Package ‘endogeneity’

August 21, 2023

Type Package
Title Recursive Two-Stage Models to Address Endogeneity
Version 2.1.3
Date 2023-08-20
Author Jing Peng
Maintainer Jing Peng <jing.peng@uconn.edu>
Description Various recursive two-stage models to address the endogeneity issue of treatment variables in observational study or mediators in experiments. The details of the models are discussed in Peng (2023) <doi:10.1287/isre.2022.1113>.
License GPL (>= 3)
Encoding UTF-8
VignetteBuilder knitr
Imports Rcpp, pbivnorm, maxLik, statmod, MASS, data.table
LinkingTo Rcpp, RcppArmadillo
Suggests knitr, rmarkdown
RoxygenNote 7.2.3
NeedsCompilation yes
Repository CRAN
Date/Publication 2023-08-21 04:02:35 UTC

R topics documented:

bilinear ................................................................. 2
biprobit ................................................................ 4
biprobit_latent ...................................................... 6
biprobit_partial ................................................... 9
endogeneity ............................................................. 11
linear_probit .......................................................... 13
pln ................................................................. 15
pln_linear ............................................................. 17
Recusrive Bivariate Linear Model

Description

Estimate two linear models with bivariate normally distributed error terms.

First stage (Linear):
\[ m_i = \alpha'w_i + \lambda u_i \]

Second stage (Linear):
\[ y_i = \beta'x_i + \gamma m_i + \sigma v_i \]

Endogeneity structure: \( u_i \) and \( v_i \) are bivariate normally distributed with a correlation of \( \rho \).

The identification of this model requires an instrumental variable that appears in \( w \) but not \( x \). This model still works if the first-stage dependent variable is not a regressor in the second stage.

Usage

\[ \text{bilinear}(\text{form1, form2, data = NULL, par = NULL, method = "BFGS", verbose = 0}) \]

Arguments

form1 Formula for the first linear model
form2 Formula for the second linear model
data Input data, a data frame
par Starting values for estimates
method Optimization algorithm. Default is BFGS
verbose A integer indicating how much output to display during the estimation process.
    • <0 - No output
    • 0 - Basic output (model estimates)
    • 1 - Moderate output, basic output + parameter and likelihood in each iteration
    • 2 - Extensive output, moderate output + gradient values on each call
Value

A list containing the results of the estimated model, some of which are inherited from the return of maxLik

- estimates: Model estimates with 95% confidence intervals. Prefix "1" means first stage variables.
- estimate or par: Point estimates
- variance_type: covariance matrix used to calculate standard errors. Either BHHH or Hessian.
- var: covariance matrix
- se: standard errors
- var_bhhh: BHHH covariance matrix, inverse of the outer product of gradient at the maximum
- se_bhhh: BHHH standard errors
- gradient: Gradient function at maximum
- hessian: Hessian matrix at maximum
- gtHg: \( g'H^{-1}g \), where \( H^{-1} \) is simply the covariance matrix. A value close to zero (e.g., <1e-3 or 1e-6) indicates good convergence.
- LL or maximum: Likelihood
- AIC: AIC
- BIC: BIC
- n_obs: Number of observations
- n_par: Number of parameters
- LR_stat: Likelihood ratio test statistic for \( \rho = 0 \)
- LR_p: p-value of likelihood ratio test
- iterations: number of iterations taken to converge
- message: Message regarding convergence status.

Note that the list inherits all the components in the output of maxLik. See the documentation of maxLik for more details.

References


See Also

Other endogeneity: biprobit_latent(), biprobit_partial(), biprobit(), linear_probit(), pln_linear(), pln_probit(), probit_linearRE(), probit_linear_latent(), probit_linear_partial(), probit_linear()
Examples

```r
library(MASS)
N = 2000
rho = -0.5
set.seed(1)
x = rbinom(N, 1, 0.5)
z = rnorm(N)
e = mvrnorm(N, mu=c(0,0), Sigma=matrix(c(1,rho,rho,1), nrow=2))
e1 = e[,1]
e2 = e[,2]
m = -1 + x + z + e1
y = -1 + x + m + e2
est = bilinear(m~x+z, y~x+m)
print(est$estimates, digits=3)
```

biprobit

Recursive Bivariate Probit Model

Description

Estimate two probit models with bivariate normally distributed error terms.

First stage (Probit):

\[ m_i = 1(\alpha'w_i + u_i > 0) \]

Second stage (Probit):

\[ y_i = 1(\beta'x_i + \gamma m_i + \sigma v_i > 0) \]

Endogeneity structure: \( u_i \) and \( v_i \) are bivariate normally distributed with a correlation of \( \rho \).

\( w \) and \( x \) can be the same set of variables. Identification can be weak if \( w \) are not good predictors of \( m \). This model still works if the first-stage dependent variable is not a regressor in the second stage.

