Package 'SemiParBIVProbit'

August 19, 2016

Version 3.8

Author
Giampiero Marra <giampiero.marra@ucl.ac.uk> and Rosalba Radice <r.radice@bbk.ac.uk>

Maintainer Giampiero Marra <giampiero.marra@ucl.ac.uk>

Title Semiparametric Copula Regression Models

Description Routines for fitting several classes of copula regression models, with several types of covariate effects, in the presence of associated error equations, endogeneity, non-random sample selection or partial observability.

Depends R (>= 3.2.1), mgcv

Imports magic, VGAM, survey, trust, VineCopula, graphics, stats, utils, grDevices, ggplot2, matrixStats, mnormt, gamlss.dist, Rmpfr

Enhances sp

LazyLoad yes

License GPL (>= 2)

URL http://www.ucl.ac.uk/statistics/people/giampieromarra

Repository CRAN

Date/Publication 2016-08-19 00:52:59

NeedsCompilation no

R topics documented:

`SemiParBIVProbit-package`

Semiparametric Copula Regression Models

Description

This package provides functions for fitting several classes of copula regression models with several types of covariate effects. Many copula and marginal distributions are supported. The copula dependence parameter can also be specified as a flexible function of covariates.

This package, and in particular its main function `SemiParBIVProbit()`, has been originally designed to deal with bivariate binary responses and in fact the first model introduced was the semiparametric bivariate probit model (hence the name of the package). However, since then more models and options have been introduced.

The main fitting functions are listed below.

`SemiParBIVProbit()` fits bivariate regression models with binary responses (where the link functions are not restricted to be just probit). This is useful to fit bivariate binary models in the presence of (i) non-random sample selection or (ii) associated responses/endogeneity or (iii) partial observability. This function includes the `Model` argument which allows the user to fit the model in one of three situations mentioned above.

`copulaReg()` fits bivariate models with binary/discrete/continuous margins in the presence of associated responses/endogeneity.

`copulaSampleSel()` fits bivariate sample selection models with continuous/discrete response (instead of binary as it would be the case when using `SemiParBIVProbit()`).

`SemiParTRIV()` fits trivariate probit models (with and without double sample selection).

Other models/options will be incorporated from time to time.

Details

SemiParBIVProbit provides functions for fitting flexible copula regression models in various situations. The underlying representation and estimation of the modelling framework is based on a penalized regression spline approach, with automatic smoothness selection. Several marginal and copula distributions are available. The numerical routine carries out function minimization using a trust region algorithm in combination with an adaptation of a smoothness estimation fitting procedure for GAMs (see mgcv for more details on this last point).

Many smoothers are supported and are extracted from mgcv. Estimation is by penalized maximum likelihood with automatic smoothness estimation achieved by using an approximate AIC.

Confidence intervals for smooth components and nonlinear functions of the model parameters are derived using a Bayesian approach. Approximate p-values for testing individual smooth terms for equality to the zero function are also provided and based on the approach implemented in mgcv. The usual plotting and summary functions are also available. Model/variable selection is also possible via the use of shrinakge smoothers and/or information criteria.

The dependence parameter of the copula distribution can be specified as a function of covariates or a grouping factor if it makes sense. More generally, it is possible to specify all marginal distribution and copula parameters as functions of covariates, as it would be typical in GAMLSS.

Author(s)

Giampiero Marra (University College London, Department of Statistical Science) and Rosalba Radice (Birkbeck, University of London, Department of Economics, Mathematics and Statistics)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

Thanks to Bear Braumoeller (Department of Political Science, The Ohio State University) for suggesting the implementation of bivariate models with partial observability.

References

Key references:

Filippou P., Marra G. and Radice R. (submitted), A Penalised Estimation Approach to Additive Trivariate Probit Regression.

Marra G. and Radice R. (2011), Estimation of a Semiparametric Recursive Bivariate Probit in the Presence of Endogeneity. *Canadian Journal of Statistics*, 39(2), 259-279.

Marra G. and Radice R. (2013), A Penalized Likelihood Estimation Approach to Semiparametric Sample Selection Binary Response Modeling. *Electronic Journal of Statistics*, 7, 1432-1455.

Marra G. and Radice R. (submitted), Bivariate Copula Additive Models for Location, Scale and Shape.

Marra G., Radice R., Barnighausen T., Wood S.N. and McGovern M.E. (in press), A Simultaneous Equation Approach to Estimating HIV Prevalence with Non-Ignorable Missing Responses. *Journal of the American Statistical Association*.

Marra G. and Wyszynski K. (in press), Semi-Parametric Copula Sample Selection Models for Count Responses. *Computational Statistics and Data Analysis*.

McGovern M.E., Barnighausen T., Marra G. and Radice R. (2015), On the Assumption of Joint Normality in Selection Models: A Copula Approach Applied to Estimating HIV Prevalence. *Epidemiology*, 26(2), 229-237.

Radice R., Marra G. and Wojtys M. (2016), Copula Regression Spline Models for Binary Outcomes. *Statistics and Computing*, 26(5), 981-995.

Wojtys M. and Marra G. (submitted). Copula based generalized additive models with non-random sample selection.

See Also

SemiParBIVProbit, copulaReg, copulaSampleSel, SemiParTRIV

---

adjCov *Adjustment for the covariance matrix from a fitted SemiParBIVProbit/copulaReg/copulaSampleSel/SemiParTRIV model*

Description

adjCov can be used to adjust the covariance matrix of a fitted SemiParBIVProbit/copulaReg/copulaSampleSel/SemiParTRI object.

Usage

```
adjCov(x, id)
```

Arguments

x	A fitted SemiParBIVProbit/copulaReg/copulaSampleSel/SemiParTRIV object as produced by the respective fitting function.
id	Cluster identifier.

Details

This adjustment can be made when dealing with clustered data and the cluster structure is neglected when fitting the model. The basic idea is that the model is fitted as though observations were independent, and subsequently adjust the covariance matrix of the parameter estimates. Using the terminology of Liang and Zeger (1986), this would correspond to using an independence structure within the context of generalized estimating equations. The parameter estimators are still consistent but are inefficient as compared to a model which accounts for the correct cluster dependence structure. The covariance matrix of the independence estimators can be adjusted as described in Liang and Zeger (1986, Section 2).

Value

This function returns a fitted object which is identical to that supplied in adjCov but with adjusted covariance matrix.

WARNINGS

This correction may not be appropriate for models fitted using penalties.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

References

Liang K.-Y. and Zeger S. (1986), Longitudinal Data Analysis Using Generalized Linear Models. *Biometrika*, 73(1), 13-22.

See Also

SemiParBIVProbit-package, SemiParBIVProbit, summary.SemiParBIVProbit, SemiParTRIV, summary.SemiParTRIV, copulaReg, summary.copulaReg, copulaSampleSel, summary.copulaSampleSel

---

`adjCovSD` *Adjustment for the covariance matrix from a SemiParBIVProbit/copulaReg/copulaSampleSel/SemiParTRIV model fitted to complex survey data.*

Description

`adjCovSD` can be used to adjust the covariance matrix of a fitted `SemiParBIVProbit/copulaReg/copulaSampleSel/SemiParT` object.

Usage

```
adjCovSD(x, design)
```

Arguments

`x`	A fitted `SemiParBIVProbit/copulaReg/copulaSampleSel/SemiParTRIV` object as produced by the respective fitting function.
`design`	A `svydesign` object as produced by `svydesign()` from the `survey` package.

Details

This function has been extracted from the `survey` package and adapted to the class of this package's models. It computes the sandwich variance estimator for a copula model fitted to data from a complex sample survey (Lumley, 2004).

Value

This function returns a fitted object which is identical to that supplied in `adjCovSD` but with adjusted covariance matrix.

WARNINGS

This correction may not be appropriate for models fitted using penalties.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

References

Lumley T. (2004), Analysis of Complex Survey Samples. *Journal of Statistical Software*, 9(8), 1-19.

See Also

`SemiParBIVProbit-package`, `SemiParBIVProbit`, `summary.SemiParBIVProbit`, `SemiParTRIV`, `summary.SemiParTRIV`, `copulaReg`, `summary.copulaReg`, `copulaSampleSel`, `summary.copulaSampleSel`

AT *Average treatment effect of a binary/continuous/discrete endogenous variable*

Description

`AT` can be used to calculate the treatment effect of a binary/continuous/discrete endogenous predictor/treatment, with corresponding interval obtained using posterior simulation.

Usage

```
AT(x, nm.end, E = TRUE, treat = TRUE, type = "bivariate", ind = NULL,
   n.sim = 100, prob.lev = 0.05, length.out = NULL,
   hd.plot = FALSE, te.plot = FALSE,
   main = "Histogram and Kernel Density of Simulated Average Effects",
   xlab = "Simulated Average Effects", ...)
```

Arguments

`x`	A fitted `SemiParBIVProbit`/`copulaReg`/ object as produced by the respective fitting function.
`nm.end`	Name of the endogenous variable.

E	If TRUE then AT calculates the sample ATE. If FALSE then it calculates the sample AT for the treated individuals only.
treat	If TRUE then AT calculates the AT using the treated only. If FALSE then it calculates the effect on the control group. This only makes sense if E = FALSE.
type	This argument can take three values: "naive" (the effect is calculated ignoring the presence of observed and unobserved confounders), "univariate" (the effect is obtained from the univariate model which neglects the presence of unobserved confounders) and "bivariate" (the effect is obtained from the simultaneous model which accounts for observed and unobserved confounders).
ind	Binary logical variable. It can be used to calculate the AT for a subset of the data. Note that it does not make sense to use ind when some observations are excluded from the AT calculation (e.g., when using E = FALSE).
n.sim	Number of simulated coefficient vectors from the posterior distribution of the estimated model parameters. This is used when delta = FALSE. It may be increased if more precision is required.
prob.lev	Overall probability of the left and right tails of the AT distribution used for interval calculations.
length.out	Ddesired length of the sequence to be used when calculating the effect that a continuous/discrete treatment has on a binary outcome.
hd.plot	If TRUE then a plot of the histogram and kernel density estimate of the simulated average effects is produced. This can only be produced when when binary responses are used.
te.plot	For the case of continuous/discrete endogenous variable and binary outcome, if TRUE then a plot showing the treatment effects that the binary outcome is equal to 1 for each incremental value of the endogenous variable and respective intervals is produced.
main	Title for the plot.
xlab	Title for the x axis.
...	Other graphics parameters to pass on to plotting commands. These are used only when hd.plot = TRUE.

Details

AT measures the average difference in outcomes under treatment (the binary predictor or treatment assumes value 1) and under control (the binary treatment assumes value 0). Posterior simulation is used to obtain a confidence/credible interval. See the references below for details.

AT can also calculate the effect that a continuous/discrete endogenous variable has on a binary outcome. In this case the effect will depend on the unit increment chosen (as shown by the plot produced).

Value

res	It returns three values: lower confidence interval limit, estimated AT and upper interval limit.
prob.lev	Probability level used.

sim.AT	It returns a vector containing simulated values of the average treatment effect. This is used to calculate intervals.
Effects	For the case of continuous/discrete endogenous variable and binary outcome, it returns a matrix made up of three columns containing the effects for each incremental value in the endogenous variable and respective intervals.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

References

Marra G. and Radice R. (2011), Estimation of a Semiparametric Recursive Bivariate Probit in the Presence of Endogeneity. *Canadian Journal of Statistics*, 39(2), 259-279.

See Also

SemiParBIVProbit-package, SemiParBIVProbit, copulaReg

Examples

```
## see examples for SemiParBIVProbit and copulaReg
```

AT2 *Average treatment effect from a two-part model*

Description

AT2 can be used to calculate the sample average treatment effect from a two-part model, with corresponding interval obtained using posterior simulation.

Usage

```
AT2(x1, x2, index1, index2, n.sim = 100, prob.lev = 0.05,
    hd.plot = FALSE,
    main = "Histogram and Kernel Density of Simulated Average Effects",
    xlab = "Simulated Average Effects", ...)
```

Arguments

x1	A fitted SemiParBIVProbit object as produced by SemiParBIVProbit().
x2	A fitted SemiParBIVProbit object as produced by SemiParBIVProbit().
index1	This is useful to pick a particular individual.
index2	As above.

n.sim	Number of simulated coefficient vectors from the posterior distribution of the estimated model parameters. This is used when delta = FALSE. It may be increased if more precision is required.
prob.lev	Overall probability of the left and right tails of the AT distribution used for interval calculations.
hd.plot	If TRUE then a plot of the histogram and kernel density estimate of the simulated average effects is produced. This can only be produced when delta = FALSE.
main	Title for the plot.
xlab	Title for the x axis.
...	Other graphics parameters to pass on to plotting commands. These are used only when hd.plot = TRUE.

Details

AT measures the sample average effect from a two-part model when a binary response (associated with a continuous outcome) takes values 0 and 1. Posterior simulation is used to obtain a confidence/credible interval.

WARNINGS

This function is not suitable for SemiParBIVProbit().

