbance term.
W	the weight matrix W, such that W=V^(-1/2) per highest level group.
l.ovt	a list of results of the Hausman OVT, named per model.

References

Hausman J (1978). "Specification Tests in Econometrics." Econometrica, 46(6), 1251–1271.

Kim, Jee-Seon and Frees, Edward W. (2007). "Multilevel Modeling with Correlated Effects". Psychometrika, 72(4), 505-533.

See Also

lmer for more details on how to specify the formula parameter

lmerControl for more details on how to provide the lmer.control parameter

summary for how fitted models are summarized

Examples

```
data("dataMultilevelIV")

# Two levels
res.ml.L2 <- multilevelIV(y ~ X11 + X12 + X13 + X14 + X15 + X21 + X22 + X23 + X24 + X31 +
                              X32 + X33 + (1|SID) | endo(X15),
                          data = dataMultilevelIV, verbose = FALSE)

# Three levels
res.ml.L3 <- multilevelIV(y ~ X11 + X12 + X13 + X14 + X15 + X21 + X22 + X23 + X24 + X31 +
                              X32 + X33 + (1| CID) + (1|SID) | endo(X15),
                          data = dataMultilevelIV, verbose = FALSE)


# L2 with multiple endogenous regressors
res.ml.L2 <- multilevelIV(y ~ X11 + X12 + X13 + X14 + X15 + X21 + X22 + X23 + X24 + X31 +
                              X32 + X33 + (1|SID) | endo(X15, X21, X22),
                          data = dataMultilevelIV, verbose = FALSE)

# same as above
res.ml.L2 <- multilevelIV(y ~ X11 + X12 + X13 + X14 + X15 + X21 + X22 + X23 + X24 + X31 +
                              X32 + X33 + (1|SID) | endo(X15, X21) + endo(X22),
                          data = dataMultilevelIV, verbose = FALSE)

# Fit above model with different settings for lmer()
lmer.control <- lme4::lmerControl(optimizer="nloptwrap",
                                  optCtrl=list(algorithm="NLOPT_LN_COBYLA",
                                               xtol_rel=1e-6))
res.ml.L2.cob <- multilevelIV(y ~ X11 + X12 + X13 + X14 + X15 + X21 + X22 + X23 + X24 +
                                 X31 + X32 + X33 + (1|SID) | endo(X15, X21) + endo(X22),
                              data = dataMultilevelIV, verbose = FALSE,
                             lmer.control = lmer.control) # use different controls for lmer


# specify argument "model" in the S3 methods to obtain results for the respective model
# default is "REF" for all methods

summary(res.ml.L3)
# same as above
summary(res.ml.L3, model = "REF")

# complete pval table for L3 fixed effects
L3.FE.p <- coef(summary(res.ml.L3, model = "FE_L3"))

# variance covariance matrix
L2.FE.var  <- vcov(res.ml.L2, model = "FE_L2")
L2.GMM.var <- vcov(res.ml.L2, model = "GMM_L2")
# residuals
L3.REF.resid <- resid(res.ml.L3, model = "REF")
```

`predict.rendo.copula.correction` *Predict method for Models using the Gaussian Copula Approach*

Description

Predicted values based on linear models with endogenous regressors estimated using the gaussian copula.

Usage

```
## S3 method for class 'rendo.copula.correction'
predict(object, newdata, ...)
```

Arguments

`object`	Object of class inheriting from "rendo.copula.correction"
`newdata`	An optional data frame in which to look for variables with which to predict. If omitted, the fitted values are returned.
`...`	ignored, for consistency with the generic function.

Value

`predict.copula.correction` produces a vector of predictions

See Also

The model fitting function `copulaCorrection`

Examples

```
## Not run:
data("dataCopCont")

c1 <- copulaCorrection(y~X1+X2+P|continuous(P), num.boots=10,
                       data=dataCopCont)

# returns the fitted values
predict(c1)

# using the data used for fitting also for predicting,
# correctly results in fitted values
all.equal(predict(c1, dataCopCont), fitted(c1)) # TRUE

## End(Not run)
```

`predict.rendo.ivreg` *Predict method for fitted Regression Models with Internal Instrumental Variables*

Description

Predicted values based on model objects fitted using the instrumental variables regression fitted with IVs generated from the data.