Usage

```r
biprobit(form1, form2, data = NULL, par = NULL, method = "BFGS", verbose = 0)
```

Arguments

- `form1`: Formula for the first probit model
- `form2`: Formula for the second probit model
- `data`: Input data, a data frame
- `par`: Starting values for estimates
method Optimization algorithm. Default is BFGS
verbose A integer indicating how much output to display during the estimation process.
  • <0 - No output
  • 0 - Basic output (model estimates)
  • 1 - Moderate output, basic output + parameter and likelihood in each iteration
  • 2 - Extensive output, moderate output + gradient values on each call

Value
A list containing the results of the estimated model, some of which are inherited from the return of maxLik
  • estimates: Model estimates with 95% confidence intervals. Prefix "1" means first stage variables.
  • estimate or par: Point estimates
  • variance_type: covariance matrix used to calculate standard errors. Either BHHH or Hessian.
  • var: covariance matrix
  • se: standard errors
  • var_bhhh: BHHH covariance matrix, inverse of the outer product of gradient at the maximum
  • se_bhhh: BHHH standard errors
  • gradient: Gradient function at maximum
  • hessian: Hessian matrix at maximum
  • gtHg: $g'H^{-1}g$, where H^-1 is simply the covariance matrix. A value close to zero (e.g., <1e-3 or 1e-6) indicates good convergence.
  • LL or maximum: Likelihood
  • AIC: AIC
  • BIC: BIC
  • n_obs: Number of observations
  • n_par: Number of parameters
  • LR_stat: Likelihood ratio test statistic for $\rho = 0$
  • LR_p: p-value of likelihood ratio test
  • iterations: number of iterations taken to converge
  • message: Message regarding convergence status.

Note that the list inherits all the components in the output of maxLik. See the documentation of maxLik for more details.

References
See Also

Other endogeneity: bilinear(), biprobit_latent(), biprobit_partial(), linear_probit(), pln_linear(), pln_probit(), probit_linearRE(), probit_linear_latent(), probit_linear_partial(), probit_linear()

Examples

```r
library(MASS)
N = 2000
rho = -0.5
set.seed(1)

x = rbinom(N, 1, 0.5)
z = rnorm(N)

e = mvrnorm(N, mu=c(0,0), Sigma=matrix(c(1,rho,rho,1), nrow=2))
e1 = e[,1]
e2 = e[,2]

m = as.numeric(1 + x + z + e1 > 0)
y = as.numeric(1 + x + z + m + e2 > 0)
est = biprobit(m~x+z, y~x+z+m)
print(est$estimates, digits=3)
```

---

### biprobit_latent

**Recursive Bivariate Probit Model with Latent First Stage**

#### Description

Estimate two probit models with bivariate normally distributed error terms, in which the dependent variable of the first stage model is unobserved.

First stage (Probit, \( m^*_i \) is unobserved):

\[
m^*_i = 1(\alpha'w_i + u_i > 0)
\]

Second stage (Probit):

\[
y_i = 1(\beta'x_i + \gamma m^*_i + \sigma v_i > 0)
\]

Endogeneity structure: \( u_i \) and \( v_i \) are bivariate normally distributed with a correlation of \( \rho \).

\( w \) and \( x \) can be the same set of variables. The identification of this model is generally weak, especially if \( w \) are not good predictors of \( m \). \( \gamma \) is assumed to be positive to ensure that the model estimates are unique.
Usage

biprobitlatent(
  form1,
  form2,
  data = NULL,
  EM = FALSE,
  par = NULL,
  method = "BFGS",
  verbose = 0,
  maxIter = 500,
  tol = 1e-05,
  tol_LL = 1e-06
)

Arguments

form1 Formula for the first probit model, in which the dependent variable is unobserved. Use a formula like ~w to avoid specifying the dependent variable.
form2 Formula for the second probit model, the latent dependent variable of the first stage is automatically added as a regressor in this model.
data Input data, a data frame.
EM Whether to maximize likelihood use the Expectation-Maximization (EM) algorithm, which is slower but more robust. Defaults to FLASE, but should change to TRUE is the model has convergence issues.
par Starting values for estimates.
method Optimization algorithm. Default is BFGS.
verbose A integer indicating how much output to display during the estimation process.
  • <0 - No output
  • 0 - Basic output (model estimates)
  • 1 - Moderate output, basic output + parameter and likelihood in each iteration
  • 2 - Extensive output, moderate output + gradient values on each call
maxIter max iterations for EM algorithm.
tol tolerance for convergence of EM algorithm.
tol_LL tolerance for convergence of likelihood.

Value

A list containing the results of the estimated model, some of which are inherited from the return of maxLik.

• estimates: Model estimates with 95% confidence intervals. Prefix "1" means first stage variables.
• estimate or par: Point estimates.
• variance_type: covariance matrix used to calculate standard errors. Either BHHH or Hessian.
• var: covariance matrix
• se: standard errors
• gradient: Gradient function at maximum
• hessian: Hessian matrix at maximum
• $g' H^{-1} g$, where $H^{-1}$ is simply the covariance matrix. A value close to zero (e.g., <1e-3 or 1e-6) indicates good convergence.
• LL or maximum: Likelihood
• AIC: AIC
• BIC: BIC
• n_obs: Number of observations
• n_par: Number of parameters
• iterations: number of iterations taken to converge
• message: Message regarding convergence status.

Note that the list inherits all the components in the output of maxLik. See the documentation of maxLik for more details.

References


See Also

Other endogeneity: bilinear(), biprobit_partial(), biprobit(), linear_probit(), pln_linear(), pln_probit(), probit_linearRE(), probit_linear_latent(), probit_linear_partial(), probit_linear()

Examples

library(MASS)
N = 2000
rho = -0.5
set.seed(1)
x = rbinom(N, 1, 0.5)
z = rnorm(N)
e = mvrnorm(N, mu=c(0,0), Sigma=matrix(c(1,rho,rho,1), nrow=2))
e1 = e[,1]
e2 = e[,2]
m = as.numeric(1 + x + z + e1 > 0)
y = as.numeric(1 + x + z + m + e2 > 0)
est = biprobit(m~x+z, y~x+z+m)
print(est$estimates, digits=3)
est_latent = biprobit_latent(~x+z, y~x+z)
print(est_latent$estimates, digits=3)

biprobit_partial  Recursive Bivariate Probit Model with Partially Observed First Stage

Description

Estimate two probit models with bivariate normally distributed error terms, in which the dependent variable of the first stage model is partially observed (or unobserved).