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

See Also

SemiParBIVProbit-package, SemiParBIVProbit, summary.SemiParBIVProbit

Examples

```
## see examples for SemiParBIVProbit
```

BCDF *Internal Function*

Description

It evaluates the cdf of several copulae.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

bcont *Internal Function*

Description

This and other similar internal functions provide the log-likelihood, gradient and observed information matrix for penalized/unpenalized maximum likelihood optimization when copula models with continuous margins are employed.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

bdiscrcont *Internal Function*

Description

This and other similar internal functions provide the log-likelihood, gradient and observed information matrix for penalized/unpenalized maximum likelihood optimization when copula models with discrete and continuous margins are employed.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

bdiscrdiscr *Internal Function*

Description

This and other similar internal functions provide the log-likelihood, gradient and observed information matrix for penalized/unpenalized maximum likelihood optimization when copula models with discrete margins are employed.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

bprobgHs *Internal Function*

Description

It provides the log-likelihood, gradient and observed/Fisher information matrix for penalized/unpenalized maximum likelihood optimization when copula models with binary outcomes are employed.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

bprobgHsCont *Internal Function*

Description

It provides the log-likelihood, gradient and observed information matrix for penalized/unpenalized maximum likelihood optimization when copula models with binary and continuous margins are employed.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

bprobgHsContSS *Internal Function*

Description

It provides the log-likelihood, gradient and observed information matrix for penalized/unpenalized maximum likelihood optimization when copula sample selection models with continuous margins are employed.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

bprobgHsContUniv *Internal Function*

Description

It provides the log-likelihood, gradient and observed information matrix for penalized/unpenalized maximum likelihood optimization when fitting univariate models with discrete/continuous response.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

bprobgHsDiscr1 *Internal Function*

Description

It provides the log-likelihood, gradient and observed information matrix for penalized/unpenalized maximum likelihood optimization when copula models with binary and discrete margins are employed.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

bprobgHsDiscr1SS *Internal Function*

Description

It provides the log-likelihood, gradient and observed information matrix for penalized/unpenalized maximum likelihood optimization when copula sample selection models with discrete margins are employed.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

bprobgHsPO *Internal Function*

Description

It provides the log-likelihood, gradient and observed or expected information matrix for penalized/unpenalized maximum likelihood optimization when bivariate probit models with partial observability are employed.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

bprobgHsSS *Internal Function*

Description

It provides the log-likelihood, gradient and observed/Fisher information matrix for penalized/unpenalized maximum likelihood optimization when copula sample selection models with binary outcomes are employed.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

conv.check *Some convergence diagnostics*

Description

It takes a fitted model object and produces some diagnostic information about the fitting procedure.

Usage

```
conv.check(x)
```

Arguments

x	SemiParBIVProbit/copulaReg/copulaSampleSel/SemiParTRIV object.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

See Also

SemiParBIVProbit, SemiParBIVProbit, copulaReg, copulaSampleSel, SemiParTRIV

---

copgHs *Internal Function*

Description

This and other similar internal functions evaluate the first and second derivatives with respect to the margins and association parameter of several copulae.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

---

copulaReg *Semiparametric Copula Bivariate Models with Continuous/Discrete Margins*

Description

copulaReg fits flexible copula bivariate models with continuous/discrete margins with several types of covariate effects, copula distributions and marginal distributions.

Usage

```
copulaReg(formula, data = list(), weights = NULL, subset = NULL,
                BivD = "N", margins = c("N","N"),
                gamlssfit = FALSE, fp = FALSE, infl.fac = 1,
                rinit = 1, rmax = 100,
                iterlimsp = 50, tolsp = 1e-07,
                gc.l = FALSE, parscale, extra.regI = "t")
```

Arguments

formula	In the basic setup this will be a list of two formulas, one for equation 1 and the other for equation 2. s terms are used to specify smooth smooth functions of predictors. For the case of more than two equations see the example below and the documentation of SemiParBIVProbit() for more details. When one outcome is binary and the other continuous/discrete then the first equation MUST refer to the binary outcome whereas the second to the continuous/discrete one. When one outcome is discrete and the other continuous then the first equation MUST refer to the discrete one.
data	An optional data frame, list or environment containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which copulaReg is called.
weights	Optional vector of prior weights to be used in fitting.
subset	Optional vector specifying a subset of observations to be used in the fitting process.
margins	It indicates the distributions used for the two margins. Possible distributions are normal ("N"), log-normal ("LN"), Gumbel ("GU"), reverse Gumbel ("rGU"), logistic ("LO"), Weibull ("WEI"), inverse Gaussian ("iG"), gamma ("GA"), gamma with identity link for the location parameter ("GAi"), Dagum ("DAGUM"), Singh-Maddala ("SM"), beta ("BE"), Fisk ("FISK", also known as log-logistic distribution), Poisson ("PO"), zero truncated Poisson ("ZTP"), negative binomial - type I ("NBI"), negative binomial - type II ("NBII"), Poisson inverse Gaussian ("PIG"). When the first equation is binary then possible link functions are "probit", "logit" and "cloglog".
gamlssfit	If gamlssfit = TRUE then gamlss univariate models are also fitted. This is useful for obtaining starting values, for instance.
BivD	Type of bivariate error distribution employed. Possible choices are "N", "C0", "C90", "C180", "C270", "J0", "J90", "J180", "J270", "G0", "G90", "G180", "G270", "F", "AMH", "FGM" which stand for bivariate normal, Clayton, rotated Clayton (90 degrees), survival Clayton, rotated Clayton (270 degrees), Joe, rotated Joe (90 degrees), survival Joe, rotated Joe (270 degrees), Gumbel, rotated Gumbel (90 degrees), survival Gumbel, rotated Gumbel (270 degrees), Frank, Ali-Mikhail-Haq and Farlie-Gumbel-Morgenstern.
fp	If TRUE then a fully parametric model with unpenalised regression splines if fitted. See the example below.
infl.fac	Inflation factor for the model degrees of freedom in the approximate AIC. Smoother models can be obtained setting this parameter to a value greater than 1.
rinit	Starting trust region radius. The trust region radius is adjusted as the algorithm proceeds. See the documentation of trust for further details.
rmax	Maximum allowed trust region radius. This may be set very large. If set small, the algorithm traces a steepest descent path.
iterlimsp	A positive integer specifying the maximum number of loops to be performed before the smoothing parameter estimation step is terminated.
tolsp	Tolerance to use in judging convergence of the algorithm when automatic smoothing parameter estimation is used.

gc.l	This is relevant when working with big datasets. If TRUE then the garbage collector is called more often than it is usually done. This keeps the memory footprint down but it will slow down the routine.
parscale	The algorithm will operate as if optimizing objfun(x / parscale, ...) where parscale is a scalar. If missing then no rescaling is done. See the documentation of trust for more details.
extra.regI	If "t" then regularization as from trust is applied to the information matrix if needed. If different from "t" then extra regularization is applied via the options "pC" (pivoted Choleski - this will only work when the information matrix is semi-positive or positive definite) and "sED" (symmetric eigen-decomposition).

Details

The underlying algorithm is based on an extension of the procedure used for SemiParBIVProbit(). For more details see ?SemiParBIVProbit.

There are many continuous/discrete distributions and copula functions to choose from and we plan to include more options. Get in touch if you are interested in a particular distribution.

Value

The function returns an object of class copulaReg as described in copulaRegObject.

WARNINGS

Convergence failure may sometimes occur. Convergence can be checked using conv.check which provides some information about the score and information matrix associated with the fitted model. The former should be close to 0 and the latter positive definite. copulaReg() will produce some warnings if there is a convergence issue.

In such a situation, the user may use some extra regularisation (see extra.regI) and/or rescaling (see parscale). Using gamlssfit = TRUE is typically more effective than the first two options as this will provide better calibrated starting values as compared to those obtained from the default starting value procedure. The default option is, however, gamlssfit = FALSE only because it tends to be computationally cheaper and because the default starting value procedure has typically been found to do a satisfactory job in most cases. (The results obtained when using gamlssfit = FALSE and gamlssfit = TRUE could also be compared to check if starting values make any difference.)

The above suggestions may help, especially the latter option. However, the user should also consider re-specifying the model, and/or using a diferrent dependence structure and/or checking that the chosen marginal distributions fit the responses well. In our experience, we found that convergence failure typically occurs when the model has been misspecified and/or the sample size is low compared to the complexity of the model. Examples of misspecification include using a Clayton copula rotated by 90 degrees when a positive association between the margins is present instead, using marginal distributions that do not fit the responses, and employing a copula which does not accommodate the type and/or strength of the dependence between the margins (e.g., using AMH when the association between the margins is strong). It is also worth bearing in mind that the use of three parameter marginal distributions requires the data to be more informative than a situation in which two parameter distributions are used instead.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

References

Marra G. and Radice R. (submitted), Bivariate Copula Additive Models for Location, Scale and Shape.

See Also

adjCov, VuongClarke, plot.SemiParBIVProbit, SemiParBIVProbit-package, copulaRegObject, conv.check, summary.copulaReg, predict.SemiParBIVProbit

Examples

```
library(SemiParBIVProbit)

############
## EXAMPLE 1
## Generate data
## Correlation between the two equations 0.5 - Sample size 400

set.seed(0)

n <- 400

Sigma <- matrix(0.5, 2, 2); diag(Sigma) <- 1
u     <- rMVN(n, rep(0,2), Sigma)

x1 <- round(runif(n)); x2 <- runif(n); x3 <- runif(n)

f1   <- function(x) cos(pi*2*x) + sin(pi*x)
f2   <- function(x) x+exp(-30*(x-0.5)^2)

y1 <- -1.55 + 2*x1    + f1(x2) + u[,1]
y2 <- -0.25 - 1.25*x1 + f2(x2) + u[,2]

dataSim <- data.frame(y1, y2, x1, x2, x3)

resp.check(y1, "N")
resp.check(y2, "N")

eq.mu.1     <- y1 ~ x1 + s(x2) + s(x3)
eq.mu.2     <- y2 ~ x1 + s(x2) + s(x3)
eq.sigma2.1 <-    ~ 1
eq.sigma2.2 <-    ~ 1
eq.theta    <-    ~ x1

fl <- list(eq.mu.1, eq.mu.2, eq.sigma2.1, eq.sigma2.2, eq.theta)

# the order above is the one to follow when
```

```
# using more than two equations

out  <- copulaReg(fl, data = dataSim)
conv.check(out)
post.check(out)
summary(out)
AIC(out)
BIC(out)
jc.probs(out, 1.4, 2.3, intervals = TRUE)[1:4,]

## Not run:

############
## EXAMPLE 2
############
## Generate data with one endogenous binary variable
## and continuous outcome

set.seed(0)

n <- 1000

Sigma <- matrix(0.5, 2, 2); diag(Sigma) <- 1
u     <- rMVN(n, rep(0,2), Sigma)

cov   <- rMVN(n, rep(0,2), Sigma)
cov   <- pnorm(cov)
x1 <- round(cov[,1]); x2 <- cov[,2]

f1   <- function(x) cos(pi*2*x) + sin(pi*x)
f2   <- function(x) x+exp(-30*(x-0.5)^2)

y1 <- ifelse(-1.55 + 2*x1    + f1(x2) + u[,1] > 0, 1, 0)
y2 <-        -0.25 - 1.25*y1 + f2(x2) + u[,2]

dataSim <- data.frame(y1, y2, x1, x2)


## RECURSIVE Model

rc <- resp.check(y2, margin = "N", print.par = TRUE, loglik = TRUE)
AIC(rc); BIC(rc)

out <- copulaReg(list(y1 ~ x1 + x2,
                      y2 ~ y1 + x2),
                 data = dataSim, margins = c("probit","N"))
conv.check(out)
summary(out)
post.check(out)

## SEMIPARAMETRIC RECURSIVE Model

eq.mu.1   <- y1 ~ x1 + s(x2)
```

```
eq.mu.2   <- y2 ~ y1 + s(x2)
eq.sigma2 <-    ~ 1
eq.theta  <-    ~ 1

fl <- list(eq.mu.1, eq.mu.2, eq.sigma2, eq.theta)

out <- copulaReg(fl, data = dataSim,
                 margins = c("probit","N"), gamlssfit = TRUE)
conv.check(out)
summary(out)
post.check(out)
jc.probs(out, 1, 1.5, intervals = TRUE)[1:4,]
AT(out, nm.end = "y1")
AT(out, nm.end = "y1", type = "univariate")


#
#

#############
## EXAMPLE 3
#############
## Generate data with one endogenous continuous exposure
## and binary outcome

set.seed(0)

n <- 1000

Sigma <- matrix(0.5, 2, 2); diag(Sigma) <- 1
u     <- rMVN(n, rep(0,2), Sigma)

cov   <- rMVN(n, rep(0,2), Sigma)
cov   <- pnorm(cov)
x1 <- round(cov[,1]); x2 <- cov[,2]

f1   <- function(x) cos(pi*2*x) + sin(pi*x)
f2   <- function(x) x+exp(-30*(x-0.5)^2)

y1 <-        -0.25 - 2*x1    + f2(x2) + u[,2]
y2 <- ifelse(-0.25 - 0.25*y1 + f1(x2) + u[,1] > 0, 1, 0)

dataSim <- data.frame(y1, y2, x1, x2)

eq.mu.1   <- y2 ~ y1 + s(x2)
eq.mu.2   <- y1 ~ x1 + s(x2)
eq.sigma2 <-    ~ 1
eq.theta  <-    ~ 1

fl <- list(eq.mu.1, eq.mu.2, eq.sigma2, eq.theta)

out <- copulaReg(fl, data = dataSim,
                 margins = c("probit","N"))
```

```
conv.check(out)
summary(out)
post.check(out)
AT(out, nm.end = "y1")
AT(out, nm.end = "y1", type = "univariate")
RR(out, nm.end = "y1", rr.plot = TRUE)
RR(out, nm.end = "y1", type = "univariate")
OR(out, nm.end = "y1", or.plot = TRUE)
OR(out, nm.end = "y1", type = "univariate")

#
#

## End(Not run)
```

`copulaRegObject` *Fitted copulaReg object*

Description

A fitted semiparametric bivariate object returned by function `copulaReg` and of class "copulaReg" and "SemiParBIVProbit".

