

Package: spldv (via r-universe)

October 11, 2026

Type Package

Title Spatial Models for Limited Dependent Variables

Version 0.2.2

Date 2026-07-08

Description The current version of this package estimates spatial autoregressive models for binary dependent variables using GMM estimators <[doi:10.18637/jss.v107.i08](https://doi.org/10.18637/jss.v107.i08)> and RIS estimator <[doi:10.1007/978-3-662-05617-2_8](https://doi.org/10.1007/978-3-662-05617-2_8)>. It supports one-step (Pinkse and Slade, 1998) <[doi:10.1016/S0304-4076\(97\)00097-3](https://doi.org/10.1016/S0304-4076(97)00097-3)> and two-step GMM estimator along with the linearized GMM estimator proposed by Klier and McMillen (2008) <[doi:10.1198/073500107000000188](https://doi.org/10.1198/073500107000000188)>. It also allows for either Probit or Logit model and compute the average marginal effects. The RIS estimator allows to estimate the SAR and SEM model. All these models are presented in Sarrias and Piras (2023) <[doi:10.1016/j.jocm.2023.100432](https://doi.org/10.1016/j.jocm.2023.100432)>.

Encoding UTF-8

RoxygenNote 7.3.3

Depends R (>= 4.0)

Imports Formula, Matrix, maxLik, stats, sphet, memisc, car, methods, numDeriv, MASS, spatialreg, miscTools, sandwich

Suggests spdep

License GPL (>= 2)

URL <https://github.com/gpiras/spldv>

BugReports <https://github.com/gpiras/spldv/issues>

LazyData no

Config/pak/sysreqs libabsl-dev cmake libgdal-dev gdal-bin libgeos-dev make libicu-dev libssl-dev libproj-dev libsqlite3-dev libudunits2-dev

Repository <https://gpiras.r-universe.dev>

Date/Publication 2026-07-10 13:05:23 UTC

RemoteUrl <https://github.com/gpiras/spldv>
RemoteRef HEAD
RemoteSha 853617619ce1e1a7cad24cbb1998dd17a0749ab

Contents

getSummary.bingmm	2
getSummary.binlgmm	3
getSummary.binris	3
impacts.bingmm	4
make.instruments	10
predict.bingmm	11
predict.binlgmm	13
predict.binris	15
sbinaryGMM	16
sbinaryLGMM	22
sbinaryRis	24
Index	30

getSummary.bingmm	<i>Get Model Summaries for use with "mtable" for objects of class bingmm</i>
-------------------	--

Description

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

Usage

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

Arguments

- obj a bingmm object,
- alpha level of the confidence intervals,
- ... further arguments,

Details

For more details see package **memisc**.

Value

A list with an array with coefficient estimates and a vector containing the model summary statistics.

getSummary.binlgmm	<i>Get Model Summaries for use with "mtable" for objects of class binlgmm</i>
--------------------	---

Description

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

Usage

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

Arguments

obj	a binlgmm object,
alpha	level of the confidence intervals,
...	further arguments,

Details

For more details see package **memisc**.

Value

A list with an array with coefficient estimates and a vector containing the model summary statistics.

getSummary.binris	<i>Get Model Summaries for use with "mtable" for objects of class binris</i>
-------------------	--

Description

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

Usage

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

Arguments

obj	a binris object,
alpha	level of the confidence intervals,
...	further arguments,

Details

For more details see package **memisc**.

Value

A list with an array with coefficient estimates and a vector containing the model summary statistics.

impacts.bingmm	<i>Probability Impacts for Spatial Binary Response Models</i>
----------------	---

Description

Computes probability impacts for fitted spatial binary response models. The function translates estimated coefficients into changes in fitted outcome probabilities, taking account of the nonlinear binary-response link and, when present, the spatial multiplier. It can return scalar average impacts, origin-specific impacts from one spatial unit to all destination units, or cumulative impacts by spatial feedback order. Standard errors can be obtained by the Delta method or by Monte Carlo simulation from the estimated coefficient distribution.

Usage

```
## S3 method for class 'bingmm'
impacts(
  obj,
  data = NULL,
  variable = NULL,
  from.unit = 1,
  dydx = c("exact", "numeric"),
  change = NULL,
  vce = c("robust", "efficient", "ml"),
  type = c("delta", "mc"),
  result = c("summary", "from.region", "cumulative"),
  vcov = NULL,
  het = TRUE,
  empirical = FALSE,
  approximation = FALSE,
  R = 50,
  pw = 5,
  draws = NULL,
  Q = 3,
  tol = 1e-06,
  verbose = FALSE,
  ...
)

## S3 method for class 'binlgmm'
impacts(
```

```

    obj,
    data = NULL,
    variable = NULL,
    from.unit = 1,
    dydx = c("exact", "numeric"),
    change = NULL,
    vce = c("robust", "efficient", "ml"),
    type = c("delta", "mc"),
    result = c("summary", "from.region", "cumulative"),
    vcov = NULL,
    het = TRUE,
    empirical = FALSE,
    approximation = FALSE,
    R = 50,
    pw = 5,
    draws = NULL,
    Q = 3,
    tol = 1e-06,
    verbose = FALSE,
    ...
)

```

```
## S3 method for class 'binris'
```

```

impacts(
  obj,
  data = NULL,
  variable = NULL,
  from.unit = 1,
  dydx = c("exact", "numeric"),
  change = NULL,
  vce = c("robust", "efficient", "ml"),
  type = c("delta", "mc"),
  result = c("summary", "from.region", "cumulative"),
  vcov = NULL,
  het = TRUE,
  empirical = FALSE,
  approximation = FALSE,
  R = 50,
  pw = 5,
  draws = NULL,
  Q = 3,
  tol = 1e-06,
  verbose = FALSE,
  ...
)