First stage (Probit, \( m_i \) is partially observed):

\[
m_i = \begin{cases} 1 & (\alpha'w_i + u_i > 0) \\ 0 & \text{otherwise} \end{cases}
\]

Second stage (Probit):

\[
y_i = 1(\beta'x_i + \gamma m_i + \sigma v_i > 0)
\]

Endogeneity structure: \( u_i \) and \( v_i \) are bivariate normally distributed with a correlation of \( \rho \).

Unobserved \( m_i \) should be coded as NA. \( w \) and \( x \) can be the same set of variables. Identification can be weak if \( w \) are not good predictors of \( m \). Observing \( m_i \) for 10%~20% of observations can significantly improve the identification of the model.

Usage

biprobit_partial(
  form1,
  form2,
  data = NULL,
  EM = FALSE,
  par = NULL,
  method = "BFGS",
  verbose = 0,
  maxIter = 500,
  tol = 1e-05,
  tol_LL = 1e-06
)

Arguments

form1  Formula for the first probit model, in which the dependent variable is partially observed.

form2  Formula for the second probit model, the partially observed dependent variable of the first stage is automatically added as a regressor in this model (do not add manually)
**biprobit_partial**

**data**
Input data, a data frame

**EM**
Whether to maximize likelihood use the Expectation-Maximization (EM) algorithm, which is slower but more robust. Defaults to FALSE, but should change to TRUE is the model has convergence issues.

**par**
Starting values for estimates

**method**
Optimization algorithm. Default is BFGS

**verbose**
A integer indicating how much output to display during the estimation process.
- <0 - No output
- 0 - Basic output (model estimates)
- 1 - Moderate output, basic output + parameter and likelihood in each iteration
- 2 - Extensive output, moderate output + gradient values on each call

**maxIter**
max iterations for EM algorithm

**tol**
tolerance for convergence of EM algorithm

**tol_LL**
tolerance for convergence of likelihood

**Value**
A list containing the results of the estimated model, some of which are inherited from the return of maxLik

- estimates: Model estimates with 95% confidence intervals. Prefix "1" means first stage variables.
- estimate or par: Point estimates
- variance_type: covariance matrix used to calculate standard errors. Either BHHH or Hessian.
- var: covariance matrix
- se: standard errors
- gradient: Gradient function at maximum
- hessian: Hessian matrix at maximum
- gtHg: $g'H^{-1}g$, where $H^{-1}$ is simply the covariance matrix. A value close to zero (e.g., <1e-3 or 1e-6) indicates good convergence.
- LL or maximum: Likelihood
- AIC: AIC
- BIC: BIC
- n_obs: Number of observations
- n_par: Number of parameters
- iterations: number of iterations taken to converge
- message: Message regarding convergence status.

Note that the list inherits all the components in the output of maxLik. See the documentation of maxLik for more details.
endogeneity

References


See Also

Other endogeneity: `bilinear()`, `biprobit_latent()`, `biprobit()`, `linear_probit()`, `pln_linear()`, `pln_probit()`, `probit_linearRE()`, `probit_linear_latent()`, `probit_linear_partial()`, `probit_linear()`

Examples

```r
library(MASS)
N = 5000
rho = -0.5
set.seed(1)
x = rbinom(N, 1, 0.5)
z = rnorm(N)
e = mvrnorm(N, mu=c(0,0), Sigma=matrix(c(1,rho,rho,1), nrow=2))
e1 = e[,1]
e2 = e[,2]

m = as.numeric(1 + x + 3*z + e1 > 0)
y = as.numeric(1 + x + z + m + e2 > 0)
est = biprobit(m~x+z, y~x+z+m)
print(est$estimates, digits=3)

# partially observed version of m
observed_pct = 0.2
m_p = m
m_p[sample(N, N*(1-observed_pct))] = NA
est_partial = biprobit_partial(m_p~x+z, y~x+z)
print(est_partial$estimates, digits=3)
```

endogeneity

Recursive two-stage models to address endogeneity

Description

This package supports various recursive two-stage models to address the endogeneity issue. The details of the implemented models are discussed in Peng (2022). In a recursive two-stage model, the dependent variable of the first stage is also the endogenous variable of interest in the second stage. The endogeneity is captured by the correlation in the error terms of the two stages.
Recursive two-stage models can be used to address the endogeneity of treatment variables in observational study and the endogeneity of mediators in experiments.

The first-stage supports linear model, probit model, and Poisson lognormal model. The second-stage supports linear and probit models. These models can be used to address the endogeneity of continuous, binary, and count variables. When the endogenous variable is binary, it can be unobserved or partially unobserved, but the identification can be weak.

**Functions**

- bilinear: recursive bivariate linear model
- biprobit: recursive bivariate probit model
- biprobit_latent: recursive bivariate probit model with latent first stage
- biprobit_partial: recursive bivariate probit model with partially observed first stage
- linear-probit: recursive linear-probit model
- probit_linear: recursive probit-linear model
- probit_linear_latent: recursive probit-linear model with latent first stage
- probit_linear_partial: recursive probit-linear model with partially observed first stage
- probit_linearRE: recursive probit-linearRE model in which the second stage is a panel linear model with random effects
- pln: Poisson lognormal (PLN) model
- pln_linear: recursive PLN-linear model
- pln_probit: recursive PLN-probit model

**References**

Recursive Linear-Probit Model

Description

Estimate linear and probit models with bivariate normally distributed error terms.

First stage (Linear):

\[ m_i = \alpha'w_i + \sigma u_i \]

Second stage (Probit):

\[ y_i = 1(\beta'x_i + \gamma m_i + v_i > 0) \]

Endogeneity structure: \( u_i \) and \( v_i \) are bivariate normally distributed with a correlation of \( \rho \).