Usage

```
## S3 method for class 'rendo.ivreg'
predict(object, newdata, ...)
```

Arguments

`object`	Object of class inheriting from "rendo.ivreg"
`newdata`	An optional data frame without any instrumental variables in which to look for variables with which to predict. If omitted, the fitted values are returned.
`...`	ignored, for consistency with the generic function.

Value

`predict.rendo.ivreg` produces a vector of predictions

See Also

The model fitting functions `hetErrorsIV`, `higherMomentsIV`.

Examples

```
data("dataHetIV")

het <- hetErrorsIV(y~X1+X2+P|P|IIV(X1, X2),
                   data = dataHetIV)

# returns the fitted values
predict(het)

# using the data used for fitting also for predicting,
# correctly results in fitted values
all.equal(predict(het, dataHetIV), fitted(het)) # TRUE
```

predict.rendo.latent.IV *Predict method for Models using the Latent Instrumental Variables approach*

Description

Predicted values based on linear models estimated using the latent instrumental variables approach for a single endogenous regressor.

Usage

```
## S3 method for class 'rendo.latent.IV'
predict(object, newdata, ...)
```

Arguments

object	Object of class inheriting from "rendo.latent.IV"
newdata	An optional data frame in which to look for variables with which to predict. If omitted, the fitted values are returned.
...	ignored, for consistency with the generic function.

Value

predict.rendo.latent.IV produces a vector of predictions

See Also

The model fitting function latentIV

Examples

```
data("dataLatentIV")

lat  <- latentIV(y ~ P, data = dataLatentIV)

# returns the fitted values
predict(lat)

# using the data used for fitting also for predicting,
# correctly results in fitted values
all.equal(predict(lat, dataLatentIV), fitted(lat)) # TRUE
```

predict.rendo.multilevel *Predict method for Multilevel GMM Estimations*

Description

Predicted values based on multilevel models employing the GMM approach for hierarchical data with endogenous regressors.

Usage

```
## S3 method for class 'rendo.multilevel'
predict(object, newdata, model = c("REF",
  "FE_L2", "FE_L3", "GMM_L2", "GMM_L3"), ...)
```

Arguments

object	Object of class inheriting from "rendo.multilevel"
newdata	An optional data frame in which to look for variables with which to predict. If omitted, the fitted values for the specified model are returned.
model	character string to indicate for which fitted model predictions are made. Possible values are: "REF","FE_L2","FE_L3","GMM_L2", or "GMM_L3".
...	ignored, for consistency with the generic function.

Value

predict.rendo.multilevel produces a vector of predictions

See Also

The model fitting function multilevelIV

Examples

```
data("dataMultilevelIV")

# Two levels
res.ml.L2 <- multilevelIV(y ~ X11 + X12 + X13 + X14 + X15 + X21 + X22 + X23 + X24 + X31 +
                          X32 + X33 + (1|SID) | endo(X15),
                          data = dataMultilevelIV, verbose = FALSE)
predict(res.ml.L2, model = "FE_L2")

# using the data used for fitting also for predicting,
#    correctly results in fitted values
all.equal(predict(res.ml.L2, dataMultilevelIV, model = "GMM_L2"),
          fitted(res.ml.L2, model = "GMM_L2")) # TRUE
```

REndo *Fitting Linear Models with Endogenous Regressors using Latent Instrumental Variables*

Description

Fits linear models with endogenous regressor using latent instrumental variable approaches.

The methods included in the package are Lewbel's (1997) <doi:10.2307/2171884> higher moments approach as well as Lewbel's (2012) <doi:10.1080/07350015.2012.643126> heteroskedasticity approach, Park and Gupta's (2012) <doi:10.1287/mksc.1120.0718> joint estimation method that uses Gaussian copula and Kim and Frees's (2007) <doi:10.1007/s11336-007-9008-1> multilevel generalized method of moment approach that deals with endogeneity in a multilevel setting. These are statistical techniques to address the endogeneity problem where no external instrumental variables are needed.

The main functions to estimate models are:

- `latentIV()`the latent instrumental variables method of Ebbes et al. (2005)
- `copulaCorrection()`copula correction method proposed by Paek and Gupta (2012)
- `hetErrorsIV()`heteroskedastic errors approach proposed by Lewbel(2012)
- `higherMomentsIV()`higher moments method proposed by Lewbel (1997)
- `multilevelIV()`multilevel GMM method proposed by Kim and Frees (2007)

Differences between current (2.0.0) and previous version of REndo

Note that with version 2.0.0 sweeping changes were which greatly improve functionality but break backwards compatibility. Various bugs were fixed, performance improved, handling of S3 objects and methods across the package was harmonized, and a set of argument checks has been added. Starting with REndo 2.0, all functions support the use of transformations such as I(x^2) or log(x) in the formulas. Moreover, the call of most of the functions (except latentIV() and multilevelIV()) changed from the previous versions, making use of the Formula package.