```

```
## S3 method for class 'impacts.bingmm'
```

```
print(x, ...)
```

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

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

Arguments

obj	An object of class bingmm, binlgmm, or binris.
data	Optional data frame used to compute fitted probabilities and probability impacts. If omitted, the estimation data stored in obj is used.
variable	Character string giving the original variable for which impacts are requested. This is required for result = "from.region" and result = "cumulative". If the model formula contains factor(CP), use variable = "CP", not variable = "factor(CP)". For result = "summary", impacts are computed for all variables in the fitted model and variable is ignored.
from.unit	Integer identifying the source spatial unit where the covariate change is applied. This argument is used only when result = "from.region" or result = "cumulative". The returned vector gives the effect of that source-unit change on the fitted probabilities of all destination units.
dydx	Character string indicating how continuous-variable impacts are computed. "exact" uses analytical derivatives when available. "numeric" uses predictive finite differences. Factor, ordered, and logical variables are always evaluated as discrete probability contrasts, because derivatives with respect to categories are not meaningful.
change	Optional numeric vector of length two, c(lower, upper), used with continuous variables when a discrete probability change is desired instead of a derivative. For example, change = c(10, 20) computes the change in fitted probabilities when the source-unit value of variable is set from 10 to 20. This argument is allowed only when result != "summary".
vce	Character string indicating which variance-covariance matrix of the estimated coefficients should be used when vcov is not supplied. For one-step GMM, the options are "robust" and "ml". For two-step GMM, the options are "robust", "efficient", and "ml". The option "ml" is exploratory and evaluates the variance-covariance matrix of the RIS estimator at the GMM estimates.
type	Character string indicating how standard errors are computed. "delta" uses the Delta method. "mc" simulates coefficient draws from the estimated asymptotic distribution and recomputes the impacts for each draw.
result	Character string selecting the output object. "summary" returns scalar average total, direct, and indirect effects. "from.region" returns the vector of destination-unit effects generated by a change in the source unit from.unit. "cumulative" returns these source-to-destination effects after successively adding spatial feedback orders from zero up to Q.
vcov	Optional user-supplied variance-covariance matrix for the estimated coefficients. If supplied, it overrides vce.

het	Logical. If TRUE (the default), the heteroskedasticity induced by the spatial multiplier is taken into account when converting the latent index to fitted probabilities.
empirical	Logical argument passed to <code>MASS::mvrnorm()</code> when <code>type = "mc"</code> . If TRUE, the simulated draws have empirical mean and covariance equal to the supplied values.
approximation	Logical. If TRUE, the spatial multiplier $(I - \lambda W)^{-1}$ is approximated by the finite expansion $I + \lambda W + \lambda^2 W^2 + \dots + \lambda^q W^q$ using <code>pw</code> as the maximum power. This option applies to the inverse used in <code>result = "summary"</code> and <code>result = "from.region"</code> ; cumulative output is controlled separately by <code>Q</code> .
R	Integer giving the number of coefficient draws used when <code>type = "mc"</code> .
pw	Integer giving the maximum power used when <code>approximation = TRUE</code> . The default is 5.
draws	Optional matrix of user-supplied coefficient draws for Monte Carlo inference. Columns must match the estimated coefficient vector.
Q	Integer giving the highest spatial-feedback order reported when <code>result = "cumulative"</code> . The output contains <code>Q + 1</code> columns, labelled <code>Q:0</code> , <code>Q:1</code> , ..., <code>Q:Q</code> .
tol	Numerical tolerance passed to <code>MASS::mvrnorm()</code> when checking positive definiteness of the coefficient covariance matrix.
verbose	Logical. If TRUE, progress messages are printed for computations that loop over spatial units.
...	Further arguments. Currently ignored.
x	An object of class <code>impacts.bingmm</code> for print methods.
object	An object of class <code>impacts.bingmm</code> for summary methods.
digits	Number of digits used by the print method.

Details

Model and probability impacts: The fitted model is written in latent-index form as

$$y^* = X\beta + WX\gamma + \lambda W y^* + \epsilon = Z\delta + \lambda W y^* + \epsilon,$$

where $y = 1[y^* > 0]$. For `link = "probit"`, ϵ is standard normal. For `link = "logit"`, ϵ follows the standard logistic distribution. The RIS estimator is probit-based.

In spatial binary response models, coefficients are not probability effects. A change in a covariate affects the latent index, propagates through the spatial multiplier, and is then transformed by the probit or logit link into fitted probabilities. For a continuous variable x_r , the analytical probability-impact matrix has the form

$$C_r(\theta) = \text{diag}(f(a)) D_\lambda^{-1} A_\lambda^{-1} (I_n \beta_r + W \gamma_r),$$

where $f()$ is the link density, $A_\lambda = I - \lambda W$, and D_λ is a diagonal scaling matrix based on the diagonal elements of the variance-covariance matrix of $A_\lambda^{-1} \epsilon$. Rows of $C_r(\theta)$ correspond to destination units and columns correspond to source units.

Average, from-region, and cumulative output: With `result = "summary"`, `impacts()` returns scalar summaries for each variable: the average total effect $n^{-1}\iota_n' C_r \iota_n$, the average direct effect $n^{-1}\text{tr}(C_r)$, and the average indirect effect, defined as total minus direct. The `S3` method `summary()` is designed only for this scalar-summary output; it formats the coefficient table into total, direct, and indirect blocks.

With `result = "from.region"`, `from.unit` is interpreted as the source region. The returned object gives the probability effect of changing variable in that source region on every destination region. For a continuous variable with `dydx = "exact"` and no change, this is the corresponding column of the analytical impact matrix. For finite differences and discrete variables, it is computed as a difference in fitted probabilities.

With `result = "cumulative"`, the function reports how the same source-region impact accumulates as additional feedback orders are included. The column `Q:0` uses only the zero-order term I . The column `Q:1` uses $I + \lambda W$. In general, `Q:q` uses $I + \lambda W + \dots + \lambda^q W^q$. This output is useful for seeing how much of a spatial probability impact is local and how much is generated by higher-order spatial feedback.

Continuous derivatives and discrete contrasts: For continuous variables, `dydx = "exact"` uses analytical derivatives where implemented. This is usually preferred because it avoids numerical step-size sensitivity. `dydx = "numeric"` computes predictive finite differences by perturbing the source-unit value of the variable and comparing fitted probabilities. Supplying `change = c(lower, upper)` requests a finite probability change from lower to upper, not a derivative divided by `upper - lower`.

Factor, ordered, and logical variables are handled as discrete probability contrasts. For factors, contrasts are reported relative to the reference level, with one component for each non-reference level. For logical variables, the contrast is from `FALSE` to `TRUE`. These contrasts are the appropriate probability-scale analogues of marginal effects for non-continuous regressors.

Inference: When `type = "delta"`, standard errors are computed by applying the Delta method to the selected impact measure. Internally, the Jacobian of the impact vector with respect to the estimated coefficients is combined with the coefficient variance-covariance matrix. When `type = "mc"`, coefficient draws are generated from the estimated asymptotic normal distribution, impacts are recomputed for each draw, and the simulation standard deviation is used as the standard error. User-supplied draws can be passed through `draws`. Draws that imply inadmissible spatial parameters are discarded.