The identification of this model requires an instrumental variable that appears in \( w \) but not \( x \). This model still works if the first-stage dependent variable is not a regressor in the second stage.

Usage

linear_probit(
  form_linear,
  form_probit,
  data = NULL,
  par = NULL,
  method = "BFGS",
  init = c("zero", "unif", "norm", "default")[4],
  verbose = 0
)

Arguments

- **form_linear**: Formula for the linear model
- **form_probit**: Formula for the probit model
- **data**: Input data, a data frame
- **par**: Starting values for estimates
- **method**: Optimization algorithm. Default is BFGS
- **init**: Initialization method
- **verbose**: A integer indicating how much output to display during the estimation process.
  - <0 - No output
  - 0 - Basic output (model estimates)
  - 1 - Moderate output, basic output + parameter and likelihood in each iteration
  - 2 - Extensive output, moderate output + gradient values on each call

lrm
Value

A list containing the results of the estimated model, some of which are inherited from the return of maxLik

- estimates: Model estimates with 95% confidence intervals
- estimate or par: Point estimates
- variance_type: covariance matrix used to calculate standard errors. Either BHHH or Hessian.
- var: covariance matrix
- se: standard errors
- var_bhhh: BHHH covariance matrix, inverse of the outer product of gradient at the maximum
- se_bhh: BHHH standard errors
- gradient: Gradient function at maximum
- hessian: Hessian matrix at maximum
- gtHg: \( g' H^{-1} g \), where \( H^{-1} \) is simply the covariance matrix. A value close to zero (e.g., \(<1e-3 \) or \( 1e-6 \)) indicates good convergence.
- LL or maximum: Likelihood
- AIC: AIC
- BIC: BIC
- n_obs: Number of observations
- n_par: Number of parameters
- LR_stat: Likelihood ratio test statistic for \( \rho = 0 \)
- LR_p: p-value of likelihood ratio test
- iterations: number of iterations taken to converge
- message: Message regarding convergence status.

Note that the list inherits all the components in the output of maxLik. See the documentation of maxLik for more details.

References


See Also

Other endogeneity: `bilinear()`, `biprobit_latent()`, `biprobit_partial()`, `biprobit()`, `pln_linear()`, `pln_probit()`, `probit_linearRE()`, `probit_linear_latent()`, `probit_linear_partial()`, `probit_linear()`
Examples

library(MASS)
N = 2000
rho = -0.5
set.seed(1)

x = rbinom(N, 1, 0.5)
z = rnorm(N)

e = mvrnorm(N, mu=c(0,0), Sigma=matrix(c(1,rho,rho,1), nrow=2))
e1 = e[,1]
e2 = e[,2]

m = 1 + x + z + e1
y = as.numeric(1 + x + m + e2 > 0)

est = linear_probit(m~x+z, y~x+m)
print(est$estimates, digits=3)

pln

Poisson Lognormal Model

Description

Estimate a Poisson model with a log-normally distributed heterogeneity term, which is also referred to as the Poisson-Normal model.

\[ E[y_i|x_i,u_i] = \exp(\alpha' x_i + \lambda u_i) \]

The estimates of this model are often similar to those of a negative binomial model.

Usage

pln(form, data = NULL, par = NULL, method = "BFGS", init = c("zero", "unif", "norm", "default"))[4], H = 20, verbose = 0)

Arguments

form Formula
data Input data, a data frame
par

Starting values for estimates

method

Optimization algorithm.

init

Initialization method

H

Number of quadrature points

verbose

A integer indicating how much output to display during the estimation process.
- <0 - No output
- 0 - Basic output (model estimates)
- 1 - Moderate output, basic output + parameter and likelihood in each iteration
- 2 - Extensive output, moderate output + gradient values on each call

Value

A list containing the results of the estimated model, some of which are inherited from the return of maxLik

• estimates: Model estimates with 95% confidence intervals
• estimate or par: Point estimates
• variance_type: covariance matrix used to calculate standard errors. Either BHHH or Hessian.
• var: covariance matrix
• se: standard errors
• gradient: Gradient function at maximum
• hessian: Hessian matrix at maximum
• gtHg: \( g' H^{-1} g \), where \( H^{-1} \) is simply the covariance matrix. A value close to zero (e.g., <1e-3 or 1e-6) indicates good convergence.
• LL or maximum: Likelihood
• AIC: AIC
• BIC: BIC
• n_obs: Number of observations
• n_par: Number of parameters
• LR_stat: Likelihood ratio test statistic for the heterogeneity term \( \lambda = 0 \)
• LR_p: p-value of likelihood ratio test
• iterations: number of iterations taken to converge
• message: Message regarding convergence status.

Note that the list inherits all the components in the output of maxLik. See the documentation of maxLik for more details.

References

Examples

```r
library(MASS)
N = 2000
set.seed(1)
# Works well when the variance of the normal term is not overly large
# When the variance is very large, it tends to be underestimated
x = rbinom(N, 1, 0.5)
z = rnorm(N)
y = rpois(N, exp(-1 + x + z + 0.5 * rnorm(N)))
est = pln(y~x+z)
print(est$estimates, digits=3)
```

---

### Description

Estimate a Poisson Lognormal model and a linear model with bivariate normally distributed error/heterogeneity terms.

First stage (Poisson Lognormal):

\[
E[m_i | w_i, u_i] = \exp(\alpha' w_i + \lambda u_i)
\]

Second stage (Linear):

\[
y_i = \beta' x_i + \gamma m_i + \sigma v_i
\]

Endogeneity structure: \( u_i \) and \( v_i \) are bivariate normally distributed with a correlation of \( \rho \).