Value

`fit`	List of values and diagnostics extracted from the output of the algorithm.
`gam1`	Univariate fit for equation 1. See the documentation of `mgcv` for full details.
`gam2, gam3, ...`	Univariate fit for equation 2, equation 3, etc.
`coefficients`	The coefficients of the fitted model.
`weights`	Prior weights used during model fitting.
`sp`	Estimated smoothing parameters of the smooth components.
`iter.sp`	Number of iterations performed for the smoothing parameter estimation step.
`iter.if`	Number of iterations performed in the initial step of the algorithm.
`iter.inner`	Number of iterations performed within the smoothing parameter estimation step.
`theta`	Estimated dependence parameter linking the two equations.
`n`	Sample size.
`X1, X2, X3, ...`	Design matrices associated with the linear predictors.
`X1.d2, X2.d2, X3.d2, ...`	Number of columns of `X1`, `X2`, `X3`, etc.
`l.sp1, l.sp2, l.sp3, ...`	Number of smooth components in the equations.
`He`	Penalized -hessian/Fisher. This is the same as `HeSh` for unpenalized models.

HeSh	Unpenalized -hessian/Fisher.
Vb	Inverse of He. This corresponds to the Bayesian variance-covariance matrix used for confidence/credible interval calculations.
t.edf	Total degrees of freedom of the estimated bivariate model. It is calculated as sum(diag(F)).
edf1, edf2, edf3, ...	Degrees of freedom for the two equations of the fitted bivariate model (and for the third and fourth equations if present. They are calculated when splines are used.
bs.mgfit	List of values and diagnostics extracted from magic in mgcv.
conv.sp	If TRUE then the smoothing parameter selection algorithm stopped before reaching the maximum number of iterations allowed.
wor.c	Working model quantities.
eta1, eta2, eta3, ...	Estimated linear predictors for the two equations (as well as the third and fourth equations if present).
y1, y2	Responses of the two equations.
logLik	Value of the (unpenalized) log-likelihood evaluated at the (penalized or unpenalized) parameter estimates.
respvec	List containing response vectors.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

See Also

copulaReg, plot.SemiParBIVProbit, summary.copulaReg, predict.SemiParBIVProbit

---

copulaSampleSel *Semiparametric Copula Bivariate Regression Models with Non-Random Sample Selection*

Description

copulaSampleSel fit flexible copula bivariate sample selection models with several types of covariate effects, copula distributions and marginal distributions.

Usage

```
copulaSampleSel(formula, data = list(), weights = NULL, subset = NULL,
                BivD = "N", margins = c("probit", "N"), gamlssfit = FALSE,
                fp = FALSE, infl.fac = 1,
                rinit = 1, rmax = 100,
                iterlimsp = 50, tolsp = 1e-07,
                gc.l = FALSE, parscale, extra.regI = "t")
```

Arguments

formula	In the basic setup, this will be a list of two formulas, one for equation 1 and the other for equation 2. s terms are used to specify smooth smooth functions of predictors. For the case of more than two equations see the example below and the documentation of SemiParBIVProbit() for more details. Note that the first formula MUST refer to the selection equation.
data	An optional data frame, list or environment containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which copulaSampleSel is called.
weights	Optional vector of prior weights to be used in fitting.
subset	Optional vector specifying a subset of observations to be used in the fitting process.
margins	It indicates the distributions used for the two margins. The first is one of "probit", "logit", "cloglog" which refer to the link function of the first equation whose response is always assumed to be binary. The response for the second equation can be normal ("N"), log-normal ("LN"), Gumbel ("GU"), reverse Gumbel ("rGU"), logistic ("LO"), Weibull ("WEI"), inverse Gaussian ("iG"), gamma ("GA"), gamma with identity link for the location parameter ("GAi"), Dagum ("DAGUM"), Singh-Maddala ("SM"), beta ("BE"), Fisk ("FISK", also known as log-logistic distribution), Poisson ("PO"), zero truncated Poisson ("ZTP"), negative binomial - type I ("NBI"), negative binomial - type II ("NBII"), Poisson inverse Gaussian ("PIG").
gamlssfit	If gamlssfit = TRUE then a gamlss is also fitted for the outcome equation. This is may used for obtaining better calibrated starting values, for instance.
BivD	Type of bivariate error distribution employed. Possible choices are "N", "C0", "C90", "C180", "C270", "J0", "J90", "J180", "J270", "G0", "G90", "G180", "G270", "F", "AMH", "FGM" which stand for bivariate normal, Clayton, rotated Clayton (90 degrees), survival Clayton, rotated Clayton (270 degrees), Joe, rotated Joe (90 degrees), survival Joe, rotated Joe (270 degrees), Gumbel, rotated Gumbel (90 degrees), survival Gumbel, rotated Gumbel (270 degrees), Frank, Ali-Mikhail-Haq and Farlie-Gumbel-Morgenstern.
fp	If TRUE then a fully parametric model with unpenalised regression splines if fitted. See the example below.
infl.fac	Inflation factor for the model degrees of freedom in the approximate AIC. Smoother models can be obtained setting this parameter to a value greater than 1.
rinit	Starting trust region radius. The trust region radius is adjusted as the algorithm proceeds. See the documentation of trust for further details.
rmax	Maximum allowed trust region radius. This may be set very large. If set small, the algorithm traces a steepest descent path.
iterlimsp	A positive integer specifying the maximum number of loops to be performed before the smoothing parameter estimation step is terminated.
tolsp	Tolerance to use in judging convergence of the algorithm when automatic smoothing parameter estimation is used.

gc.l	This is relevant when working with big datasets. If TRUE then the garbage collector is called more often than it is usually done. This keeps the memory footprint down but it will slow down the routine.
parscale	The algorithm will operate as if optimizing objfun(x / parscale, ...) where parscale is a scalar. If missing then no rescaling is done. See the documentation of trust for more details.
extra.regI	If "t" then regularization as from trust is applied to the information matrix if needed. If different from "t" then extra regularization is applied via the options "pC" (pivoted Choleski - this will only work when the information matrix is semi-positive or positive definite) and "sED" (symmetric eigen-decomposition).

Details

The underlying algorithm is based on an extension of the procedure used for SemiParBIVProbit(). For more details see ?SemiParBIVProbit.

This function works as SemiParSampleSel() in SemiParSampleSel and has been included in SemiParBIVProbit (which already included sample selection models for binary outcomes) for the user's convenience (given several requests). copulaSampleSel() allows for several continuous/discrete distributions and link functions for the selection equation. SemiParSampleSel() allows for a probit link, normal, gamma and several discrete distributions.

If there are factors in the model, before fitting, the user has to ensure that the numbers of factor variables' levels in the selected sample are the same as those in the complete dataset. Even if a model could be fitted in such a situation, the model may produce fits which are not coherent with the nature of the correction sought. For more details see ?SemiParBIVProbit.

There are many continuous/discrete distributions and copula functions to choose from and we plan to include more options. Get in touch if you are interested in a particular distribution.

Value

The function returns an object of class copulaSampleSel as described in copulaSampleSelObject.

WARNINGS

Convergence failure may sometimes occur. Convergence can be checked using conv.check which provides some information about the score and information matrix associated with the fitted model. The former should be close to 0 and the latter positive definite. SemiParBIVProbit() will produce some warnings if there is a convergence issue.

In such a situation, the user may use some extra regularisation (see extra.regI) and/or rescaling (see parscale). Using gamlssfit = TRUE is typically more effective than the first two options as this may provide better calibrated starting values as compared to those obtained from the default starting value procedure. The default option is, however, gamlssfit = FALSE only because it tends to be computationally cheaper and because the default starting value procedure has typically been found to do a satisfactory job in most cases. (The results obtained when using gamlssfit = FALSE and gamlssfit = TRUE could also be compared to check if starting values make any difference.)

The above suggestions may help, especially the latter option. However, the user should also consider re-specifying the model and/or using a diferrent dependence structure and/or checking that the chosen marginal distributions are adequate. In our experience, we found that convergence failure

typically occurs when the model has been misspecified and/or the sample size/number of selected observations is low compared to the complexity of the model. Examples of misspecification include using a Clayton copula rotated by 90 degrees when a positive association between the margins is present instead, using marginal distributions that do not fit the responses or that contain many parameters, and employing a copula which does not accommodate the type and/or strength of the dependence between the margins (e.g., using AMH when the association between the margins is strong). It is also worth bearing in mind that the use of a three parameter marginal distribution requires the data to be more informative than a situation in which a two parameter distribution is used instead.

Extra attention is required when specifying the dependence parameter as a function of covariates. This is because in these situations the dependence parameter mainly models the association between the unobserved confounders in the two equations. Therefore, this option would make sense when it is believed that the strength of the association between the unobservables in the two equations varies based on some grouping factor or across geographical areas, for instance.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

References

Marra G. and Radice R. (2013), Estimation of a Regression Spline Sample Selection Model. *Computational Statistics and Data Analysis*, 61, 158-173.

Marra G. and Wyszynski K. (in press), Semi-Parametric Copula Sample Selection Models for Count Responses. *Computational Statistics and Data Analysis*.

Wojtys M. and Marra G. (submitted). Copula based generalized additive models with non-random sample selection.

See Also

copulaReg, SemiParBIVProbit, adjCov, VuongClarke, plot.SemiParBIVProbit, SemiParBIVProbit-package, copulaSampleSelObject, conv.check, summary.copulaSampleSel, predict.SemiParBIVProbit

Examples

```
library(SemiParBIVProbit)

######################################################################
## Generate data
## Correlation between the two equations and covariate correlation 0.5
## Sample size 2000
######################################################################

set.seed(0)

n <- 2000

rh <- 0.5
```

```
sigmau <- matrix(c(1, rh, rh, 1), 2, 2)
u      <- rMVN(n, rep(0,2), sigmau)

sigmac <- matrix(rh, 3, 3); diag(sigmac) <- 1
cov    <- rMVN(n, rep(0,3), sigmac)
cov    <- pnorm(cov)

bi <- round(cov[,1]); x1 <- cov[,2]; x2 <- cov[,3]

f11 <- function(x) -0.7*(4*x + 2.5*x^2 + 0.7*sin(5*x) + cos(7.5*x))
f12 <- function(x) -0.4*( -0.3 - 1.6*x + sin(5*x))
f21 <- function(x) 0.6*(exp(x) + sin(2.9*x))

ys <-  0.58 + 2.5*bi + f11(x1) + f12(x2) + u[, 1] > 0
y  <- -0.68 - 1.5*bi + f21(x1) +           u[, 2]
yo <- y*(ys > 0)

dataSim <- data.frame(ys, yo, bi, x1, x2)

## CLASSIC SAMPLE SELECTION MODEL
## the first equation MUST be the selection equation

resp.check(yo[ys > 0], "N")

out <- copulaSampleSel(list(ys ~ bi + x1 + x2,
                            yo ~ bi + x1),
                       data = dataSim)
conv.check(out)
post.check(out)
summary(out)

AIC(out)
BIC(out)


## SEMIPARAMETRIC SAMPLE SELECTION MODEL

## "cr" cubic regression spline basis      - "cs" shrinkage version of "cr"
## "tp" thin plate regression spline basis - "ts" shrinkage version of "tp"
## for smooths of one variable, "cr/cs" and "tp/ts" achieve similar results
## k is the basis dimension - default is 10
## m is the order of the penalty for the specific term - default is 2

out <- copulaSampleSel(list(ys ~ bi + s(x1, bs = "tp", k = 10, m = 2) + s(x2),
                            yo ~ bi + s(x1)),
                       data = dataSim)
conv.check(out)
post.check(out)
AIC(out)

## compare the two summary outputs
## the second output produces a summary of the results obtained when only
## the outcome equation is fitted, i.e. selection bias is not accounted for
```

```
summary(out)
summary(out$gam2)

## estimated smooth function plots
## the red line is the true curve
## the blue line is the naive curve not accounting for selection bias

x1.s <- sort(x1[dataSim$ys>0])
f21.x1 <- f21(x1.s)[order(x1.s)] - mean(f21(x1.s))

plot(out, eq = 2, ylim = c(-1, 0.8)); lines(x1.s, f21.x1, col = "red")
par(new = TRUE)
plot(out$gam2, se = FALSE, lty = 3, lwd = 2, ylim = c(-1, 0.8),
     ylab = "", rug = FALSE)


## Not run:

## SEMIPARAMETRIC SAMPLE SELECTION MODEL with association
## and dispersion parameters
## depending on covariates as well

eq.mu.1   <- ys ~ bi + s(x1) + s(x2)
eq.mu.2   <- yo ~ bi + s(x1)
eq.sigma2 <-    ~ bi
eq.theta  <-    ~ bi + x1

fl <- list(eq.mu.1, eq.mu.2, eq.sigma2, eq.theta)

out <- copulaSampleSel(fl, data = dataSim)
conv.check(out)
post.check(out)
summary(out)
out$sigma2
out$theta

jc.probs(out, 0, 0.3, intervals = TRUE)[1:4,]

outC0 <- copulaSampleSel(fl, data = dataSim, BivD = "C0")
conv.check(outC0)
post.check(outC0)
AIC(out, outC0)
BIC(out, outC0)

#
#

#########################################################
## example using Gumbel copula and normal-gamma margins
#########################################################

set.seed(1)
```

```
y  <- rgamma(n, shape = 1/2^2, exp(-0.68 - 1.5*bi + f21(x1) + u[, 2])*2^2)
yo <- y*(ys > 0)

dataSim <- data.frame(ys, yo, bi, x1, x2)


out <- copulaSampleSel(list(ys ~ bi + s(x1) + s(x2),
                            yo ~ bi + s(x1)),
                       data = dataSim, BivD = "G0",
                       margins = c("probit", "GA"))
conv.check(out)
post.check(out)
summary(out)


#
#

## End(Not run)
```

`copulaSampleSelObject` *Fitted copulaSampleSel object*

Description

A fitted semiparametric bivariate object returned by function `copulaSampleSel` and of class "copulaSampleSel" and "SemiParBIVProbit".