Formula restrictions: The current implementation does not support formulas containing `poly()`, `scale()`, `splines::bs()`, or `splines::ns()` inside `impacts()`. These transformations can make probability impacts ambiguous because the derivative or discrete change must be propagated through the transformed design matrix and the spatial multiplier. Create transformed variables explicitly before fitting the model. For example, use `x2 <- x^2` and include `x + x2`, or create `x_s <- as.numeric(scale(x))` before estimation and request impacts for `"x_s"`.

Value

An object of class `impacts.bingmm`. The object contains at least two elements: `out`, with the requested estimates and standard errors, and `result`, recording the selected output mode. For `result = "summary"`, `out` is a table that can be formatted by `summary()`. For `result = "from.region"`, `out` is a data frame for continuous or logical variables and a list of data frames for factor variables. For `result = "cumulative"`, `out` is a list of cumulative tables, or a nested list when the requested variable is a factor.

Author(s)

Mauricio Sarrias and Gianfranco Piras.

See Also

[sbinaryGMM](#), [sbinaryLGMM](#).

Examples

```
# Data set
data(oldcol, package = "spdep")

# Create dependent (dummy) variable
COL.OLD$CRIMED <- as.numeric(COL.OLD$CRIME > 35)

# Two-step probit GMM estimator. CP is used as a factor directly in the
# formula; impacts should later be requested with variable = "CP".
ts <- sbinaryGMM(CRIMED ~ INC * HOVAL + factor(CP) | HOVAL,
                 link = "probit",
                 listw = spdep::nb2listw(COL.nb, style = "W"),
                 data = COL.OLD,
                 type = "twostep")

# Scalar average probability impacts: total, direct, and indirect effects.
summary(impacts(ts, type = "delta"))

# The same scalar summaries using simulation-based standard errors.
summary(impacts(ts, type = "mc", R = 100))

# Use the efficient two-step GMM variance-covariance matrix.
summary(impacts(ts, type = "delta", vce = "efficient"))

# Ignore the heteroskedasticity adjustment induced by the spatial multiplier.
summary(impacts(ts, type = "delta", vce = "efficient", het = FALSE))

# Source-region impact: change INC in region 20 and report effects on all
# destination regions. With dydx = "exact" this is an analytical derivative.
imp_inc <- impacts(ts, type = "delta", result = "from.region",
                  from.unit = 20, variable = "INC")
head(imp_inc$out, 6)

# Discrete probability change for a continuous variable: set INC in region 20
# from 10 to 20 and compare fitted probabilities.
imp_inc_10_20 <- impacts(ts, type = "delta", result = "from.region",
                        from.unit = 20, variable = "INC",
                        dydx = "numeric", change = c(10, 20))
head(imp_inc_10_20$out, 6)

# Factor impacts are discrete contrasts relative to the reference level. Use
# the original variable name, not "factor(CP)".
imp_cp <- impacts(ts, type = "delta", result = "from.region",
                 from.unit = 20, variable = "CP")
```

```

names(imp_cp$out)
head(imp_cp$out[[1]], 6)

# Cumulative source-region impacts up to third-order spatial feedback.
cum_inc <- impacts(ts, type = "delta", result = "cumulative",
                  from.unit = 20, variable = "INC", Q = 3)
names(cum_inc$out)
head(cum_inc$out$`Q:0`, 6)
head(cum_inc$out$`Q:3`, 6)

# Marginal effects using the RIS estimator.
ris_sar <- sbinaryRis(CRIMED ~ INC + HOVAL, data = COL.OLD,
                    R = 50,
                    listw = spdep::nb2listw(COL.nb, style = "W"))
summary(impacts(ris_sar, type = "delta"))
summary(impacts(ris_sar, type = "mc", R = 100))

```

make.instruments

Make instruments for spatial models

Description

Make instruments for spatial models

Usage

```
make.instruments(listw, x, q)
```

Arguments

listw	object. An object of class <code>matrix</code> , or <code>Matrix</code> .
x	variable(s) to be lagged
q	number of lags

Author(s)

Mauricio Sarrias and Gianfranco Piras.

predict.bingmm	<i>Predictions for Spatial Binary GMM Models</i>
----------------	--

Description

Computes predicted probabilities for spatial binary response models estimated via GMM. Supports both probit and logit links, accounts for spatial heteroskedasticity, and optionally returns standard errors using the Delta method with robust or efficient variance estimators.

Usage

```
## S3 method for class 'bingmm'
predict(
  object,
  newdata,
  Sinv = NULL,
  het = TRUE,
  approximation = FALSE,
  pw = 5,
  ses = FALSE,
  vce = c("robust", "efficient", "ml"),
  theta = NULL,
  ...
)
```

Arguments

object	An object of class bingmm returned by a spatial binary GMM estimation function.
newdata	An optional data frame in which to look for variables with which to predict. If omitted, the original data used to fit the model is used.
Sinv	Optional user-supplied spatial multiplier matrix $S = (I - \lambda W)^{-1}$. If NULL, it is computed using the spatial weight matrix.
het	Logical. If TRUE, assumes a heteroskedastic error structure with spatially varying variances.
approximation	Logical. If TRUE, uses power-series approximation to compute the inverse spatial matrix.
pw	Integer. Power-order to use when approximation = TRUE.
ses	Logical. If TRUE, standard errors of the predictions are computed using the Delta method.
vce	Type of variance-covariance estimator: "robust" (default), "efficient", or "ml".
theta	Optional parameter vector (including lambda) to use for prediction instead of the estimated one.
...	Additional arguments (currently unused).

Details

The function computes predicted probabilities $\hat{p}_i = F(a_i)$ where a_i is a spatially filtered linear index. In the presence of heteroskedasticity (`het = TRUE`), the normalization involves the row-wise standard deviation of the spatial multiplier. When `ses = TRUE`, standard errors are computed using the analytical Jacobian of the prediction function with respect to the parameters and the estimated variance-covariance matrix.

Value

A numeric vector of predicted probabilities if `ses = FALSE`. If `ses = TRUE`, returns a matrix with:

`p_hat` Predicted probabilities.

`Std. error` Standard errors of the predictions.

`z value` Z-statistics.