This model is typically well-identified even if \( w \) and \( x \) are the same set of variables. This model still works if the first-stage dependent variable is not a regressor in the second stage.

### Usage

```r
pln_linear(
  form_pln,
  form_linear,
  data = NULL,
  par = NULL,
  method = "BFGS",
  init = c("zero", "unif", "norm", "default")[4],
  H = 20,
  verbose = 0
)
```
Arguments

- **form_pln**: Formula for the first-stage Poisson lognormal model
- **form_linear**: Formula for the second-stage linear model
- **data**: Input data, a data frame
- **par**: Starting values for estimates
- **method**: Optimization algorithm.
- **init**: Initialization method
- **H**: Number of quadrature points
- **verbose**: A integer indicating how much output to display during the estimation process.
  - <0 - No output
  - 0 - Basic output (model estimates)
  - 1 - Moderate output, basic output + parameter and likelihood in each iteration
  - 2 - Extensive output, moderate output + gradient values on each call

Value

A list containing the results of the estimated model, some of which are inherited from the return of maxLik

- **estimates**: Model estimates with 95% confidence intervals. Prefix "pln" means first stage variables.
- **estimate or par**: Point estimates
- **variance_type**: covariance matrix used to calculate standard errors. Either BHHH or Hessian.
- **var**: covariance matrix
- **se**: standard errors
- **gradient**: Gradient function at maximum
- **hessian**: Hessian matrix at maximum
- **gtHg**: \( g^\prime H^{-1} g \), where \( H^{-1} \) is simply the covariance matrix. A value close to zero (e.g., <1e-3 or 1e-6) indicates good convergence.
- **LL or maximum**: Likelihood
- **AIC**: AIC
- **BIC**: BIC
- **n_obs**: Number of observations
- **n_par**: Number of parameters
- **LR_stat**: Likelihood ratio test statistic for \( \rho = 0 \)
- **LR_p**: p-value of likelihood ratio test
- **iterations**: number of iterations taken to converge
- **message**: Message regarding convergence status.

Note that the list inherits all the components in the output of maxLik. See the documentation of maxLik for more details.
References


See Also

Other endogeneity: bilinear(), biprobit_latent(), biprobit_partial(), biprobit(), linear_probit(), pln_probit(), probit_linearRE(), probit_linear_latent(), probit_linear_partial(), probit_linear()

Examples

library(MASS)
N = 2000
rho = -0.5
set.seed(1)
x = rbinom(N, 1, 0.5)
z = rnorm(N)
e = mvrnorm(N, mu=c(0,0), Sigma=matrix(c(1,rho,rho,1), nrow=2))
e1 = e[,1]
e2 = e[,2]
m = rpois(N, exp(1 + x + z + e1))
y = 1 + x + m + e2
est = pln_linear(m~x+z, y~x+m)
print(est$estimates, digits=3)

---

pln_probit
Recursive PLN-Probit Model

Description

Estimate a Poisson Lognormal model and a Probit model with bivariate normally distributed error/heterogeneity terms.

First stage (Poisson Lognormal):

\[ E[m_i | w_i, u_i] = \exp(\alpha'w_i + \lambda u_i) \]

Second stage (Probit):

\[ y_i = 1(\beta'x_i + \gamma m_i + \sigma v_i > 0) \]

Endogeneity structure: \( u_i \) and \( v_i \) are bivariate normally distributed with a correlation of \( \rho \).

This model is typically well-identified even if \( w \) and \( x \) are the same set of variables. This model still works if the first-stage dependent variable is not a regressor in the second stage.
Usage

```r
pln_probit(
  form_pln,
  form_probit,
  data = NULL,
  par = NULL,
  method = "BFGS",
  init = c("zero", "unif", "norm", "default")[4],
  H = 20,
  verbose = 0
)
```

Arguments

- **form_pln**: Formula for the first-stage Poisson lognormal model
- **form_probit**: Formula for the second-stage probit model
- **data**: Input data, a data frame
- **par**: Starting values for estimates
- **method**: Optimization algorithm. Without gradient, NM is much faster than BFGS
- **init**: Initialization method
- **H**: Number of quadrature points
- **verbose**: A integer indicating how much output to display during the estimation process.
  - `<0` - No ouput
  - `0` - Basic output (model estimates)
  - `1` - Moderate output, basic ouput + parameter and likelihood in each iteration
  - `2` - Extensive output, moderate output + gradient values on each call

Value

A list containing the results of the estimated model, some of which are inherited from the return of `maxLik`

- **estimates**: Model estimates with 95% confidence intervals. Prefix "pln" means first stage variables.
- **estimate or par**: Point estimates
- **variance_type**: covariance matrix used to calculate standard errors. Either BHHH or Hessian.
- **var**: covariance matrix
- **se**: standard errors
- **gradient**: Gradient function at maximum
- **hessian**: Hessian matrix at maximum
- **gtHg**: $g' H^{-1} g$, where $H^{-1}$ is simply the covariance matrix. A value close to zero (e.g., <1e-3 or 1e-6) indicates good convergence.
- LL or maximum: Likelihood
- AIC: AIC
- BIC: BIC
- n_obs: Number of observations
- n_par: Number of parameters
- LR_stat: Likelihood ratio test statistic for $\rho = 0$
- LR_p: p-value of likelihood ratio test
- iterations: number of iterations taken to converge
- message: Message regarding convergence status.

Note that the list inherits all the components in the output of maxLik. See the documentation of maxLik for more details.