Check the NEWS file or our github page for the latest updates and for reporting issues.

summary.rendo.copula.correction *Summarizing Bootstrapped copulaCorrection Model Fits*

Description

`summary` method for a model of class `rendo.copula.correction` resulting from fitting `copulaCorrection`.

Usage

```
## S3 method for class 'rendo.copula.correction'
summary(object, ...)
```

Arguments

object	an object of class rendo.copula.correction, a result of a call to copulaCorrection.
...	ignored, for consistency with the generic function.

Details

For a single continuous endogenous regressor, the estimation is realized in two steps by first obtaining the empirical distribution of the endogenous regressor and then the likelihood function is built. Also for all other cases the estimation is realized in two steps and hence the standard errors reported by the fitted OLS model are not correct.

The standard errors and the confidence intervals are therefore obtained using bootstrapping with replacement as described in Effron (1979). The reported lower and upper boundaries are from the 95% bootstrapped percentile confidence interval. If there are too few bootstrapped estimates, no boundaries are reported.

For a single continuous endogenous regressor the model was fitted using maximum likelihood optimization. The related goodness of fit measures and convergence indicators are also reported here.

Value

The function computes and returns a list of summary statistics which contains the following components:

coefficients	a px4 matrix with columns for the estimated coefficients for the the original data, the standard error derived from the bootstrapped parameters, and the lower and upper boundaries of the 95% bootstrap confidence interval.
num.boots	the number of bootstraps performed.
names.main.coefs	a vector specifying which coefficients are from the model. For internal usage.
start.params	a named vector with the initial set of parameters used to optimize the log-likelihood function.
vcov	variance covariance matrix derived from the bootstrapped parameters.
names.vars.continuous	the names of the continuous endogenous regressors.
names.vars.discrete	the names of the discrete endogenous regressors.

For the case of a single continuous endogenous regressor, also the following components resulting from the log-likelihood optimization are returned:

AIC	Akaike's An Information Criterion for the model fitted on the provided data.
BIC	Schwarz's Bayesian Criterion for the model fitted on the provided data.
KKT1	first Kuhn, Karush, Tucker optimality condition as returned by optimx.
KKT2	second Kuhn, Karush, Tucker optimality condition as returned by optimx.
conv.code	the convergence code as returned by optimx.
log.likelihood	the value of the log-likelihood function at the found solution for the provided data.

References

Effron, B.(1979). "Bootstrap Methods: Another Look at the Jackknife", The Annals of Statistics, 7(1), 1-26.

See Also

The model fitting function `copulaCorrection`

`confint` for how the confidence intervals are derived

`vcov` for how the variance-covariance matrix is derived

`optimx` for explanations about the returned `conv.code` and `KKT`.

Function `coef` will extract the `coefficients` matrix and function `vcov` will extract the component `vcov` from the returned summary object.

`summary.rendo.latent.IV`

Summarizing latentIV Model Fits

Description

`summary` method for a model of class `rendo.latent.IV` resulting from fitting `latentIV`

Usage

```
## S3 method for class 'rendo.latent.IV'
summary(object, ...)
```

Arguments

`object`	an object of class `rendo.latent.IV`, a result of a call to `latentIV`.
`...`	ignored, for consistency with the generic function.

Value

The function `summary.rendo.latent.IV` computes and returns a list of summary statistics which contains the following components:

`coefficients`	a `px4` matrix with columns for the estimated coefficients, its standard error, the t-statistic and corresponding (two-sided) p-value.
`start.params`	a named vector with the initial set of parameters used to optimize the log-likelihood function.
`names.main.coefs`	a vector specifying which coefficients are from the model. For internal usage.
`vcov`	variance covariance matrix derived from the hessian.
`AIC`	Akaike's An Information Criterion for the model fitted on the provided data.

BIC	Schwarz's Bayesian Criterion for the model fitted on the provided data.
KKT1	first Kuhn, Karush, Tucker optimality condition as returned by optimx.
KKT2	second Kuhn, Karush, Tucker optimality condition as returned by optimx.
conv.code	the convergence code as returned by optimx.
log.likelihood	the value of the log-likelihood function at the found solution for the provided data.