`Pr(> z)` Two-sided p-values.

Author(s)

Mauricio Sarrias and Gianfranco Piras.

See Also

[sbinaryGMM](#), [vcov.bingmm](#)

Examples

```
# Data set
data(oldcol, package = "spdep")

# Create dependent (dummy) variable
COL.OLD$CRIMED <- as.numeric(COL.OLD$CRIME > 35)

# Two-step (Probit) GMM estimator
ts <- sbinaryGMM(CRIMED ~ INC*HOVAL + factor(CP),
  link = "probit",
  listw = spdep::nb2listw(COL.nb, style = "W"),
  data = COL.OLD,
  type = "twostep",
  verbose = FALSE)

# Compute just the predicted probability
p_hat <- predict(ts)

# Compute predicted probability and standard errors
res <- predict(ts, ses = TRUE, het = TRUE, vce = "efficient")
head(res)
```

predict.binlgmm

*Predictions for Spatial Binary LGMM Models***Description**

Computes predicted probabilities for spatial binary response models estimated via LGMM. Supports both probit and logit links, accounts for spatial heteroskedasticity, and optionally returns standard errors using the Delta method.

Usage

```
## S3 method for class 'binlgmm'
predict(
  object,
  newdata,
  Sinv = NULL,
  het = TRUE,
  approximation = FALSE,
  pw = 5,
  ses = FALSE,
  theta = NULL,
  ...
)
```

Arguments

object	An object of class binlgmm returned by a spatial binary GMM estimation function.
newdata	An optional data frame in which to look for variables with which to predict. If omitted, the original data used to fit the model is used.
Sinv	Optional user-supplied spatial multiplier matrix $S = (I - \lambda W)^{-1}$. If NULL, it is computed using the spatial weight matrix.
het	Logical. If TRUE, assumes a heteroskedastic error structure with spatially varying variances.
approximation	Logical. If TRUE, uses power-series approximation to compute the inverse spatial matrix.
pw	Integer. Power-order to use when approximation = TRUE.
ses	Logical. If TRUE, standard errors of the predictions are computed using the Delta method.
theta	Optional parameter vector (including lambda) to use for prediction instead of the estimated one.
...	Additional arguments (currently unused).

Details

The function computes predicted probabilities $\hat{p}_i = F(a_i)$ where a_i is a spatially filtered linear index. In the presence of heteroskedasticity (`het = TRUE`), the normalization involves the row-wise standard deviation of the spatial multiplier. When `ses = TRUE`, standard errors are computed using the analytical Jacobian of the prediction function with respect to the parameters and the estimated variance-covariance matrix.

Value

A numeric vector of predicted probabilities if `ses = FALSE`. If `ses = TRUE`, returns a matrix with:

`p_hat` Predicted probabilities.

`Std. error` Standard errors of the predictions.

`z value` Z-statistics.

`Pr(> z)` Two-sided p-values.

Author(s)

Mauricio Sarrias and Gianfranco Piras.

See Also

[sbinaryLGMM](#), [vcov.binlgmm](#)

Examples

```
# Data set
data(oldcol, package = "spdep")

# Create dependent (dummy) variable
COL.OLD$CRIMED <- as.numeric(COL.OLD$CRIME > 35)

# Estimate the model
lgmm <- sbinaryLGMM(CRIMED ~ INC + HOVAL | INC,
  link = "probit",
  listw = spdep::nb2listw(COL.nb, style = "W"),
  nins = 3,
  data = COL.OLD)

# Predicted probabilities with SES
out <- predict(lgmm, ses = TRUE)
head(out, 5)
```

predict.binris

*Predictions for Spatial Binary RIS Models***Description**

Computes predicted probabilities for spatial binary response models estimated via RIS. Supports probit models and accounts for spatial heteroskedasticity, and optionally returns standard errors using the Delta method.

Usage

```
## S3 method for class 'binris'
predict(
  object,
  newdata,
  Sinv = NULL,
  het = TRUE,
  approximation = FALSE,
  pw = 5,
  ses = FALSE,
  theta = NULL,
  ...
)
```

Arguments

object	An object of class binris.
newdata	An optional data frame in which to look for variables with which to predict. If omitted, the original data used to fit the model is used.
Sinv	Optional user-supplied spatial multiplier matrix $S = (I - \lambda W)^{-1}$. If NULL, it is computed using the spatial weight matrix.
het	Logical. If TRUE, assumes a heteroskedastic error structure with spatially varying variances.
approximation	Logical. If TRUE, uses power-series approximation to compute the inverse spatial matrix.
pw	Integer. Power-order to use when approximation = TRUE.
ses	Logical. If TRUE, standard errors of the predictions are computed using the Delta method.
theta	Optional parameter vector (including lambda) to use for prediction instead of the estimated one.
...	Additional arguments (currently unused).

Details

The function computes predicted probabilities $\hat{p}_i = F(a_i)$ where a_i is a spatially filtered linear index. In the presence of heteroskedasticity (`het = TRUE`), the normalization involves the row-wise standard deviation of the spatial multiplier. When `ses = TRUE`, standard errors are computed using the analytical Jacobian of the prediction function with respect to the parameters and the estimated variance-covariance matrix.

Value

A numeric vector of predicted probabilities if `ses = FALSE`. If `ses = TRUE`, returns a matrix with:

`p_hat` Predicted probabilities.

`Std. error` Standard errors of the predictions.

`z value` Z-statistics.

`Pr(> z)` Two-sided p-values.

Author(s)

Mauricio Sarrias and Gianfranco Piras.

See Also

[sbinaryRis](#)

Examples

```
data(oldcol, package = "spdep")
# Create dependent (dummy) variable
COL.OLD$CRIMED <- as.numeric(COL.OLD$CRIME > 35)

# Estimate the model
ris_sar <- sbinaryRis(CRIMED ~ INC + HOVAL, data = COL.OLD,
                     R = 50,
                     listw = spdep::nb2listw(COL.nb, style = "W"))

# Predicted probabilities with SES
out <- predict(ris_sar, ses = TRUE)
head(out, 5)
```

Description

Estimates a Spatial Autoregressive (SAR) model for binary outcomes using the Generalized Method of Moments (GMM). The model supports both Probit and Logit links, with one-step or two-step GMM estimation. The structural equation is:

$$y^* = X\beta + WX\gamma + \lambda Wy^* + \epsilon = Z\delta + \lambda Wy^* + \epsilon,$$

where $y = 1$ if $y^* > 0$ and 0 otherwise. The error term ϵ follows a standard normal distribution (`link = "probit"`) or logistic distribution (`link = "logit"`).