References


See Also

Other endogeneity: bilinear(), biprobit_latent(), biprobit_partial(), biprobit(), linear_probit(), pln_linear(), probit_linearRE(), probit_linear_latent(), probit_linear_partial(), probit_linear()

Examples

```r
library(MASS)
N = 2000
rho = -0.5
set.seed(1)

x = rbinom(N, 1, 0.5)
z = rnorm(N)
e = mvrnorm(N, mu=c(0,0), Sigma=matrix(c(1,rho,rho,1), nrow=2))
e1 = e[,1]
e2 = e[,2]

m = rpois(N, exp(-1 + x + z + e1))
y = as.numeric(1 + x + z + log(1+m) + e2 > 0)

est = pln_probit(m~x+z, y~x+z+log(1+m))
print(est$estimates, digits=3)
```
Recursive Probit-Linear Model

Description

Estimate probit and linear models with bivariate normally distributed error terms.

First stage (Probit):
\[ m_i = 1(\alpha'w_i + u_i > 0) \]

Second stage (Linear):
\[ y_i = \beta'x_i + \gamma m_i + \sigma v_i \]

Endogeneity structure: \(u_i\) and \(v_i\) are bivariate normally distributed with a correlation of \(\rho\).

\(w\) and \(x\) can be the same set of variables. Identification can be weak if \(w\) are not good predictors of \(m\). This model still works if the first-stage dependent variable is not a regressor in the second stage.

Usage

\[
\text{probit_linear}( \\
\text{form_probit,} \\
\text{form_linear,} \\
\text{data = NULL,} \\
\text{par = NULL,} \\
\text{method = "BFGS",} \\
\text{init = c("zero", "unif", "norm", "default")[4],} \quad \text{ verbose = 0} \\
\)
\]

Arguments

- \textbf{form_probit} Formula for the probit model
- \textbf{form_linear} Formula for the linear model
- \textbf{data} Input data, a data frame
- \textbf{par} Starting values for estimates
- \textbf{method} Optimization algorithm. Default is BFGS
- \textbf{init} Initialization method
- \textbf{verbose} A integer indicating how much output to display during the estimation process.
  - \(<0\) - No output
  - \(0\) - Basic output (model estimates)
  - \(1\) - Moderate output, basic output + parameter and likelihood in each iteration
  - \(2\) - Extensive output, moderate output + gradient values on each call
Value

A list containing the results of the estimated model, some of which are inherited from the return of maxLik

- estimates: Model estimates with 95% confidence intervals
- estimate or par: Point estimates
- variance_type: covariance matrix used to calculate standard errors. Either BHHH or Hessian.
- var: covariance matrix
- se: standard errors
- var_bhhh: BHHH covariance matrix, inverse of the outer product of gradient at the maximum
- se_bhhh: BHHH standard errors
- gradient: Gradient function at maximum
- hessian: Hessian matrix at maximum
- gtHg: $g' H^{-1} g$, where $H^{-1}$ is simply the covariance matrix. A value close to zero (e.g., <1e-3 or 1e-6) indicates good convergence.
- LL or maximum: Likelihood
- AIC: AIC
- BIC: BIC
- n_obs: Number of observations
- n_par: Number of parameters
- LR_stat: Likelihood ratio test statistic for $\rho = 0$
- LR_p: p-value of likelihood ratio test
- iterations: number of iterations taken to converge
- message: Message regarding convergence status.

Note that the list inherits all the components in the output of maxLik. See the documentation of maxLik for more details.

References


See Also

Other endogeneity: `bilinear()`, `biprobit_latent()`, `biprobit_partial()`, `biprobit()`, `linear_probit()`, `pln_linear()`, `pln_probit()`, `probit_linearRE()`, `probit_linear_latent()`, `probit_linear_partial()`
Examples

```r
library(MASS)
N = 2000
rho = -0.5
set.seed(1)
x = rbinom(N, 1, 0.5)
z = rnorm(N)
e = mvrnorm(N, mu=c(0,0), Sigma=matrix(c(1,rho,rho,1), nrow=2))
e1 = e[,1]
e2 = e[,2]
m = as.numeric(1 + x + z + e1 > 0)
y = 1 + x + z + m + e2
est = probit_linear(m~x+z, y~x+z+m)
print(est$estimates, digits=3)
```

**probit_linearRE**  
*Recursive Probit-LinearRE Model*

**Description**

A panel extension of the probit_linear model. The first stage is a probit model at the individual level. The second stage is a panel linear model at the individual-time level with individual-level random effects. The random effect is correlated with the error term in the first stage.

First stage (Probit):

\[ m_i = 1(\alpha' w_i + u_i > 0) \]

Second stage (Panel linear model with individual-level random effects):

\[ y_{it} = \beta' x_{it} + \gamma m_i + \lambda v_i + \sigma \epsilon_{it} \]

Endogeneity structure: \( u_i \) and \( v_i \) are bivariate normally distributed with a correlation of \( \rho \).

This model uses Adaptive Gaussian Quadrature to overcome numerical challenges with long panels. \( w \) and \( x \) can be the same set of variables. Identification can be weak if \( w \) are not good predictors of \( m \). This model still works if the first-stage dependent variable is not a regressor in the second stage.