Value

`fit`	List of values and diagnostics extracted from the output of the algorithm.
`gam1`	Univariate fit for equation 1. See the documentation of `mgcv` for full details.
`gam2, gam3, ...`	Univariate fit for equation 2, equation 3, etc.
`coefficients`	The coefficients of the fitted model.
`weights`	Prior weights used during model fitting.
`sp`	Estimated smoothing parameters of the smooth components.
`iter.sp`	Number of iterations performed for the smoothing parameter estimation step.
`iter.if`	Number of iterations performed in the initial step of the algorithm.
`iter.inner`	Number of iterations performed within the smoothing parameter estimation step.
`theta`	Estimated dependence parameter linking the two equations.
`n`	Sample size.
`X1, X2, X3, ...`	Design matrices associated with the linear predictors.

`X1.d2, X2.d2, X3.d2, ...`
: Number of columns of `X1`, `X2`, `X3`, etc.

`l.sp1, l.sp2, l.sp3, ...`
: Number of smooth components in the equations.

`He`
: Penalized -hessian/Fisher. This is the same as `HeSh` for unpenalized models.

`HeSh`
: Unpenalized -hessian/Fisher.

`Vb`
: Inverse of `He`. This corresponds to the Bayesian variance-covariance matrix used for confidence/credible interval calculations.

`t.edf`
: Total degrees of freedom of the estimated bivariate model. It is calculated as `sum(diag(F))`.

`edf1, edf2, edf3, ...`
: Degrees of freedom for the two equations of the fitted bivariate model (and for the third and fourth equations if present. They are calculated when splines are used.

`bs.mgfit`
: List of values and diagnostics extracted from `magic` in `mgcv`.

`conv.sp`
: If `TRUE` then the smoothing parameter selection algorithm stopped before reaching the maximum number of iterations allowed.

`wor.c`
: Working model quantities.

`eta1, eta2, eta3, ...`
: Estimated linear predictors for the two equations (as well as the third and fourth equations if present).

`y1, y2`
: Responses of the two equations.

`logLik`
: Value of the (unpenalized) log-likelihood evaluated at the (penalized or unpenalized) parameter estimates.

`respvec`
: List containing response vectors.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

See Also

`copulaSampleSel`, `plot.SemiParBIVProbit`, `summary.copulaSampleSel`, `predict.SemiParBIVProbit`

---

distrHs *Internal Function*

Description

This and other similar internal functions evaluate the margins' derivatives needed in the likelihood function for the binary, discrete and continuous cases.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

eta.tr *Internal Function*

Description

This and other similar internal functions map certain key quantities into a feasible parameter space.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

g.tri *Internal Function*

Description

This and other similar internal functions calculate the score for trivariate binary models.

Author(s)

Author: Panagiota Filippou

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

gt.bpm *Gradient test*

Description

gt.bpm can be used to test the hypothesis of absence of endogeneity, correlated model equations/errors or non-random sample selection in binary bivariate probit models.

Usage

```
gt.bpm(x)
```

Arguments

x	A fitted SemiParBIVProbit object as produced by SemiParBIVProbit().

Details

The gradient test was first proposed by Terrell (2002) and it is based on classic likelihood theory. See Marra et al. (in press) for full details.

Value

It returns a numeric p-value corresponding to the null hypothesis that the correlation, θ, is equal to 0.

WARNINGS

This test's implementation is only valid for bivariate binary probit models with normal errors.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

References

Marra G., Radice R. and Filippou P. (in press), Regression Spline Bivariate Probit Models: A Practical Approach to Testing for Exogeneity. *Communications in Statistics - Simulation and Computation*.

Terrell G. (2002), The Gradient Statistic. *Computing Science and Statistics*, 34, 206-215.

See Also

SemiParBIVProbit

Examples

```
## see examples for SemiParBIVProbit
```

H.tri *Internal Function*

Description

This and other similar internal functions calculate the Hessian for trivariate binary models.

Author(s)

Author: Panagiota Filippou

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

`hiv` *HIV Zambian data*

Description

HIV Zambian data by region, together with polygons describing the regions' shapes.

Usage

```
data(hiv)
data(hiv.polys)
```

Format

`hiv` is a 6416 row data frame with the following columns

hivconsent binary variable indicating consent to test for HIV.

hiv binary variable indicating whether an individual is HIV positive.

age age in years.

education years of education.

region code identifying region, and matching `names(hiv.polys)`. It can take nine possible values: 1 central, 2 copperbelt, 3 eastern, 4 luapula, 5 lusaka, 6 northwestern, 7 northern, 8 southern, 9 western.

marital never married, currently married, formerly married.

std had a sexually transmitted disease.

highhiv had high risk sex.

condom used condom during last intercourse.

aidscare equal to 1 if would care for an HIV-infected relative.

knowsdiedofaids equal to 1 if know someone who died of HIV.

evertestedHIV equal to 1 if previously tested for HIV.

smoke smoker.

ethnicity bemba, lunda (luapula), lala, ushi, lamba, tonga, luvale, lunda (northwestern), mbunda, kaonde, lozi, chewa, nsenga, ngoni, mambwe, namwanga, tumbuka, other.

language English, Bemba, Lozi, Nyanja, Tonga, other.

interviewerID interviewer identifier.

sw survey weights.

`hiv.polys` contains the polygons defining the areas in the format described below.

Details

The data frame `hiv` relates to the regions whose boundaries are coded in `hiv.polys`. `hiv.polys[[i]]` is a 2 column matrix, containing the vertices of the polygons defining the boundary of the ith region. `names(hiv.polys)` matches `hiv$region` (order unimportant).

Source

The data have been produced as described in:

McGovern M.E., Barnighausen T., Marra G. and Radice R. (2015), On the Assumption of Joint Normality in Selection Models: A Copula Approach Applied to Estimating HIV Prevalence. *Epidemiology*, 26(2), 229-237.

References

Marra G., Radice R., Barnighausen T., Wood S.N. and McGovern M.E. (in press), A Simultaneous Equation Approach to Estimating HIV Prevalence with Non-Ignorable Missing Responses. *Journal of the American Statistical Association*.

Examples

```
## Not run:

#########################################################
#########################################################

library("SemiParBIVProbit")

data("hiv", package = "SemiParBIVProbit")
data("hiv.polys", package = "SemiParBIVProbit")

#########################################################
#########################################################
## The stuff below is useful if the user wishes to employ
## a Markov Random Field (MRF) smoother. It provides
## the instructions to set up polygons automatically
## and the dataset variable needed to fit a model with
## MRF.
#########################################################
#########################################################
#
# ## hiv.polys was already created and
# ## made available via the call
# ## data("hiv.polys", package = "SemiParBIVProbit")
# ## hiv.polys was created using the code below
#
# obj <- readRDS("ZMB_adm1.rds")
# ## RDS Zambian Level 1 file obtained from
# ## http://www.gadm.org.
#
# pol <- polys.setup(obj)
#
# hiv.polys <- pol$polys
# name <- cbind(names(hiv.polys), pol$names1)
# name
#
## last step was to create a factor variable with range
```

```
## range(name[,1]) where the numerical values were linked
## to the regions in name[, 2]. This is what was done in
## the hiv dataset; see hiv$region. Specifically,
## the procedure used was
##
# reg <- NULL
#
# for(i in 1:dim(hiv)[1]){
#
# if(hiv$region[i] == "Central")       reg[i] <- 1
# if(hiv$region[i] == "Copperbelt")    reg[i] <- 2
# if(hiv$region[i] == "Eastern")       reg[i] <- 3
# if(hiv$region[i] == "Luapula")       reg[i] <- 4
# if(hiv$region[i] == "Lusaka")        reg[i] <- 5
# if(hiv$region[i] == "North-Western") reg[i] <- 6
# if(hiv$region[i] == "Northern")      reg[i] <- 7
# if(hiv$region[i] == "Southern")      reg[i] <- 8
# if(hiv$region[i] == "Western")       reg[i] <- 9
#
# }
#
# hiv$region <- as.factor(reg)
#
#
############################################################
############################################################

xt <- list(polys = hiv.polys)

# neighbourhood structure info for MRF
# to use in model specification

############################################################
# Bivariate brobit model with non-random sample selection
############################################################

sel.eq <- hivconsent ~ s(age) + s(education) + s(wealth) +
                       s(region, bs = "mrf", xt = xt, k = 7) +
                       marital + std + age1sex_cat + highhiv +
                       partner + condom + aidscare +
                       knowsdiedofaids + evertestedHIV +
                       smoke + religion + ethnicity +
                       language + s(interviewerID, bs = "re")

out.eq <- hiv        ~ s(age) + s(education) + s(wealth) +
                       s(region, bs = "mrf", xt = xt, k = 7) +
                       marital + std + age1sex_cat + highhiv +
                       partner + condom + aidscare +
                       knowsdiedofaids + evertestedHIV +
                       smoke + religion + ethnicity +
                       language

theta.eq <-          ~ s(region, bs = "mrf", xt = xt, k = 7)
```

```
fl <- list(sel.eq, out.eq, theta.eq)

# the above model specification is fairly
# complex and it serves to illustrate the
# flexibility of the modelling approach

bss <- SemiParBIVProbit(fl, data = hiv, BivD = "J90",
                        Model = "BSS")

conv.check(bss)

set.seed(1)
sb <- summary(bss)
sb

plot(bss, eq = 1, seWithMean = TRUE, scheme = 1,
     scale = 0, pages = 1, jit = TRUE)

plot(bss, eq = 2, seWithMean = TRUE, scheme = 1,
     scale = 0, pages = 1, jit = TRUE)


prev(bss, sw = hiv$sw, type = "naive")

set.seed(1)
prev(bss, sw = hiv$sw, type = "univariate")

prev(bss, sw = hiv$sw)


lr <- length(hiv.polys)
prevBYreg  <- matrix(NA, lr, 2)
thetaBYreg <- NA

for(i in 1:lr) {
prevBYreg[i,1] <- prev(bss, sw = hiv$sw, ind = hiv$region==i,
                       type = "univariate")$res[2]
prevBYreg[i,2] <- prev(bss, sw = hiv$sw, ind = hiv$region==i)$res[2]
thetaBYreg[i]  <- bss$theta[hiv$region==i][1]
}


zlim <- range(prevBYreg)  # to establish a common prevalence range

par(mfrow = c(1, 3), cex.axis = 1.3)

polys.map(hiv.polys, prevBYreg[,1], zlim = zlim, lab = "",
          cex.lab = 1.5, cex.main = 1.5,
          main = "HIV - Imputation Model")

polys.map(hiv.polys, prevBYreg[,2], zlim = zlim, cex.main = 1.5,
          main = "HIV - Selection Model")
```

```
polys.map(hiv.polys, thetaBYreg, rev.col = FALSE, cex.main = 1.7,
          main = expression(paste("Copula parameter (",hat(theta),")")))

sb$CItheta[1,]

## End(Not run)

#
```

---

jc.probs *Joint or conditional probabilities from a fitted bivariate model*

Description

jc.probs can be used to calculate the joint or conditional probabilities from a fitted bivariate model with intervals obtained using posterior simulation.

Usage

```
jc.probs(x, y1, y2, newdata, type = "bivariate", cond = 0,
         intervals = FALSE, n.sim = 100, prob.lev = 0.05)
```

Arguments

x	A fitted SemiParBIVProbit/copulaReg/copulaSampleSel object as produced by the respective fitting function.
y1	Value of response for first margin.
y2	Value of response for second margin.
newdata	A data frame or list containing the values of the model covariates at which predictions are required. If not provided then predictions corresponding to the original data are returned. When newdata is provided, it should contain all the variables needed for prediction.
type	This argument can take two: "bivariate" (the probabilities are calculated from the fitted bivariate model) and "independence" (the calculation is done from univariate fits).
cond	There are three possible values: 0 (joint probabilities are delivered), 1 (conditional probabilities are delivered and conditioning is with the respect to the first margin), 2 (as before but conditioning is with the respect to the second margin).
intervals	If TRUE then intervals for the probabilities are also produced.
n.sim	Number of simulated coefficient vectors from the posterior distribution of the estimated model parameters. This is used for interval calculations.
prob.lev	Overall probability of the left and right tails of the probabilities' distributions used for interval calculations.