Usage

```
sbinaryGMM(
  formula,
  data,
  listw = NULL,
  nins = 2,
  link = c("probit", "logit"),
  winitial = c("optimal", "identity"),
  s.matrix = c("robust", "iid"),
  type = c("onestep", "twostep"),
  gradient = TRUE,
  start = NULL,
  cons.opt = FALSE,
  approximation = FALSE,
  verbose = TRUE,
  print.init = FALSE,
  pw = 5,
  tol.solve = .Machine$double.eps,
  fastmom = FALSE,
  ...
)

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

## S3 method for class 'bingmm'
vcov(
  object,
  vce = c("robust", "efficient", "ml"),
  method = "bhhh",
  R = 1000,
  tol.solve = .Machine$double.eps,
  ...
)

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

```
## S3 method for class 'bingmm'
summary(
  object,
  vce = c("robust", "efficient", "ml"),
  method = "bhhh",
  R = 1000,
  tol.solve = .Machine$double.eps,
  ...
)

## S3 method for class 'summary.bingmm'
print(x, digits = max(5, getOption("digits") - 3), ...)
```

Arguments

formula	A symbolic description of the model of the form $y \sim x \mid wx$ where y is the binary dependent variable, x are the independent variables. The variables after $\mid$ are those variables that enter spatially lagged: WX .
data	A data.frame containing the variables in the model.
listw	A spatial weights matrix of class listw, matrix, or Matrix.
nins	Integer. Order of spatial instruments to generate; default is $nins = 2$, such that $H = (Z, WZ, W^2Z)$ are used as instruments.
link	Character string; distributional assumption on the error term. Either "probit" (default) or "logit".
winitial	Character string. A string indicating the initial moment-weighting matrix Ψ ; it can be either <code>winitial = "optimal"</code> (the default, $(H'H/n)^{-1}$) or <code>winitial = "identity"</code> .
s.matrix	Character string. Only valid if <code>type = "twostep"</code> is used. This is a string indicating the type of variance-covariance matrix $\hat{S}$ to be used in the second-step procedure; it can be <code>s.matrix = "robust"</code> (the default) or <code>s.matrix = "iid"</code> .
type	Character string. A string indicating whether the one-step (<code>type = "onestep"</code>), or two-step GMM (<code>type = "twostep"</code>) should be computed.
gradient	Logical. Only for testing procedures. Should the analytic gradient be used in the GMM optimization procedure? TRUE as default. If FALSE, then the numerical gradient is used.
start	Vector of optional starting values. If not NULL, the user must provide a vector of initial parameters for the optimization procedure. When <code>start = NULL</code> , sbinaryGMM uses the traditional Probit or Logit estimates as initial values for the parameters, and the correlation between y and Wy as initial value for λ .
cons.opt	Logical. Should a constrained optimization procedure for λ be used? FALSE as default.
approximation	Logical. If TRUE, then $(I - \lambda W)^{-1}$ is approximated as $I + \lambda W + \lambda^2 W^2 + \lambda^3 W^3 + \dots + \lambda^q W^q$. The default is FALSE.
verbose	Logical. If TRUE, the code reports messages and some values during optimization.

<code>print.init</code>	Logical. If TRUE the initial parameters used in the optimization of the first step are printed.
<code>pw</code>	Integer. The power used for the approximation $I + \lambda W + \lambda^2 W^2 + \lambda^3 W^3 + \dots + \lambda^q W^q$. The default is 5.
<code>tol.solve</code>	Tolerance for <code>solve()</code> .
<code>fastmom</code>	Logical. Use faster approximate gradient for λ ? Default is FALSE.
<code>...</code>	additional arguments passed to <code>maxLik</code> .
<code>vce</code>	Character string. A string indicating what kind of standard errors should be computed when using <code>summary</code> . For the one-step GMM estimator, the options are "robust" and "ml". For the two-step GMM estimator, the options are "robust", "efficient" and "ml". The option "vce = ml" is an exploratory method that evaluates the VC of the RIS estimator using the GMM estimates.
<code>method</code>	Character string. Only valid if <code>vce = "ml"</code> . It indicates the algorithm used to compute the Hessian matrix of the RIS estimator. The default is "bhhh".
<code>R</code>	Integer. Only valid if <code>vce = "ml"</code> . It indicates the number of draws used to compute the simulated probability in the RIS estimator.
<code>x, object</code>	an object of class <code>bingmm</code> .
<code>digits</code>	Integer. The number of digits for summary methods.

Details

The data generating process is:

$$y^* = X\beta + WX\gamma + \lambda Wy^* + \epsilon = Z\delta + \lambda Wy^* + \epsilon$$

where $y = 1$ if $y^* > 0$ and 0 otherwise; $\epsilon \sim N(0, 1)$ if `link = "probit"` or $\epsilon \sim L(0, \pi^2/3)$ if `link = "logit"`. The general GMM estimator minimizes:

$$J(\theta) = g'(\theta)\hat{\Psi}g(\theta),$$

where $\theta = (\beta, \gamma, \lambda)$ and

$$g = n^{-1}H'v,$$

where v is the generalized residuals. Let $Z = (X, WX)$, then the instrument matrix H contains the linearly independent columns of $H = (Z, WZ, \dots, W^q Z)$. The one-step GMM estimator minimizes $J(\theta)$ setting either $\hat{\Psi} = I_p$ if `winitial = "identity"` or $\hat{\Psi} = (H'H/n)^{-1}$ if `winitial = "optimal"`. The two-step GMM estimator uses an additional step to achieve higher efficiency by computing the variance-covariance matrix of the moments $\hat{S}$ to weight the sample moments. This matrix is computed using the residuals or generalized residuals from the first-step, which are consistent. This matrix is computed as $\hat{S} = n^{-1} \sum_{i=1}^n h_i(f^2/(F(1-F)))h_i'$ if `s.matrix = "robust"` or $\hat{S} = n^{-1} \sum_{i=1}^n \hat{v}_i h_i h_i'$, where $\hat{v}$ are the first-step generalized residuals.

Value

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

`coefficients` the estimated coefficients,

call	the matched call,
callF	the full matched call,
X	the X matrix, which contains also WX if the second part of the formula is used,
H	the H matrix of instruments used,
y	the dependent variable,
listw	the spatial weight matrix,
link	the string indicating the distribution of the error term,
Psi	the moment-weighting matrix used in the last round,
type	type of model that was fitted,
s.matrix	the type of S matrix used in the second round,
winitial	the moment-weighting matrix used for the first step procedure,
opt	object of class maxLik,
approximation	a logical value indicating whether approximation was used to compute the inverse matrix,
pw	the powers for the approximation,
formula	the formula,
data	the data,
contrastsX	the contrasts used in the first part of the formula,
contrastsD	the contrasts used in the second part of the formula,
Xlevels	a record of the levels of the factors used in fitting,
fastmom	whether the model was fitted using the faster gradient for λ .

Author(s)

Mauricio Sarrias and Gianfranco Piras.

References

- Pinkse, J., & Slade, M. E. (1998). Contracting in space: An application of spatial statistics to discrete-choice models. *Journal of Econometrics*, 85(1), 125-154.
- Fleming, M. M. (2004). Techniques for estimating spatially dependent discrete choice models. In *Advances in spatial econometrics* (pp. 145-168). Springer, Berlin, Heidelberg.
- Klier, T., & McMillen, D. P. (2008). Clustering of auto supplier plants in the United States: generalized method of moments spatial logit for large samples. *Journal of Business & Economic Statistics*, 26(4), 460-471.
- LeSage, J. P., Kelley Pace, R., Lam, N., Campanella, R., & Liu, X. (2011). New Orleans business recovery in the aftermath of Hurricane Katrina. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 174(4), 1007-1027.
- Piras, G., & Sarrias, M. (2023). One or Two-Step? Evaluating GMM Efficiency for Spatial Binary Probit Models. *Journal of choice modelling*, 48, 100432.
- Piras, G., & Sarrias, M. (2023). GMM Estimators for Binary Spatial Models in R. *Journal of Statistical Software*, 107(8), 1-33.

See Also

[sbinaryLGMM](#), [impacts.bingmm](#).

Examples