**Usage**

```r
probit_linearRE(
  form_probit,
  form_linear,
  id,
  data = NULL,
  par = NULL,
)```
method = "BFGS",
H = 20,
stopUpdate = F,
init = c("zero", "unif", "norm", "default")[4],
verbose = 0
)

Arguments

form_probit  Formula for the probit model at the individual level
form_linear  Formula for the linear model at the individual-time level
id           group id, character if data supplied or numerical vector if data not supplied
data         Input data, must be a data.table object
par          Starting values for estimates
method       Optimization algorithm. Default is BFGS
H             Number of quadrature points
stopUpdate   Adaptive Gaussian Quadrature disabled if TRUE
init         Initialization method
verbose      A integer indicating how much output to display during the estimation process.
             • <0 - No output
             • 0 - Basic output (model estimates)
             • 1 - Moderate output, basic output + parameter and likelihood in each iteration
             • 2 - Extensive output, moderate output + gradient values on each call

Value

A list containing the results of the estimated model, some of which are inherited from the return of maxLik

• estimates: Model estimates with 95% confidence intervals
• estimate or par: Point estimates
• variance_type: covariance matrix used to calculate standard errors. Either BHHH or Hessian.
• var: covariance matrix
• se: standard errors
• var_bhhh: BHHH covariance matrix, inverse of the outer product of gradient at the maximum
• se_bhhh: BHHH standard errors
• gradient: Gradient function at maximum
• hessian: Hessian matrix at maximum
• g'H^-1g, where H^-1 is simply the covariance matrix. A value close to zero (e.g., <1e-3 or 1e-6) indicates good convergence.
• LL or maximum: Likelihood
• AIC: AIC
• BIC: BIC
• n_obs: Number of observations
• n_par: Number of parameters
• time: Time takes to estimate the model
• LR_stat: Likelihood ratio test statistic for $\rho = 0$
• LR_p: p-value of likelihood ratio test
• iterations: number of iterations taken to converge
• message: Message regarding convergence status.

Note that the list inherits all the components in the output of maxLik. See the documentation of maxLik for more details.

References


See Also

Other endogeneity: bilinear(), biprobit_latent(), biprobit_partial(), biprobit(), linear_probit(), pln_linear(), pln_probit(), probit_linear_latent(), probit_linear_partial(), probit_linear()

Examples

library(MASS)
library(data.table)
N = 500
period = 5
obs = N*period
rho = -0.5
set.seed(100)
e = mvrnorm(N, mu=c(0,0), Sigma=matrix(c(1,rho,rho,1), nrow=2))
e1 = e[,1]
e2 = e[,2]

t = rep(1:period, N)
id = rep(1:N, each=period)
w = rnorm(N)
m = as.numeric(t+w+e1>0)
m_long = rep(m, each=period)

x = rnorm(obs)
y = 1 + x + m_long + rep(e2, each=period) + rnorm(obs)

dt = data.table(y, x, id, t, m=rep(m, each=period), w=rep(w, each=period))
est = probit_linearRE(m~w, y~x+m, 'id', dt)
print(est$estimates, digits=3)
Recursive Probit-Linear Model with Latent First Stage

Description

Latent version of the Probit-Linear Model.

First stage (Probit, \(m_i^\ast\) is unobserved):

\[ m_i^\ast = 1(\alpha' w_i + u_i > 0) \]

Second stage (Linear):

\[ y_i = \beta' x_i + \gamma m_i^\ast + \sigma v_i \]

Endogeneity structure: \(u_i\) and \(v_i\) are bivariate normally distributed with a correlation of \(\rho\).

\(w\) and \(x\) can be the same set of variables. The identification of this model is generally weak, especially if \(w\) are not good predictors of \(m\). \(\gamma\) is assumed to be positive to ensure that the model estimates are unique.

Usage

```r
probit_linear_latent(form_probit, form_linear, data = NULL, EM = TRUE, par = NULL, method = "BFGS", verbose = 0, maxIter = 500, tol = 1e-06, tol_LL = 1e-08)
```

Arguments

- `form_probit`: Formula for the first-stage probit model, in which the dependent variable is latent
- `form_linear`: Formula for the second stage linear model. The latent dependent variable of the first stage is automatically added as a regressor in this model
- `data`: Input data, a data frame
- `EM`: Whether to maximize likelihood use the Expectation-Maximization (EM) algorithm, which is slower but more robust. Defaults to TRUE.
- `par`: Starting values for estimates
- `method`: Optimization algorithm. Default is BFGS
- `verbose`: A integer indicating how much output to display during the estimation process.
probit_linear_latent

- <0 - No output
- 0 - Basic output (model estimates)
- 1 - Moderate output, basic output + parameter and likelihood in each iteration
- 2 - Extensive output, moderate output + gradient values on each call

maxIter  max iterations for EM algorithm
tol      tolerance for convergence of EM algorithm
tol_LL   tolerance for convergence of likelihood

Value

A list containing the results of the estimated model, some of which are inherited from the return of maxLik

- estimates: Model estimates with 95% confidence intervals
- estimate or par: Point estimates
- variance_type: covariance matrix used to calculate standard errors. Either BHHH or Hessian.
- var: covariance matrix
- se: standard errors
- gradient: Gradient function at maximum
- hessian: Hessian matrix at maximum
- gtHg: $g' H^{-1} g$, where $H^{-1}$ is simply the covariance matrix. A value close to zero (e.g., <1e-3 or 1e-6) indicates good convergence.
- LL or maximum: Likelihood
- AIC: AIC
- BIC: BIC
- n_obs: Number of observations
- n_par: Number of parameters
- iter: number of iterations taken to converge
- message: Message regarding convergence status.

Note that the list inherits all the components in the output of maxLik. See the documentation of maxLik for more details.

References


See Also

Other endogeneity: bilinear(), biprobit_latent(), biprobit_partial(), biprobit(), linear_probit(), pln_linear(), pln_probit(), probit_linearRE(), probit_linear_partial(), probit_linear()
Examples