Details

This function calculates joint or conditional probabilities from a fitted bivariate model or a model assuming independence, with intervals obtained using posterior simulation.

Value

res	It returns three values: estimated probabilities (p12), with lower and upper interval limits (CIpr) if intervals = TRUE, and p1 and p2 (the marginal probabilities).

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

See Also

SemiParBIVProbit-package, SemiParBIVProbit, copulaReg, copulaSampleSel

Examples

```
## see examples for SemiParBIVProbit, copulaReg and copulaSampleSel
```

LM.bpm *Lagrange Multiplier Test (Score Test)*

Description

Before fitting a bivariate probit model, LM.bpm can be used to test the hypothesis of absence of endogeneity, correlated model equations/errors or non-random sample selection.

Usage

```
LM.bpm(formula, data = list(), weights = NULL, subset = NULL, Model,
       hess = TRUE)
```

Arguments

formula	A list of two formulas, one for equation 1 and the other for equation 2. s terms are used to specify smooth smooth functions of predictors. Note that if Model = "BSS" then the first formula MUST refer to the selection equation.
data	An optional data frame, list or environment containing the variables in the model. If not found in data, the variables are taken from environment(formula).
weights	Optional vector of prior weights to be used in fitting.
subset	Optional vector specifying a subset of observations to be used in the fitting process.

`Model`	It indicates the type of model to be used in the analysis. Possible values are "B" (bivariate model) and "BSS" (bivariate model with sample selection). The two marginal equations have probit links.
`hess`	If `FALSE` then the expected (rather than observed) information matrix is employed.

Details

This Lagrange multiplier test (also known as score test) is used here for testing the null hypothesis that θ is equal to 0 (i.e. no endogeneity, non-random sample selection or correlated model equations/errors, depending on the model being fitted). Its main advantage is that it does not require an estimate of the model parameter vector under the alternative hypothesis. Asymptotically, it takes a Chi-squared distribution with one degree of freedom. Full details can be found in Marra et al. (2014) and Marra et al. (in press).

Value

It returns a numeric p-value corresponding to the null hypothesis that the correlation, θ, is equal to 0.

WARNINGS

This test's implementation is ONLY valid for bivariate binary probit models with normal errors.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

References

Marra G., Radice R. and Filippou P. (in press), Regression Spline Bivariate Probit Models: A Practical Approach to Testing for Exogeneity. *Communications in Statistics - Simulation and Computation*.

Marra G., Radice R. and Missiroli S. (2014), Testing the Hypothesis of Absence of Unobserved Confounding in Semiparametric Bivariate Probit Models. *Computational Statistics*, 29(3-4), 715-741.

See Also

`SemiParBIVProbit`

Examples

```
## see examples for SemiParBIVProbit
```

logLik.SemiParBIVProbit *Extract the log likelihood for a fitted copula model*

Description

It extracts the log-likelihood for a fitted SemiParBIVProbit/copulaReg/copulaSampleSel/SemiParTRIV model.

Usage

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

Arguments

object	A fitted SemiParBIVProbit/copulaReg/copulaSampleSel/SemiParTRIV object.
...	Un-used for this function.

Details

Modification of the classic logLik which accounts for the estimated degrees of freedom used in SemiParBIVProbit/copulaReg/copulaSampleSel/SemiParTRIV objects. This function is provided so that information criteria work correctly by using the correct number of degrees of freedom.

Value

Standard logLik object.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

See Also

AIC, BIC

Examples

```
## see examples for SemiParBIVProbit
```

mb *Nonparametric (worst-case and IV) Manski's bounds*

Description

mb can be used to calculate the (worst-case and IV) Manski's bounds and confidence interval covering the true effect of interest with a fixed probability.

Usage

```
mb(treat, outc, IV = NULL, Model, B = 100, sig.lev = 0.05)
```

Arguments

treat	Binary treatment/selection variable.
outc	Binary outcome variable.
IV	An instrumental binary variable can be used if available.
Model	Possible values are "B" (model with endogenous variable) and "BSS" (model with non-random sample selection).
B	Number of bootstrap replicates. This is used to obtain some components needed for confidence interval calculations.
sig.lev	Significance level.

Details

Based on Manski (1990), this function returns the nonparametric lower and upper (worst-case) Manski's bounds for the average treatment effect (ATE) when `Model = "B"` or prevalence when `Model = "BSS"`. When an IV is employed the function returns IV Manski bounds.

For comparison, it also returns the estimated effect assuming random assignment (i.e., the treatment received or selection relies on the assumption of ignorable observed and unobserved selection). Note that this is equivalent to what provided by AT or prev when `type = "naive"`, and is different from what obtained by AT or prev when `type = "univariate"` as observed confounders are accounted for and the assumption here is of ignorable unobserved selection.

A confidence interval covering the true ATE/prevalence with a fixed probability is also provided. This is based on the approach described in Imbens and Manski (2004). NOTE that this interval is typically very close (if not identical) to the lower and upper bounds.

The ATE can be at most 1 (or 100 in percentage) and the worst-case Manski's bounds have width 1. This means that 0 is always included within the possibilites of these bounds. Nevertheless, this may be useful to check whether the effect from a bivariate recursive model is included within the possibilites of the bounds.

When estimating a prevalance the worst-case Manski's bounds have width equal to the non-response probability, which provides a measure of the uncertainty about the prevalence caused by non-response. Again, this may be useful to check whether the prevalence from a bivariate non-random sample selection model is included within the possibilites of the bounds.

See SemiParBIVProbit for some examples.

Value

LB, UP	Lower and upper bounds for the true effect of interest.
CI	Confidence interval covering the true effect of interest with a fixed probability.
ate.ra	Estimated effect of interest assuming random assignment.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

References

Manski C.F. (1990), Nonparametric Bounds on Treatment Effects. *American Economic Review, Papers and Proceedings*, 80(2), 319-323.

Imbens G.W. and Manski C.F (2004), Confidence Intervals for Partially Identified Parameters. *Econometrica*, 72(6), 1845-1857.

See Also

SemiParBIVProbit

Examples

```
## see examples for SemiParBIVProbit
```

---

meps *MEPS data*

Description

2008 MEPS data.

Usage

```
data(meps)
```

Format

meps is a 18592 row data frame with the following columns

bmi body mass index.

age age in years.

gender equal to 1 if male.

race levels: 2 white, 3 black, 4 native American, 5 others.

education years of education.

health levels: 5 excellent, 6 very good, 7 good, 8 fair, 9 poor.

limitation equal to 1 if health limits physical activity.

region levels: 2 northeast, 3 mid-west, 4 south, 5 west.

private equal to 1 if individual has private health insurance.

visits.hosp equal to 1 if at least one visit to hospital outpatient departments.

diabetes equal to 1 if diabetic.

hypertension equal to 1 if hypertensive.

hyperlipidemia equal to 1 if hyperlipidemic.

income income (000's).

Source

The data have been obtained from http://www.meps.ahrq.gov/.

References

Radice R., Marra G. and Wojtys M. (2016), Copula Regression Spline Models for Binary Outcomes. *Statistics and Computing*, 26(5), 981-995.

Examples

```
## Not run:

#################################################
#################################################

library("SemiParBIVProbit")
data("meps", package = "SemiParBIVProbit")

#################################################
# Bivariate brobit models with endogenous treatment
#################################################

treat.eq <- private ~ s(bmi) + s(income) + s(age) + s(education) +
                      as.factor(health) + as.factor(race) +
                      as.factor(limitation) + as.factor(region) +
                      gender  + hypertension + hyperlipidemia + diabetes
```

```
out.eq <- visits.hosp ~ private + s(bmi) + s(income) + s(age) +
                        s(education) + as.factor(health) +
                        as.factor(race) + as.factor(limitation) +
                        as.factor(region) + gender + hypertension +
                        hyperlipidemia + diabetes

f.list <- list(treat.eq, out.eq)
bpN    <- SemiParBIVProbit(f.list, data = meps)
bpF    <- SemiParBIVProbit(f.list, data = meps, BivD = "F")
bpC0   <- SemiParBIVProbit(f.list, data = meps, BivD = "C0")
bpC180 <- SemiParBIVProbit(f.list, data = meps, BivD = "C180")
bpJ0   <- SemiParBIVProbit(f.list, data = meps, BivD = "J0")
bpJ180 <- SemiParBIVProbit(f.list, data = meps, BivD = "J180")
bpG0   <- SemiParBIVProbit(f.list, data = meps, BivD = "G0")
bpG180 <- SemiParBIVProbit(f.list, data = meps, BivD = "G180")

conv.check(bpJ0)

AIC(bpN, bpF, bpC0, bpC180, bpJ0, bpJ180, bpG0, bpG180)

set.seed(1)
summary(bpJ0, cm.plot = TRUE, cex.axis = 1.6,
        cex.lab = 1.6, cex.main = 1.7)

#dev.copy(postscript, "contplot.eps")
#dev.off()

par(mfrow = c(2, 2), mar = c(4.5, 4.5, 2, 2),
    cex.axis = 1.6, cex.lab = 1.6)
plot(bpJ0, eq = 1, seWithMean = TRUE, scale = 0, shade = TRUE,
     pages = 1, jit = TRUE)

#dev.copy(postscript, "spline1.eps")
#dev.off()

par(mfrow = c(2, 2), mar = c(4.5, 4.5, 2, 2),
    cex.axis = 1.6, cex.lab = 1.6)
plot(bpJ0, eq = 2, seWithMean = TRUE, scale = 0, shade = TRUE,
     pages = 1, jit = TRUE)

#dev.copy(postscript, "spline2.eps")
#dev.off()

set.seed(1)
AT(bpJ0, nm.end = "private", hd.plot = TRUE, cex.axis = 1.5,
   cex.lab = 1.5, cex.main = 1.6)

#dev.copy(postscript, "hd.plotAT.eps")
#dev.off()

AT(bpJ0, nm.end = "private", type = "univariate")

AT(bpJ0, nm.end = "private", type = "naive")
```

```
## End(Not run)

#
```

---

numgh *Internal Function*

Description

This and other similar internal functions calculate numerical derivatives.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

OR *Causal odds ratio of a binary/continuous/discrete endogenous variable*

Description

OR can be used to calculate the causal odds ratio of a binary/continuous/discrete endogenous predictor/treatment, with corresponding interval obtained using posterior simulation.

Usage

```
OR(x, nm.end, E = TRUE, treat = TRUE, type = "bivariate", ind = NULL,
   n.sim = 100, prob.lev = 0.05, length.out = NULL, hd.plot = FALSE,
   or.plot = FALSE,
   main = "Histogram and Kernel Density of Simulated Odds Ratios",
   xlab = "Simulated Odds Ratios", ...)
```

Arguments

x	A fitted SemiParBIVProbit/copulaReg object.
nm.end	Name of the endogenous variable.
E	If TRUE then OR calculates the sample OR. If FALSE then it calculates the sample OR for the treated individuals only.
treat	If TRUE then OR calculates the OR using the treated only. If FALSE then it calculates the ratio using the control group. This only makes sense if E = FALSE.

type	This argument can take three values: "naive" (the effect is calculated ignoring the presence of observed and unobserved confounders), "univariate" (the effect is obtained from the univariate model which neglects the presence of unobserved confounders) and "bivariate" (the effect is obtained from the bivariate model which accounts for observed and unobserved confounders).
ind	Binary logical variable. It can be used to calculate the OR for a subset of the data. Note that it does not make sense to use ind when some observations are excluded from the OR calculation (e.g., when using E = FALSE).
n.sim	Number of simulated coefficient vectors from the posterior distribution of the estimated model parameters. This is used when delta = FALSE. It may be increased if more precision is required.
prob.lev	Overall probability of the left and right tails of the OR distribution used for interval calculations.
length.out	Ddesired length of the sequence to be used when calculating the effect that a continuous/discrete treatment has on a binary outcome.
hd.plot	If TRUE then a plot of the histogram and kernel density estimate of the simulated odds ratios is produced. This can only be produced when binary responses are used.
or.plot	For the case of continuous/discrete endogenous variable and binary outcome, if TRUE then a plot (on the log scale) showing the odd ratios that the binary outcome is equal to 1 for each incremental value of the endogenous variable and respective intervals is produced.
main	Title for the plot.
xlab	Title for the x axis.
...	Other graphics parameters to pass on to plotting commands. These are used only when hd.plot = TRUE.

Details

OR calculates the causal odds ratio for a binary/continuous/discrete treatment. Posterior simulation is used to obtain a confidence/credible interval.

Value

prob.lev	Probability level used.
sim.OR	It returns a vector containing simulated values of the average OR. This is used to calculate intervals.
Ratios	For the case of continuous/discrete endogenous treatment and binary outcome, it returns a matrix made up of three columns containing the odds ratios for each incremental value in the endogenous variable and respective intervals.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

See Also

SemiParBIVProbit-package, SemiParBIVProbit, copulaReg

Examples

```
## see examples for SemiParBIVProbit and copulaReg
```

---

pen *Internal Function*

Description

It provides an overall penalty matrix in a format suitable for estimation conditional on smoothing parameters.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

---

plot.SemiParBIVProbit *Plotting function*

Description

It takes a fitted SemiParBIVProbit/copulaReg/copulaSampleSel/SemiParTRIV object produced by SemiParBIVProbit(), copulaReg(), copulaSampleSel(), SemiParTRIV() and plots the estimated smooth functions on the scale of the linear predictors. This function is a wrapper for plot.gam() in mgcv. Please see the documentation of plot.gam() for full details.