```
# Data set
data(oldcol, package = "spdep")

# Create dependent (dummy) variable
COL.OLD$CRIMED <- as.numeric(COL.OLD$CRIME > 35)

# Two-step (Probit) GMM estimator
ts <- sbinaryGMM(CRIMED ~ INC + HOVAL,
  link = "probit",
  listw = spdep::nb2listw(COL.nb, style = "W"),
  data = COL.OLD,
  type = "twostep",
  verbose = TRUE)

# Robust standard errors
summary(ts)
# Efficient standard errors
summary(ts, vce = "efficient")

# One-step (Probit) GMM estimator
os <- sbinaryGMM(CRIMED ~ INC + HOVAL,
  link = "probit",
  listw = spdep::nb2listw(COL.nb, style = "W"),
  data = COL.OLD,
  type = "onestep",
  verbose = TRUE)

summary(os)

# One-step (Logit) GMM estimator with identity matrix as initial weight matrix
os_l <- sbinaryGMM(CRIMED ~ INC + HOVAL,
  link = "logit",
  listw = spdep::nb2listw(COL.nb, style = "W"),
  data = COL.OLD,
  type = "onestep",
  winitial = "identity",
  verbose = TRUE)

summary(os_l)

# Two-step (Probit) GMM estimator with WX
ts_wx <- sbinaryGMM(CRIMED ~ INC + HOVAL | INC + HOVAL,
  link = "probit",
  listw = spdep::nb2listw(COL.nb, style = "W"),
  data = COL.OLD,
  type = "twostep",
  verbose = FALSE)

summary(ts_wx)
```

```
# Constrained two-step (Probit) GMM estimator
ts_c <- sbinaryGMM(CRIMED ~ INC + HOVAL,
  link = "probit",
  listw = spdep::nb2listw(COL.nb, style = "W"),
  data = COL.OLD,
  type = "twostep",
  verbose = TRUE,
  cons.opt = TRUE)

summary(ts_c)
```

sbinaryLGMM

Estimation of SAR for binary models using Linearized GMM.

Description

Estimation of SAR model for binary dependent variables (either Probit or Logit), using Linearized GMM estimator suggested by Klier and McMillen (2008). The model is:

$$y^* = X\beta + WX\gamma + \lambda Wy^* + \epsilon = Z\delta + \lambda Wy^* + \epsilon,$$

where $y = 1$ if $y^* > 0$ and 0 otherwise; $\epsilon \sim N(0, 1)$ if link = "probit" or $\epsilon \sim L(0, \pi^2/3)$ link = "logit".

Usage

```
sbinaryLGMM(
  formula,
  data,
  listw = NULL,
  nins = 2,
  link = c("logit", "probit"),
  ...
)

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

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

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

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

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

Arguments

formula	A symbolic description of the model of the form $y \sim x \mid wx$ where y is the binary dependent variable, x are the independent variables. The variables after $\mid$ are those variables that enter spatially lagged: WX .
data	A data.frame containing the variables in the model.
listw	A spatial weights matrix of class listw, matrix, or Matrix.
nins	Integer. Order of spatial instruments to generate; default is $nins = 2$, such that $H = (Z, WZ, W^2Z)$ are used as instruments.
link	Character string; distributional assumption on the error term. Either "probit" (default) or "logit".
...	additional arguments.
x, object	an object of class binlgmm.
digits	Integer. The number of digits for summary methods.

Details

The steps for the linearized spatial Probit/Logit model are the following:

1. Estimate the model by standard Probit/Logit model, in which spatial autocorrelation and heteroskedasticity are ignored. The estimated values are β_0 . Calculate the generalized residuals assuming that $\lambda = 0$ and the gradient terms G_β and G_λ .
2. The second step is a two-stage least squares estimator of the linearized model. Thus regress G_β and G_λ on $H = (Z, WZ, W^2Z, \dots, W^qZ)$ and obtain the predicted values $\hat{G}$. Then regress $u_0 + G'_\beta \hat{\beta}_0$ on $\hat{G}$. The coefficients are the estimated values of β and λ .

The variance-covariance matrix can be computed using the traditional White-corrected coefficient covariance matrix from the last two-stage least squares estimator of the linearized model.