```r
library(MASS)
N = 2000
rho = -0.5
set.seed(1)
x = rbinom(N, 1, 0.5)
z = rnorm(N)
e = mvrnorm(N, mu=c(0,0), Sigma=matrix(c(1,rho,rho,1), nrow=2))
e1 = e[,1]
e2 = e[,2]
m = as.numeric(1 + x + z + e1 > 0)
y = 1 + x + z + m + e2
est = probit_linear(m~x+z, y~x+z+m)
print(est$estimates, digits=3)
est_latent = probit_linear_latent(~x+z, y~x+z)
print(est_latent$estimates, digits=3)
```

Description

Partially observed version of the Probit-Linear Model.

First stage (Probit, \( m_i \) is partially observed):

\[
m_i = I(\alpha'w_i + u_i > 0)
\]

Second stage (Linear):

\[
y_i = \beta'x_i + \gamma m_i + \sigma v_i
\]

Endogeneity structure: \( u_i \) and \( v_i \) are bivariate normally distributed with a correlation of \( \rho \).

Unobserved \( m_i \) should be coded as NA. \( w \) and \( x \) can be the same set of variables. Identification can be weak if \( w \) are not good predictors of \( m \). Observing \( m_i \) for a small proportion of observations (e.g., 10–20%) can significantly improve the identification of the model.

Usage

```r
probit_linear_partial(
  form_probit,
  form_linear,
  data = NULL,
)```
EM = TRUE,
par = NULL,
method = "BFGS",
verbose = 0,
maxIter = 500,
tol = 1e-06,
tol_LL = 1e-08
)

Arguments

form_probit  Formula for the first-stage probit model, in which the dependent variable is partially observed
form_linear Formula for the second stage linear model. The partially observed dependent variable of the first stage is automatically added as a regressor in this model (do not add manually)
data           Input data, a data frame
EM             Whether to maximize likelihood use the Expectation-Maximization (EM) algorithm, which is slower but more robust. Defaults to TRUE.
par            Starting values for estimates
method         Optimization algorithm. Default is BFGS
verbose        A integer indicating how much output to display during the estimation process.
               • <0 - No output
               • 0 - Basic output (model estimates)
               • 1 - Moderate output, basic output + parameter and likelihood in each inter-
               • 2 - Extensive output, moderate output + gradient values on each call
maxIter        max iterations for EM algorithm
tol            tolerance for convergence of EM algorithm
tol_LL         tolerance for convergence of likelihood

Value

A list containing the results of the estimated model, some of which are inherited from the return of
maxLik

• estimates: Model estimates with 95% confidence intervals
• estimate or par: Point estimates
• variance_type: covariance matrix used to calculate standard errors. Either BHHH or Hessian.
• var: covariance matrix
• se: standard errors
• gradient: Gradient function at maximum
• hessian: Hessian matrix at maximum
• $g^H g': g^H H^{-1} g$, where $H^{-1}$ is simply the covariance matrix. A value close to zero (e.g., <1e-3 or 1e-6) indicates good convergence.
• LL or maximum: Likelihood
• AIC: AIC
• BIC: BIC
• n_obs: Number of observations
• n_par: Number of parameters
• iterations: number of iterations taken to converge
• message: Message regarding convergence status.

Note that the list inherits all the components in the output of maxLik. See the documentation of maxLik for more details.

References


See Also

Other endogeneity: bilinear(), biprobit_latent(), biprobit_partial(), biprobit(), linear_probit(), pln_linear(), pln_probit(), probit_linearRE(), probit_linear_latent(), probit_linear()

Examples

library(MASS)
N = 1000
rho = -0.5
set.seed(1)

x = rbinom(N, 1, 0.5)
z = rnorm(N)

e = mvrnorm(N, mu=c(0,0), Sigma=matrix(c(1,rho,rho,1), nrow=2))
e1 = e[,1]
e2 = e[,2]

m = as.numeric(1 + x + z + e1 > 0)
y = 1 + x + z + m + e2

est = probit_linear(m~x+z, y~x+z+m)
print(est$estimates, digits=3)

# partially observed version of m
observed_pct = 0.2
m_p = m
m_p[sample(N, N*(1-observed_pct))] = NA

est_latent = probit_linear_partial(m_p~x+z, y~x+z)
print(est_latent$estimates, digits=3)
# Index

* **endogeneity**
  - bilinear, 2
  - biprobit, 4
  - biprobit_latent, 6
  - biprobit_partial, 9
  - linear_probit, 13
  - pln_linear, 17
  - pln_probit, 19
  - probit_linear, 22
  - probit_linear_latent, 27
  - probit_linear_partial, 29
  - probit_linearRE, 24

  bilinear, 2, 6, 8, 11, 14, 19, 21, 23, 26, 28, 31
  biprobit, 3, 4, 8, 11, 14, 19, 21, 23, 26, 28, 31
  biprobit_latent, 3, 6, 6, 11, 14, 19, 21, 23, 26, 28, 31
  biprobit_partial, 3, 6, 8, 9, 14, 19, 21, 23, 26, 28, 31

  endogeneity, 11

  linear_probit, 3, 6, 8, 11, 13, 19, 21, 23, 26, 28, 31

  pln, 15
  pln_linear, 3, 6, 8, 11, 14, 17, 21, 23, 26, 28, 31
  pln_probit, 3, 6, 8, 11, 14, 19, 23, 26, 28, 31
  probit_linear, 3, 6, 8, 11, 14, 19, 21, 22, 26, 28, 31
  probit_linear_latent, 3, 6, 8, 11, 14, 19, 21, 23, 26, 27, 31
  probit_linear_partial, 3, 6, 8, 11, 14, 19, 21, 23, 26, 28, 29
  probit_linearRE, 3, 6, 8, 11, 14, 19, 21, 23, 24, 28, 31