Usage

```
## S3 method for class 'SemiParBIVProbit'
plot(x, eq, ...)
```

Arguments

x	A fitted SemiParBIVProbit/copulaReg/copulaSampleSel/SemiParTRIV() object.
eq	The equation from which smooth terms should be considered for printing.
...	Other graphics parameters to pass on to plotting commands, as described for plot.gam() in mgcv.

Details

This function produces plots showing the smooth terms of a fitted semiparametric bivariate probit model. In the case of 1-D smooths, the x axis of each plot is labelled using the name of the regressor, while the y axis is labelled as `s(regr, edf)` where `regr` is the regressor's name, and `edf` the effective degrees of freedom of the smooth. For 2-D smooths, perspective plots are produced with the x axes labelled with the first and second variable names and the y axis is labelled as `s(var1, var2, edf)`, which indicates the variables of which the term is a function and the `edf` for the term.

If `seWithMean = TRUE` then the intervals include the uncertainty about the overall mean. Note that the smooths are still shown centred. The theoretical arguments and simulation study of Marra and Wood (2012) suggest that `seWithMean = TRUE` results in intervals with close to nominal frequentist coverage probabilities.

Value

The function generates plots.

WARNING

The function can not deal with smooths of more than 2 variables.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

References

Marra G. and Wood S.N. (2012), Coverage Properties of Confidence Intervals for Generalized Additive Model Components. *Scandinavian Journal of Statistics*, 39(1), 53-74.

See Also

`SemiParBIVProbit`, `copulaReg`, `copulaSampleSel`, `SemiParTRIV`, `predict.SemiParBIVProbit`

Examples

```
## see examples for SemiParBIVProbit
```

---

polys.map *Geographic map with regions defined as polygons*

Description

This function produces a map with geographic regions defined by polygons. It is essentially the same function as `polys.plot()` in `mgcv` but with added arguments `zlim` and `rev.col` and a wider set of choices for `scheme`.

Usage

```
polys.map(lm, z, scheme = "gray", lab = "", zlim, rev.col = TRUE, ...)
```

Arguments

lm	Named list of matrices where each matrix has two columns. The matrix rows each define the vertex of a boundary polygon.
z	A vector of values associated with each area (item) of lm.
scheme	Possible values are "heat", "terrain", "topo", "cm" and "gray", indicating how to fill the polygons in accordance with the value of z.
lab	label for plot.
zlim	If missing then the range of z will be chosen using pretty(z) otherwise the range provided will be used.
rev.col	If FALSE then coloring scheme is not reversed.
...	other arguments to pass to plot.

Details

See help file of polys.plot in mgcv.

Value

It produces a plot.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

---

polys.setup *Set up geographic polygons*

Description

This function creates geographic polygons in a format suitable for smoothing.

Usage

```
polys.setup(object)
```

Arguments

object	An RDS file object as extracted from http://www.gadm.org.

Value

It produces a list with polygons (polys), and various names (names0, names1 - first level of aggregation, names2 - second level of aggregation).

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

Thanks to Guy Harling for suggesting the implementation of this function.

Examples

```
?hiv
```

---

post.check *Diagnostic plots for discrete/continuous response margin*

Description

It produces diagnostic plots based on (randomised) quantile residuals.

Usage

```
post.check(x, main = "Histogram and Density Estimate of Residuals",
           main2 = "Histogram and Density Estimate of Residuals",
           xlab = "Quantile Residuals", xlab2 = "Quantile Residuals",
           test = FALSE, ...)
```

Arguments

x	A fitted copulaReg/copulaSampleSel object.
main	Title for the plot.
main2	Title for the plot in the second row. This comes into play only when fitting models with two non-binary margins.
xlab	Title for the x axis.
xlab2	Title for the x axis in the second row. As above.
test	If TRUE then the shapiro test for normality is performed on normalised quantile residuals.
...	Other graphics parameters to pass on to plotting commands.

Details

If the model fits the response well then the plots should look normally distributed. When fitting models with discrete and/or continuous margins, four plots will be produced. In this case, the arguments `main2` and `xlab2` come into play and allow for different labelling across the plots.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

See Also

copulaReg, copulaSampleSel

---

predict.SemiParBIVProbit

Prediction function

Description

It takes a fitted SemiParBIVProbit/copulaReg/copulaSampleSel/SemiParTRIV object and, for each equation, produces predictions for a new set of values of the model covariates or the original values used for the model fit. Standard errors of predictions can be produced and are based on the posterior distribution of the model coefficients. This function is a wrapper for `predict.gam()` in `mgcv`. Please see the documentation of `predict.gam()` for full details.

Usage

```
## S3 method for class 'SemiParBIVProbit'
predict(object, eq, ...)
```

Arguments

`object`	A fitted codeSemiParBIVProbit/copulaReg/copulaSampleSel/SemiParTRIV object.
`eq`	The equation to be considered for prediction.
`...`	Other arguments as in `predict.gam()` in `mgcv`.

WARNINGS

When `type = "response"` (which gives predictions on the scale of the response variable). For the case of continuous responses this function will NOT produce correct predictions for the outcome variable (except for the Gaussian case). This is because for all distributions (except the Gaussian) implemented in this package the distribution parameters determine the mean and variance through functions of them.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

See Also

SemiParBIVProbit, copulaReg, copulaSampleSel, SemiParTRIV, plot.SemiParBIVProbit

---

prev *Estimated overall prevalence from sample selection model*

Description

prev can be used to calculate the overall estimated prevalence from a sample selection model with binay outcome, with corresponding interval obtained using the delta method or posterior simulation.

Usage

```
prev(x, sw = NULL, type = "simultaneous", ind = NULL, delta = FALSE,
     n.sim = 100, prob.lev = 0.05, hd.plot = FALSE,
     main = "Histogram and Kernel Density of Simulated Prevalences",
     xlab = "Simulated Prevalences", ...)
```

Arguments

x	A fitted SemiParBIVProbit/SemiParTRIV object.
sw	Survey weights.
type	This argument can take three values: "naive" (the prevalence is calculated ignoring the presence of observed and unobserved confounders), "univariate" (the prevalence is obtained from the univariate probit/single imputation model which neglects the presence of unobserved confounders) and "simultaneous" (the prevalence is obtained from the bivariate/trivariate model which accounts for observed and unobserved confounders).
ind	Binary logical variable. It can be used to calculate the prevalence for a subset of the data.
delta	If TRUE then the delta method is used for confidence interval calculations, otherwise Bayesian posterior simulation is employed.
n.sim	Number of simulated coefficient vectors from the posterior distribution of the estimated model parameters. This is used when delta = FALSE. It may be increased if more precision is required.
prob.lev	Overall probability of the left and right tails of the prevalence distribution used for interval calculations.
hd.plot	If TRUE then a plot of the histogram and kernel density estimate of the simulated prevalences is produced. This can only be produced when delta = FALSE.

main	Title for the plot.
xlab	Title for the x axis.
...	Other graphics parameters to pass on to plotting commands. These are used only when hd.plot = TRUE.

Details

prev estimates the overall prevalence of a disease (e.g., HIV) when there are missing values that are not at random. An interval for the estimated prevalence can be obtained using the delta method or posterior simulation.

Value

res	It returns three values: lower confidence interval limit, estimated prevalence and upper confidence interval limit.
prob.lev	Probability level used.
sim.prev	If delta = FALSE then it returns a vector containing simulated values of the prevalence. This is used to calculate an interval.

Author(s)

Authors: Giampiero Marra, Rosalba Radice, Guy Harling, Mark E McGovern

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

References

McGovern M.E., Barnighausen T., Marra G. and Radice R. (2015), On the Assumption of Joint Normality in Selection Models: A Copula Approach Applied to Estimating HIV Prevalence. *Epidemiology*, 26(2), 229-237.

Marra G., Radice R., Barnighausen T., Wood S.N. and McGovern M.E. (in press), A Simultaneous Equation Approach to Estimating HIV Prevalence with Non-Ignorable Missing Responses. *Journal of the American Statistical Association*.

See Also

SemiParBIVProbit-package, SemiParBIVProbit, SemiParTRIV

Examples

```
## see examples for SemiParBIVProbit and SemiParTRIV
```

print.AT *Print an AT object*

Description

The print method for an AT object.

Usage

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

Arguments

x	AT object produced by AT().
...	Other arguments.

Details

print.AT prints the lower confidence interval limit, estimated AT and upper confidence interval limit.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

See Also

AT

print.AT2 *Print an AT2 object*

Description

The print method for an AT2 object.

Usage

```
## S3 method for class 'AT2'
print(x, ...)
```

Arguments

x	AT2 object produced by AT2().
...	Other arguments.

Details

print.AT2 prints the lower confidence interval limit, estimated AT and upper confidence interval limit.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

See Also

AT2

---

print.copulaReg *Print a copulaReg object*

Description

The print method for a copulaReg object.

Usage

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

Arguments

x	copulaReg object produced by copulaReg().
...	Other arguments.

Details

print.copulaReg prints out the family, model equations, total number of observations, estimated association coefficient, etc for the penalized or unpenalized model.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

See Also

copulaReg

---

print.copulaSampleSel *Print a copulaSampleSel object*

Description

The print method for a copulaSampleSel object.

Usage

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

Arguments

x	copulaSampleSel object produced by copulaSampleSel().
...	Other arguments.

Details

print.copulaSampleSel prints out the family, model equations, total number of observations, estimated association coefficient, etc for the penalized or unpenalized model.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

See Also

copulaSampleSel

print.mb *Print an mb object*

Description

The print method for an mb object.

Usage

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

Arguments

x	mb object produced by mb().
...	Other arguments.

Details

print.mb prints the lower and upper bounds, confidence interval, and effect assuming random assignment.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

See Also

mb

print.OR *Print an OR object*

Description

The print method for an OR object.

Usage

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

Arguments

x	OR object produced by OR().
...	Other arguments.

Details

print.OR prints the lower confidence interval limit, estimated OR and upper confidence interval limit.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

See Also

OR

---

print.prev *Print an prev object*

Description

The print method for an prev object.

Usage

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

Arguments

x	prev object produced by prev().
...	Other arguments.

Details

`print.prev` prints the lower interval limit, estimated prevalence and upper interval limit.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

See Also

`prev`

`print.RR` *Print an RR object*

Description

The print method for an `RR` object.

Usage

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

Arguments

`x`	`RR` object produced by `RR()`.
`...`	Other arguments.

Details

`print.RR` prints the lower confidence interval limit, estimated RR and upper confidence interval limit.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

See Also

`RR`

print.SemiParBIVProbit *Print a SemiParBIVProbit object*

Description

The print method for a SemiParBIVProbit object.

Usage

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

Arguments

x	SemiParBIVProbit object produced by SemiParBIVProbit().
...	Other arguments.

Details

print.SemiParBIVProbit prints out the family, model equations, total number of observations, estimated association coefficient and total effective degrees of freedom for the penalized or unpenalized model.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

See Also

SemiParBIVProbit

print.SemiParTRIV *Print a SemiParTRIV object*

Description

The print method for a SemiParTRIV object.

Usage

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

Arguments

x	SemiParTRIV object produced by SemiParTRIV().
...	Other arguments.

Details

print.SemiParTRIV prints out the family, model equations, total number of observations, estimated association coefficient and total effective degrees of freedom for the penalized or unpenalized model.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

See Also

SemiParTRIV

---

probm *Internal Function*

Description

Internal fitting function.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

regH *Internal Function*

Description

It applies one of two regularisations on the information matrix if desired. These are based on the Cholesky and eigen decompositions.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

resp.check *Plots for response variable*

Description

It produces a histogram of the response along with the estimated density from the assumed distribution as well as a normal Q-Q plot for the (randomised) normalised quantile response. It also provides the log-likelihood for AIC calculation, for instance.

Usage

```
resp.check(y, margin = "N", main = "Histogram and Density of Response",
           xlab = "Response", print.par = FALSE, plots = TRUE,
           loglik = FALSE, os = FALSE, test = FALSE, i.f = FALSE, ...)
```

Arguments

y	Response.
margin	The distributions allowed are: normal ("N"), log-normal ("LN"), Gumbel ("GU"), reverse Gumbel ("rGU"), logistic ("LO"), Weibull ("WEI"), inverse Gaussian ("iG"), gamma ("GA"), Dagum ("DAGUM"), Singh-Maddala ("SM"), beta ("BE"), Fisk ("FISK"), Poisson ("PO"), zero truncated Poisson ("ZTP"), negative binomial - type I ("NBI"), negative binomial - type II ("NBII"), Poisson inverse Gaussian ("PIG").
main	Title for the plot.
xlab	Title for the x axis.
print.par	If TRUE then the estimated parameters used to construct the plots are returned.
plots	If FALSE then no plots are produced and only parameter estimates returned.
loglik	If TRUE then it returns the logLik.

os	If TRUE then the estimated parameters are returned on the original scale.
test	If TRUE then the shapiro test for normality is performed on normalised quantile residuals.
i.f	Internal fitting option. This is not for user purposes.
...	Other graphics parameters to pass on to plotting commands.