Value

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

coefficients	the estimated coefficients,
call	the matched call,
X	the X matrix, which contains also WX if the second part of the formula is used,
H	the H matrix of instruments used,
y	the dependent variable,
listw	the spatial weight matrix,
link	the string indicating the distribution of the error term,
fit	an object of lm representing the T2SLS,
formula	the formula,
data	the data,
contrastsX	the contrasts used in the first part of the formula,
contrastsD	the contrasts used in the second part of the formula,
Xlevels	a record of the levels of the factors used in fitting.

Author(s)

Mauricio Sarrias and Gianfranco Piras.

References

Klier, T., & McMillen, D. P. (2008). Clustering of auto supplier plants in the United States: generalized method of moments spatial logit for large samples. *Journal of Business & Economic Statistics*, 26(4), 460-471.

Piras, G., & Sarrias, M. (2023). One or Two-Step? Evaluating GMM Efficiency for Spatial Binary Probit Models. *Journal of choice modelling*, 48, 100432.

Piras, G., & Sarrias, M. (2023). GMM Estimators for Binary Spatial Models in R. *Journal of Statistical Software*, 107(8), 1-33.

See Also

[sbinaryGMM](#), [impacts.bingmm](#).

Examples

```
# Data set
data(oldcol, package = "spdep")

# Create dependent (dummy) variable
COL.OLD$CRIMED <- as.numeric(COL.OLD$CRIME > 35)

# LGMM for probit using q = 3 for instruments
lgmm <- sbinaryLGMM(CRIMED ~ INC + HOVAL | INC,
  link = "probit",
  listw = spdep::nb2listw(COL.nb, style = "W"),
  nins = 3,
  data = COL.OLD)

summary(lgmm)
```

sbinaryRis

Estimation of spatial probit model for binary outcomes using RIS (GHK) simulator

Description

Estimation of spatial probit model using RIS-normal (a.k.a GHK) simulator. The models can be the SAR or SEM probit model model. The SAR probit model has the following structure:

$$y^* = X\beta + WX\gamma + \lambda Wy^* + \epsilon = Z\delta + \lambda Wy^* + \epsilon = A_\lambda^{-1}Z\delta + u,$$

where $y = 1$ if $y^* > 0$ and 0 otherwise, $Z = (X, WX)$, $\delta = (\beta', \gamma')'$, $u = A_\lambda^{-1}\epsilon$ with $A_\lambda = (I - \lambda W)$, and $\epsilon \sim N(0, I)$. The SEM probit model has the following structure:

$$y^* = X\beta + WX\gamma + u = Z\delta + u$$

where $y = 1$ if $y^* > 0$ and 0 otherwise, $Z = (X, WX)$, $\delta = (\beta', \gamma')'$, $u = \rho Wu + \epsilon$ such that $u = A_\rho^{-1}\epsilon$, and $\epsilon \sim N(0, I)$,

Usage

```
sbinaryRis(
  formula,
  data,
  subset,
  na.action,
  listw = NULL,
  R = 20,
  model = c("SAR", "SEM"),
  varcov = c("invsigma", "sigma"),
  approximation = TRUE,
  pw = 5,
  start = NULL,
  Qneg = FALSE,
  print.init = FALSE,
  ...
)

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

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

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

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

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

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

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

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

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

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

Arguments

formula	A symbolic description of the model of the form $y \sim x \mid wx$ where y is the binary dependent variable, x are the independent variables. The variables after $\mid$ are those variables that enter spatially lagged: WX .
data	A <code>data.frame</code> containing the variables in the model.
subset	An optional vector specifying a subset of observations to be used in the fitting process.
na.action	A function which indicates what should happen when the data contains NAs.
listw	Object. An object of class <code>listw</code> , <code>matrix</code> , or <code>Matrix</code> .
R	Integer. The number of draws used in RIS (GHK) simulator.
model	String. A string indicating which model to estimate. It can be "SAR" for the spatial autoregressive spatial model or "SEM" for the spatial error model.
varcov	String. A string indicating over which variance-covariance matrix to apply the Cholesky factorization.
approximation	Logical. If TRUE (the default) then $(I - \lambda W)^{-1}$ or $(I - \rho W)^{-1}$ is approximated as $I + \lambda W + \lambda^2 W^2 + \lambda^3 W^3 + \dots + \lambda^q W^q$. If FALSE, then the inverse is computed without approximations.
pw	Integer. The power used for the approximation $I + \lambda W + \lambda^2 W^2 + \lambda^3 W^3 + \dots + \lambda^q W^q$. The default is 5.
start	If not NULL, the user must provide a vector of initial parameters for the optimization procedure. When <code>start = NULL</code> , <code>sbinaryRis</code> uses the traditional Probit estimates as initial values for the parameters, and the correlation between y and Wy as initial value for λ or ρ .
Qneg	Logical. Whether to construct the negative of the diagonal elements of Q . If <code>Qneg = FALSE</code> , then $q_{ii} = 2y_i - 1$. If <code>Qneg = TRUE</code> , then $q_{ii} = 1 - 2y_i$.
print.init	Logical. If TRUE the initial parameters used in the optimization of the first step are printed.
...	additional arguments passed to <code>maxLik</code> .
x, object	An object of class <code>bingmm</code> .
eigentol	The standard errors are only calculated if the ratio of the smallest and largest eigenvalue of the Hessian matrix is less than <code>eigentol</code> . Otherwise the Hessian is treated as singular.
digits	The number of digits for summary methods.

Details

The models are estimated by simulating the probabilities using the RIS-normal (GHK) simulator. The aim is to evaluate the multivariate density function $P(v = Qu < s)$, where Q is a diagonal matrix with entries $q_{ii} = 2y_i - 1$, and the $n \times 1$ vector s depends on whether the model is SAR or SEM. If model = "SAR", then $s = QA_\lambda^{-1}Z\delta$ where $A_\lambda = (I - \lambda W)$; if model = "SEM", then $s = QZ\delta$.

Let $\Sigma_v = QVar(u)Q'$ be the variance-covariance model of the transformed model. If model = "SAR" $\Sigma_v = Q\Sigma_\lambda Q'$, where $\Sigma_\lambda = (A'_\lambda A_\lambda)^{-1}$. If model = "SEM", then $\Sigma_v = Q\Sigma_\rho Q'$, where $\Sigma_\rho = (A'_\rho A_\rho)^{-1}$.

Since Σ_v is positive definite, there exists a Cholesky decomposition such that $C'C = \Sigma_v^{-1}$, where C is the upper triangular Cholesky matrix and Σ_v^{-1} is the precision matrix. Let $B = C^{-1}$. Then the random vector Qu can be replaced by $Qu = C^{-1}\eta = B\eta$, where η is a vector of standard normal variables. Then, the upper limit of the integration becomes $Qu = B\eta < s$, which can be written as $\eta < B^{-1}s = \nu$.

The RIS simulator is implemented by drawing a large number R of random vector η and computing η_{rj} recursively for $j = 1, \dots, n$. The parameters are estimated using the simulated maximum likelihood (SML) function:

$$\ln \tilde{L}(\theta) = \ln \left(\frac{1}{R} \sum_{r=1}^R \tilde{p}_r \right),$$

where:

$$\tilde{p}_r = \prod_{j=1}^n \Phi(\hat{\eta}_{jr}),$$

and $\Phi(\cdot)$ is the univariate CDF of the standard normal density.

By default, sbinaryRis compute the SML using the Cholesky transformation on Σ_v^{-1} , varcov = "invsigma". The transformation can also be applied to Σ_v using varcov = "sigma", which is slower than the previous option.

This estimator can take several minutes for large datasets. Thus, by default the inverse matrices A_λ^{-1} and A_ρ^{-1} are approximated using the Leontief expansion.

Value

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

varcov	the matrix over which the Chokesly factorization is applied,
model	type of model that was fitted,
Qneg	matrix Q used,
approximation	a logical value indicating whether approximation was used to compute the inverse matrix,
pw	the powers for the approximation,
call	the matched call,

X	the X matrix, which contains also WX if the second part of the formula is used,
y	the dependent variable,
listw	the spatial weight matrix,
formula	the formula,
R	number of draws,
mf	model frame,
data	the data,
contrastsX	the contrasts used in the first part of the formula,
contrastsD	the contrasts used in the second part of the formula,
Xlevels	a record of the levels of the factors used in fitting.

Author(s)

Mauricio Sarrias and Gianfranco Piras.

References

- Beron, K. J., & Vijverberg, W. P. (2004). Probit in a spatial context: a Monte Carlo analysis. In *Advances in spatial econometrics: methodology, tools and applications* (pp. 169-195). Berlin, Heidelberg: Springer Berlin Heidelberg.
- Fleming, M. M. (2004). Techniques for estimating spatially dependent discrete choice models. In *Advances in spatial econometrics* (pp. 145-168). Springer, Berlin, Heidelberg.
- Pace, R. K., & LeSage, J. P. (2016). Fast simulated maximum likelihood estimation of the spatial probit model capable of handling large samples. In *Spatial Econometrics: Qualitative and Limited Dependent Variables* (pp. 3-34). Emerald Group Publishing Limited.
- Piras, G., & Sarrias, M. (2023). One or Two-Step? Evaluating GMM Efficiency for Spatial Binary Probit Models. *Journal of choice modelling*, 48, 100432.

See Also

[sbinaryGMM](#), [impacts.binris](#).

Examples