See Also

The model fitting function latentIV

Function coef will extract the coefficients matrix and function vcov will extract the component vcov from the returned summary object.

---

summary.rendo.multilevel

Summarizing Multilevel GMM Estimation with Endogenous Regressors Model Fits

---

Description

summary method for class "rendo.multilevel".

Usage

```
## S3 method for class 'rendo.multilevel'
summary(object, model = c("REF", "FE_L2",
  "FE_L3", "GMM_L2", "GMM_L3"), ...)
```

Arguments

object	an object of class "rendo.multilevel", usually, a result of a call to multilevelIV.
model	character string to indicate which fitted model should be summarized. Possible values are: "REF","FE_L2","FE_L3","GMM_L2", or "GMM_L3".
...	ignored, for consistency with the generic function.

Details

The multilevelIV() function estimates three models, namely: the usual random effects model (REF), the fixed effects model (FE) and the hierarchical GMM model (GMM) proposed by Kim and Frees (2007). The fixed effects and the GMM estimators are calculated at each level - so in the case of a three-level model, the function estimates, besides the random effects, fixed effects models at level two (FE_L2) and at level three (FE_L3). The same is true for the GMM estimators, the multilevelIV() function will return a GMM estimator at level-three (GMM_L3) and a GMM estimator at level two (GMM_L2).

In order to facilitate the choice of estimator to be used, the summary() function also returns an omitted variable test (OVT). This test is based on the Hausman test for panel data. The OVT allows

the comparison of a robust eastimator and an estimator which is efficient under the null hypothesis of no omitted variables. Moreover, it allows the comparison of two robust estimators at different levels.

For the model specified in argument `model`, the `summary()` function returns the summary statistics of the estimated coefficients, together with the results of the omitted variable test between the specified model and each other model.

Value

For the model specified in argument `model`, the function `summary.rendo.multilevel` computes and returns a list of summary statistics and the results of the omitted variable tests for the fitted multilevel object given in `object`.

An object of class `summary.rendo.multilevel` is returned that is a list using the component `call` of argument `object`, plus,

`summary.model`	the model parameter with which the summary function was called.
`coefficients`	a `px4` matrix with columns for the estimated coefficients, its standard error, the t-statistic and corresponding (two-sided) p-value.
`OVT.table`	results of the Hausman omitted variable test for the specified model compared to all other models.
`vcov`	variance covariance matrix derived from the GMM fit of this model.

See Also

The model fitting function `multilevelIV`

Function `coef` will extract the `coefficients` matrix and function `vcov` will extract the component `vcov`.

Examples

```
data("dataMultilevelIV")
# Fit two levels model
res.ml.L2 <- multilevelIV(y ~ X11 + X12 + X13 + X14 + X15 + X21 + X22 + X23 + X24 + X31 +
                            X32 + X33 + (1|SID) | endo(X15),
                          data = dataMultilevelIV, verbose = FALSE)

# Get summary for FE_L2 (does not print)
res.sum <- summary(res.ml.L2, model = "FE_L2")
# extract table with coefficients summary statistics
sum.stat.FE_L2 <- coef(res.sum)
# extract vcov of model FE_L2
FE_L2.vcov <- vcov(res.sum)
# same as above
FE_L2.vcov <- vcov(res.ml.L2, model = "FE_L2")
```

`vcov.rendo.boots` *Calculate Variance-Covariance Matrix for Models Fitted with Bootstrapped Parameters*

Description

The variance-covariance matrix is derived from the bootstrapped parameter estimates stored in the object. It is based on Efron (1979) and calculates the result as follows:

$$\frac{1}{B-1}\sum_{b=1}^{B}(\theta_b - \bar{\theta})(\theta_b - \bar{\theta})$$

where B is the number of bootstraps and $\bar{\theta}$ is the mean of the bootstrapped coefficients.

Usage

```
## S3 method for class 'rendo.boots'
vcov(object, ...)
```

Arguments

`object`	a fitted model object with bootstrapped parameters. Typically from `copulaCorrection`
`...`	ignored, for consistency with the generic function.

Value

A matrix of the estimated covariances between the parameter estimates of the model. The row and column names correspond to the parameter names given by the `coef` method.

References

Effron, B.(1979). "Bootstrap Methods: Another Look at the Jackknife", The Annals of Statistics, 7(1), 1-26.

Index