Details

Prior to fitting a model with discrete and/or continuous margins, the distributions for the responses may be chosen by looking at the histogram of the response along with the estimated density from the assumed distribution, and at the normalised quantile responses. These will provide a rough guide to the adequacy of the chosen distribution. The latter are defined as the quantile standard normal function of the cumulative distribution function of the response with scale and location estimated by MLE. These should behave approximately as normally distributed variables (even though the original observations are not). Therefore, a normal Q-Q plot is appropriate here.

If `loglik = TRUE` then this function also provides the log-likelihood for AIC calculation, for instance.

The shapiro test can also be performed.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

See Also

copulaReg

Examples

```
## see examples in copulaReg
```

---

rMVN *Multivariate Normal Variates*

Description

This function simply generates random multivariate normal variates.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

RR *Causal risk ratio of a binary/continuous/discrete endogenous variable*

Description

RR can be used to calculate the causal risk ratio of a binary/continuous/discrete endogenous predictor/treatment, with corresponding interval obtained using posterior simulation.

Usage

```
RR(x, nm.end, E = TRUE, treat = TRUE, type = "bivariate", ind = NULL,
   n.sim = 100, prob.lev = 0.05, length.out = NULL, hd.plot = FALSE,
   rr.plot = FALSE,
   main = "Histogram and Kernel Density of Simulated Risk Ratios",
   xlab = "Simulated Risk Ratios", ...)
```

Arguments

x	A fitted SemiParBIVProbit/copulaReg object.
nm.end	Name of the endogenous variable.
E	If TRUE then RR calculates the sample RR. If FALSE then it calculates the sample RR for the treated individuals only.
treat	If TRUE then RR calculates the RR using the treated only. If FALSE then it calculates the ratio using the control group. This only makes sense if E = FALSE.
type	This argument can take three values: "naive" (the effect is calculated ignoring the presence of observed and unobserved confounders), "univariate" (the effect is obtained from the univariate probit model which neglects the presence of unobserved confounders) and "bivariate" (the effect is obtained from the bivariate model which accounts for observed and unobserved confounders).
ind	Binary logical variable. It can be used to calculate the RR for a subset of the data. Note that it does not make sense to use ind when some observations are excluded from the RR calculation (e.g., when using E = FALSE).
n.sim	Number of simulated coefficient vectors from the posterior distribution of the estimated model parameters. This is used when delta = FALSE. It may be increased if more precision is required.
prob.lev	Overall probability of the left and right tails of the RR distribution used for interval calculations.
length.out	Ddesired length of the sequence to be used when calculating the effect that a continuous/discrete treatment has on a binary outcome.
hd.plot	If TRUE then a plot of the histogram and kernel density estimate of the simulated risk ratios is produced. This can only be produced when binary responses are used.

rr.plot	For the case of continuous/discrete endogenous variable and binary outcome, if TRUE then a plot (on the log scale) showing the risk ratios that the binary outcome is equal to 1 for each incremental value of the endogenous variable and respective intervals is produced.
main	Title for the plot.
xlab	Title for the x axis.
...	Other graphics parameters to pass on to plotting commands. These are used only when hd.plot = TRUE.

Details

RR calculates the causal risk ratio of the probabilities of positive outcome under treatment (the binary predictor or treatment assumes value 1) and under control (the binary treatment assumes value 0). Posterior simulation is used to obtain a confidence/credible interval.

RR works also for the case of continuous/discrete endogenous treatment variable.

Value

prob.lev	Probability level used.
sim.RR	It returns a vector containing simulated values of the average RR. This is used to calculate intervals.
Ratios	For the case of continuous/discrete endogenous variable and binary outcome, it returns a matrix made up of three columns containing the risk ratios for each incremental value in the endogenous variable and respective intervals.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

See Also

SemiParBIVProbit-package, SemiParBIVProbit, copulaReg

Examples

```
## see examples for SemiParBIVProbit and copulaReg
```

S.m *Internal Function*

Description

It provides penalty matrices in a format suitable for automatic multiple smoothing parameter estimation.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

SemiParBIVProbit *Semiparametric Copula Bivariate Binary Models*

Description

SemiParBIVProbit fits flexible copula bivariate binary models with several types of covariate effects, copula distributions and link functions. During the model fitting process, the possible presence of associated error equations, endogeneity, non-random sample selection or partial observability is accounted for.

Usage

```
SemiParBIVProbit(formula, data = list(), weights = NULL, subset = NULL,
                 Model = "B", BivD = "N",
                 margins = c("probit","probit"), gamlssfit = FALSE,
                 fp = FALSE, hess = TRUE, infl.fac = 1, theta.fx = NULL,
                 rinit = 1, rmax = 100,
                 iterlimsp = 50, tolsp = 1e-07,
                 gc.l = FALSE, parscale, extra.regI = "t", intf = FALSE)
```

Arguments

formula	In the basic setup this will be a list of two formulas, one for equation 1 and the other for equation 2. s terms are used to specify smooth smooth functions of predictors. SemiParBIVProbit supports the use shrinkage smoothers for variable selection purposes and more. See the examples below and the documentation of mgcv for further details on formula specifications. Note that if Model = "BSS" then the first formula MUST refer to the selection equation. Furthermore, if it makes sense, a third equation for the dependence parameter can be specified (see Example 1 below).
data	An optional data frame, list or environment containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which SemiParBIVProbit is called.
weights	Optional vector of prior weights to be used in fitting.
subset	Optional vector specifying a subset of observations to be used in the fitting process.
Model	It indicates the type of model to be used in the analysis. Possible values are "B" (bivariate model), "BSS" (bivariate model with non-random sample selection), "BPO" (bivariate model with partial observability) and "BPO0" (bivariate model with partial observability and zero correlation).
margins	It indicates the link functions used for the two margins. Possible choiced are "probit", "logit" and "cloglog".
gamlssfit	This is for internal purposes only.

BivD	Type of bivariate error distribution employed. Possible choices are "N", "C0", "C90", "C180", "C270", "J0", "J90", "J180", "J270", "G0", "G90", "G180", "G270", "F", "AMH", "FGM" which stand for bivariate normal, Clayton, rotated Clayton (90 degrees), survival Clayton, rotated Clayton (270 degrees), Joe, rotated Joe (90 degrees), survival Joe, rotated Joe (270 degrees), Gumbel, rotated Gumbel (90 degrees), survival Gumbel, rotated Gumbel (270 degrees), Frank, Ali-Mikhail-Haq and Farlie-Gumbel-Morgenstern. Note that Clayton, Joe and Gumbel are somewhat similar. Also, there might be situations in which the use of a specific copula may result in more stable computations.
fp	If TRUE then a fully parametric model with unpenalised regression splines if fitted. See the example below.
hess	If FALSE then the expected/Fisher (rather than observed) information matrix is employed. The Fisher information matrix is not available for cases different from binary treatment and binary outcome.
infl.fac	Inflation factor for the model degrees of freedom in the approximate AIC. Smoother models can be obtained setting this parameter to a value greater than 1.
theta.fx	If Model = "B" and BivD = "N" then the theta parameter can be fixed in estimation.
rinit	Starting trust region radius. The trust region radius is adjusted as the algorithm proceeds. See the documentation of trust for further details.
rmax	Maximum allowed trust region radius. This may be set very large. If set small, the algorithm traces a steepest descent path.
iterlimsp	A positive integer specifying the maximum number of loops to be performed before the smoothing parameter estimation step is terminated.
tolsp	Tolerance to use in judging convergence of the algorithm when automatic smoothing parameter estimation is used.
gc.l	This is relevant when working with big datasets. If TRUE then the garbage collector is called more often than it is usually done. This keeps the memory footprint down but it will slow down the routine.
parscale	The algorithm will operate as if optimizing objfun(x / parscale, ...) where parscale is a scalar. If missing then no rescaling is done. See the documentation of trust for more details.
extra.regI	If "t" then regularization as from trust is applied to the information matrix if needed. If different from "t" then extra regularization is applied via the options "pC" (pivoted Choleski - this will only work when the information matrix is semi-positive or positive definite) and "sED" (symmetric eigen-decomposition).
intf	This is for internal use.

Details

The bivariate models considered in this package consist of two model equations which depend on flexible linear predictors and whose association between the responses is modelled through parameter θ of a standardised bivariate normal distribution or that of a bivariate copula distribution. The linear predictors of the two equations are flexibly specified using parametric components and smooth functions of covariates. The same can be done for the dependence parameter if it makes

sense. Estimation is achieved within a penalized likelihood framework with integrated automatic multiple smoothing parameter selection. The use of penalty matrices allows for the suppression of that part of smooth term complexity which has no support from the data. The trade-off between smoothness and fitness is controlled by smoothing parameters associated with the penalty matrices. Smoothing parameters are chosen to minimise an approximate AIC.

Details of the underlying fitting methods are given in Radice, Marra and Wojtys (2016) and Marra et al. (in press). Releases previous to 3.2-7 were based on the algorithms detailed in Marra and Radice (2011, 2013).

For sample selection models, if there are factors in the model, before fitting, the user has to ensure that the numbers of factor variables' levels in the selected sample are the same as those in the complete dataset. Even if a model could be fitted in such a situation, the model may produce fits which are not coherent with the nature of the correction sought. As an example consider the situation in which the complete dataset contains a factor variable with five levels and that only three of them appear in the selected sample. For the outcome equation (which is the one of interest) only three levels of such variable exist in the population, but their effects will be corrected for non-random selection using a selection equation in which five levels exist instead. Having differing numbers of factors' levels between complete and selected samples will also make prediction not feasible (an aspect which may be particularly important for selection models); clearly it is not possible to predict the response of interest for the missing entries using a dataset that contains all levels of a factor variable but using an outcome model estimated using a subset of these levels.

Value

The function returns an object of class `SemiParBIVProbit` as described in `SemiParBIVProbitObject`.

WARNINGS

Convergence failure may sometimes occur. Convergence can be checked using `conv.check` which provides some information about the score and information matrix associated with the fitted model. The former should be close to 0 and the latter positive definite. `SemiParBIVProbit()` will produce some warnings when there is a convergence issue.

In such a situation, the user may use some extra regularisation (see `extra.regI`) and/or rescaling (see `parscale`). These suggestions may help, especially the latter option. However, the user should also consider re-specifying the model and/or using a diferrent dependence structure and/or using different links. In our experience, we found that convergence failure typically occurs when the model has been misspecified and/or the sample size (and/or number of selected observations in selection models) is low compared to the complexity of the model. Examples of misspecification include using a Clayton copula rotated by 90 degrees when a positive association between the margins is present instead, using marginal distributions that are not adequate, and employing a copula which does not accommodate the type and/or strength of the dependence between the margins (e.g., using AMH when the association between the margins is strong).

In the contexts of endogeneity and non-random sample selection, extra attention is required when specifying the dependence parameter as a function of covariates. This is because in these situations the dependence parameter mainly models the association between the unobserved confounders in the two equations. Therefore, this option would make sense when it is believed that the strength of the association between the unobservables in the two equations varies based on some grouping factor or across geographical areas, for instance.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

References

Marra G. and Radice R. (2011), Estimation of a Semiparametric Recursive Bivariate Probit in the Presence of Endogeneity. *Canadian Journal of Statistics*, 39(2), 259-279.

Marra G. and Radice R. (2013), A Penalized Likelihood Estimation Approach to Semiparametric Sample Selection Binary Response Modeling. *Electronic Journal of Statistics*, 7, 1432-1455.

Marra G., Radice R., Barnighausen T., Wood S.N. and McGovern M.E. (in press), A Simultaneous Equation Approach to Estimating HIV Prevalence with Non-Ignorable Missing Responses. *Journal of the American Statistical Association*.

McGovern M.E., Barnighausen T., Marra G. and Radice R. (2015), On the Assumption of Joint Normality in Selection Models: A Copula Approach Applied to Estimating HIV Prevalence. *Epidemiology*, 26(2), 229-237.

Radice R., Marra G. and Wojtys M. (2016), Copula Regression Spline Models for Binary Outcomes. *Statistics and Computing*, 26(5), 981-995.

Poirier D.J. (1980), Partial Observability in Bivariate Probit Models. *Journal of Econometrics*, 12, 209-217.