```
data(oldcol, package = "spdep")

# Create dependent (dummy) variable
COL.OLD$CRIMED <- as.numeric(COL.OLD$CRIME > 35)

# Estimate SAR probit model using RIS simulator using Sigma_v^{-1}
ris_sar <- sbinaryRis(CRIMED ~ INC + HOVAL,
  data = COL.OLD,
  R = 50,
  listw = spdep::nb2listw(COL.nb, style = "W"),
  print.level = 2)

summary(ris_sar)
```

```
# Estimate SAR probit model using RIS simulator using Sigma_v
ris_sar_2 <- sbinaryRis(CRIMED ~ INC + HOVAL,
  data = COL.OLD,
  R = 50,
  listw = spdep::nb2listw(COL.nb, style = "W"),
  varcov = "sigma",
  print.level = 2)

summary(ris_sar_2)

# Estimate SDM probit model using RIS simulator
ris_sdm <- sbinaryRis(CRIMED ~ INC + HOVAL | INC + HOVAL,
  data = COL.OLD,
  R = 50,
  listw = spdep::nb2listw(COL.nb, style = "W"),
  print.level = 2)

summary(ris_sdm)

# Estimate SEM probit model using RIS simulator
ris_sem <- sbinaryRis(CRIMED ~ INC + HOVAL | INC,
  data = COL.OLD,
  R = 50,
  listw = spdep::nb2listw(COL.nb, style = "W"),
  model = "SEM")

summary(ris_sem)
```

Index

- * **effects**
 - impacts.bingmm, 4
- * **instruments**
 - make.instruments, 10
- * **marginal**
 - impacts.bingmm, 4
- * **models**
 - sbinaryGMM, 16
 - sbinaryLGMM, 22
 - sbinaryRis, 24
- * **prediction**
 - predict.bingmm, 11
 - predict.binlgmm, 13
 - predict.binris, 15

bread.binris(sbinaryRis), 24

coef.bingmm(sbinaryGMM), 16

coef.binlgmm(sbinaryLGMM), 22

coef.binris(sbinaryRis), 24

df.residual.binris(sbinaryRis), 24

estfun.binris(sbinaryRis), 24

getSummary.bingmm, 2

getSummary.binlgmm, 3

getSummary.binris, 3

impacts(impacts.bingmm), 4

impacts.bingmm, 4, 21, 24

impacts.binris, 28

logLik.binris(sbinaryRis), 24

make.instruments, 10

predict.bingmm, 11

predict.binlgmm, 13

predict.binris, 15

print.bingmm(sbinaryGMM), 16

print.binlgmm(sbinaryLGMM), 22

print.binris(sbinaryRis), 24

print.impacts.bingmm(impacts.bingmm), 4

print.summary.bingmm(sbinaryGMM), 16

print.summary.binlgmm(sbinaryLGMM), 22

print.summary.binris(sbinaryRis), 24

print.summary.impacts.bingmm(impacts.bingmm), 4

sbinaryGMM, 9, 12, 16, 24, 28

sbinaryLGMM, 9, 14, 21, 22

sbinaryRis, 16, 24

summary.bingmm(sbinaryGMM), 16

summary.binlgmm(sbinaryLGMM), 22

summary.binris(sbinaryRis), 24

summary.impacts.bingmm(impacts.bingmm), 4

terms.binris(sbinaryRis), 24

vcov.bingmm, 12

vcov.bingmm(sbinaryGMM), 16

vcov.binlgmm, 14

vcov.binlgmm(sbinaryLGMM), 22

vcov.binris(sbinaryRis), 24