See Also

copulaReg, copulaSampleSel, SemiParTRIV, AT, OR, RR, adjCov, prev, gt.bpm, LM.bpm, VuongClarke, plot.SemiParBIVProbit, SemiParBIVProbit-package, SemiParBIVProbitObject, conv.check, summary.SemiParBIVProbit, predict.SemiParBIVProbit

Examples

```
library(SemiParBIVProbit)

############
## EXAMPLE 1
############
## Generate data
## Correlation between the two equations 0.5 - Sample size 400

set.seed(0)

n <- 400

Sigma <- matrix(0.5, 2, 2); diag(Sigma) <- 1
u     <- rMVN(n, rep(0,2), Sigma)

x1 <- round(runif(n)); x2 <- runif(n); x3 <- runif(n)

f1   <- function(x) cos(pi*2*x) + sin(pi*x)
f2   <- function(x) x+exp(-30*(x-0.5)^2)
```

```
y1 <- ifelse(-1.55 + 2*x1    + f1(x2) + u[,1] > 0, 1, 0)
y2 <- ifelse(-0.25 - 1.25*x1 + f2(x2) + u[,2] > 0, 1, 0)

dataSim <- data.frame(y1, y2, x1, x2, x3)

#
#

## CLASSIC BIVARIATE PROBIT

out  <- SemiParBIVProbit(list(y1 ~ x1 + x2 + x3,
                              y2 ~ x1 + x2 + x3),
                         data = dataSim)
conv.check(out)
summary(out)
AIC(out)
BIC(out)

## SEMIPARAMETRIC BIVARIATE PROBIT

## "cr" cubic regression spline basis      - "cs" shrinkage version of "cr"
## "tp" thin plate regression spline basis - "ts" shrinkage version of "tp"
## for smooths of one variable, "cr/cs" and "tp/ts" achieve similar results
## k is the basis dimension - default is 10
## m is the order of the penalty for the specific term - default is 2
## For COPULA models use BivD argument

out  <- SemiParBIVProbit(list(y1 ~ x1 + s(x2, bs = "tp", k = 10, m = 2) + s(x3),
                              y2 ~ x1 + s(x2) + s(x3)),
                         data = dataSim)
conv.check(out)
summary(out, cm.plot = TRUE)
AIC(out)


## estimated smooth function plots - red lines are true curves

x2 <- sort(x2)
f1.x2 <- f1(x2)[order(x2)] - mean(f1(x2))
f2.x2 <- f2(x2)[order(x2)] - mean(f2(x2))
f3.x3 <- rep(0, length(x3))

par(mfrow=c(2,2),mar=c(4.5,4.5,2,2))
plot(out, eq = 1, select = 1, seWithMean = TRUE, scale = 0)
lines(x2, f1.x2, col = "red")
plot(out, eq = 1, select = 2, seWithMean = TRUE, scale = 0)
lines(x3, f3.x3, col = "red")
plot(out, eq = 2, select = 1, seWithMean = TRUE, scale = 0)
lines(x2, f2.x2, col = "red")
plot(out, eq = 2, select = 2, seWithMean = TRUE, scale = 0)
lines(x3, f3.x3, col = "red")

## p-values suggest to drop x3 from both equations, with a stronger
```

```
## evidence for eq. 2. This can be also achieved using shrinkage smoothers

outSS <- SemiParBIVProbit(list(y1 ~ x1 + s(x2, bs = "ts") + s(x3, bs = "cs"),
                               y2 ~ x1 + s(x2, bs = "cs") + s(x3, bs = "ts")),
                          data = dataSim)
conv.check(outSS)

plot(outSS, eq = 1, select = 1, scale = 0, shade = TRUE)
plot(outSS, eq = 1, select = 2, ylim = c(-0.1,0.1))
plot(outSS, eq = 2, select = 1, scale = 0, shade = TRUE)
plot(outSS, eq = 2, select = 2, ylim = c(-0.1,0.1))

## Not run:

## SEMIPARAMETRIC BIVARIATE PROBIT with association parameter
## depending on covariates as well

eq.mu.1  <- y1 ~ x1 + s(x2)
eq.mu.2  <- y2 ~ x1 + s(x2)
eq.theta <-    ~ x1 + s(x2)

fl <- list(eq.mu.1, eq.mu.2, eq.theta)

outD <- SemiParBIVProbit(fl, data = dataSim)
conv.check(outD)
summary(outD)
outD$theta

plot(outD, eq = 1, seWithMean = TRUE)
plot(outD, eq = 2, seWithMean = TRUE)
plot(outD, eq = 3, seWithMean = TRUE)
graphics.off()

#
#

############
## EXAMPLE 2
############
## Generate data with one endogenous variable
## and exclusion restriction

set.seed(0)

n <- 400

Sigma <- matrix(0.5, 2, 2); diag(Sigma) <- 1
u     <- rMVN(n, rep(0,2), Sigma)

cov   <- rMVN(n, rep(0,2), Sigma)
cov   <- pnorm(cov)
x1 <- round(cov[,1]); x2 <- cov[,2]
```

```
f1   <- function(x) cos(pi*2*x) + sin(pi*x)
f2   <- function(x) x+exp(-30*(x-0.5)^2)

y1 <- ifelse(-1.55 + 2*x1    + f1(x2) + u[,1] > 0, 1, 0)
y2 <- ifelse(-0.25 - 1.25*y1 + f2(x2) + u[,2] > 0, 1, 0)

dataSim <- data.frame(y1, y2, x1, x2)

#

## Testing the hypothesis of absence of endogeneity...

LM.bpm(list(y1 ~ x1 + s(x2), y2 ~ y1 + s(x2)), dataSim, Model = "B")

# p-value suggests presence of endogeneity, hence fit a bivariate model


## CLASSIC RECURSIVE BIVARIATE PROBIT

out <- SemiParBIVProbit(list(y1 ~ x1 + x2,
                             y2 ~ y1 + x2),
                        data = dataSim)
conv.check(out)
summary(out)
AIC(out); BIC(out)

## SEMIPARAMETRIC RECURSIVE BIVARIATE PROBIT

out <- SemiParBIVProbit(list(y1 ~ x1 + s(x2),
                             y2 ~ y1 + s(x2)),
                        data = dataSim)
conv.check(out)
summary(out)
AIC(out); BIC(out)

#

## Testing the hypothesis of absence of endogeneity post estimation...

gt.bpm(out)

#
## reatment effect, risk ratio and odds ratio with CIs

mb(y1, y2, Model = "B")
AT(out, nm.end = "y1", hd.plot = TRUE)
RR(out, nm.end = "y1")
OR(out, nm.end = "y1")
AT(out, nm.end = "y1", type = "univariate")


## try a Clayton copula model...
```

```
outC <- SemiParBIVProbit(list(y1 ~ x1 + s(x2),
                              y2 ~ y1 + s(x2)),
                         data = dataSim, BivD = "C0")
conv.check(outC)
summary(outC)
AT(outC, nm.end = "y1")

## try a Joe copula model...

outJ <- SemiParBIVProbit(list(y1 ~ x1 + s(x2),
                              y2 ~ y1 + s(x2)),
                         data = dataSim, BivD = "J0")
conv.check(outJ)
summary(outJ, cm.plot = TRUE)
AT(outJ, "y1")


VuongClarke(out, outJ)

#
## recursive bivariate probit modelling with unpenalized splines
## can be achieved as follows

outFP <- SemiParBIVProbit(list(y1 ~ x1 + s(x2, bs = "cr", k = 5),
                               y2 ~ y1 + s(x2, bs = "cr", k = 6)),
                          fp = TRUE, data = dataSim)
conv.check(outFP)
summary(outFP)

# in the above examples a third equation could be introduced
# as illustrated in Example 1

#
##################
## See also ?meps
##################

#############
## EXAMPLE 3
#############
## Generate data with a non-random sample selection mechanism
## and exclusion restriction

set.seed(0)

n <- 2000

Sigma <- matrix(0.5, 2, 2); diag(Sigma) <- 1
u     <- rMVN(n, rep(0,2), Sigma)

SigmaC <- matrix(0.5, 3, 3); diag(SigmaC) <- 1
cov    <- rMVN(n, rep(0,3), SigmaC)
cov    <- pnorm(cov)
```

```
bi <- round(cov[,1]); x1 <- cov[,2]; x2 <- cov[,3]

f11 <- function(x) -0.7*(4*x + 2.5*x^2 + 0.7*sin(5*x) + cos(7.5*x))
f12 <- function(x) -0.4*( -0.3 - 1.6*x + sin(5*x))
f21 <- function(x) 0.6*(exp(x) + sin(2.9*x))

ys <-  0.58 + 2.5*bi + f11(x1) + f12(x2) + u[, 1] > 0
y  <- -0.68 - 1.5*bi + f21(x1) +         + u[, 2] > 0
yo <- y*(ys > 0)

dataSim <- data.frame(y, ys, yo, bi, x1, x2)

## Testing the hypothesis of absence of non-random sample selection...

LM.bpm(list(ys ~ bi + s(x1) + s(x2), yo ~ bi + s(x1)), dataSim, Model = "BSS")

# p-value suggests presence of sample selection, hence fit a bivariate model

#
## SEMIPARAMETRIC SAMPLE SELECTION BIVARIATE PROBIT
## the first equation MUST be the selection equation

out <- SemiParBIVProbit(list(ys ~ bi + s(x1) + s(x2),
                             yo ~ bi + s(x1)),
                        data = dataSim, Model = "BSS")
conv.check(out)
gt.bpm(out)

## compare the two summary outputs
## the second output produces a summary of the results obtained when
## selection bias is not accounted for

summary(out)
summary(out$gam2)

## corrected predicted probability that 'yo' is equal to 1

mb(ys, yo, Model = "BSS")
prev(out, hd.plot = TRUE)
prev(out, type = "univariate", hd.plot = TRUE)

## estimated smooth function plots
## the red line is the true curve
## the blue line is the univariate model curve not accounting for selection bias

x1.s <- sort(x1[dataSim$ys>0])
f21.x1 <- f21(x1.s)[order(x1.s)]-mean(f21(x1.s))

plot(out, eq = 2, ylim = c(-1.65,0.95)); lines(x1.s, f21.x1, col="red")
par(new = TRUE)
plot(out$gam2, se = FALSE, col = "blue", ylim = c(-1.65,0.95),
     ylab = "", rug = FALSE)
```

```
#
#
## try a Clayton copula model...

outC <- SemiParBIVProbit(list(ys ~ bi + s(x1) + s(x2),
                              yo ~ bi + s(x1)),
                         data = dataSim, Model = "BSS", BivD = "C0")
conv.check(outC)
summary(outC, cm.plot = TRUE)
prev(outC)

# in the above examples a third equation could be introduced
# as illustrated in Example 1

#
#################
## See also ?hiv
#################

#############
## EXAMPLE 4
#############
## Generate data with partial observability

set.seed(0)

n <- 10000

Sigma <- matrix(0.5, 2, 2); diag(Sigma) <- 1
u     <- rMVN(n, rep(0,2), Sigma)

x1 <- round(runif(n)); x2 <- runif(n); x3 <- runif(n)

y1 <- ifelse(-1.55 + 2*x1 + x2 + u[,1] > 0, 1, 0)
y2 <- ifelse( 0.45 - x3        + u[,2] > 0, 1, 0)
y  <- y1*y2

dataSim <- data.frame(y, x1, x2, x3)


## BIVARIATE PROBIT with Partial Observability

out  <- SemiParBIVProbit(list(y ~ x1 + x2,
                              y ~ x3),
                         data = dataSim, Model = "BPO")
conv.check(out)
summary(out)

# first ten estimated probabilities for the four events from object out

cbind(out$p11, out$p10, out$p00, out$p01)[1:10,]
```

```
# case with smooth function
# (more computationally intensive)

f1 <- function(x) cos(pi*2*x) + sin(pi*x)

y1 <- ifelse(-1.55 + 2*x1 + f1(x2) + u[,1] > 0, 1, 0)
y2 <- ifelse( 0.45 - x3            + u[,2] > 0, 1, 0)
y  <- y1*y2

dataSim <- data.frame(y, x1, x2, x3)

out  <- SemiParBIVProbit(list(y ~ x1 + s(x2),
                              y ~ x3),
                         data = dataSim, Model = "BPO")

conv.check(out)
summary(out, cm.plot = TRUE)


# plot estimated and true functions

x2 <- sort(x2); f1.x2 <- f1(x2)[order(x2)] - mean(f1(x2))
plot(out, eq = 1, scale = 0); lines(x2, f1.x2, col = "red")

#
################
## See also ?war
################


## End(Not run)
```

SemiParBIVProbit.fit *Internal Function*

Description

Wrapper of core algorithm.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

SemiParBIVProbit.fit.post

Internal Function

Description

This and other similar internal functions calculate useful post estimation quantities.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

SemiParBIVProbitObject

Fitted SemiParBIVProbit object

Description

A fitted semiparametric bivariate object returned by function `SemiParBIVProbit` and of class "SemiParBIVProbit".

Value

`fit`	List of values and diagnostics extracted from the output of the algorithm. For instance, `fit$gradient`, `fit$Fisher` and `fit$S.h` return the gradient vector, Fisher information (when used) and overall penalty matrix scaled by its smoothing parameters, for the fitted bivariate probit model. See the documentation of `trust` for details on the diagnostics provided.
`gam1`	Univariate fit for equation 1. See the documentation of `mgcv` for full details.
`gam2, gam3, ...`	Univariate fit for equation 2 and equations 3 and 4 (these are available when the dispersion and association parameters are modelled as functions of covariates).
`coefficients`	The coefficients of the fitted model. They are given in the following order: parametric and regression spline (if present) coefficients for the first equation, parametric and regression spline coefficients for the second equation, and dispersion parameter (or coefficients for the third equation) and association coefficient (or coefficients for the fourth equation).
`weights`	Prior weights used during model fitting.
`sp`	Estimated smoothing parameters of the smooth components.
`iter.sp`	Number of iterations performed for the smoothing parameter estimation step.
`iter.if`	Number of iterations performed in the initial step of the algorithm.
`iter.inner`	Number of iterations performed within the smoothing parameter estimation step.

theta	Estimated dependence parameter linking the two equations.
n	Sample size.
n.sel	Number of selected observations in the sample selection model case.
X1, X2, X3, ...	Design matrices associated with the linear predictors.
X1.d2, X2.d2, X3.d2, ...	Number of columns of X1, X2, X3, etc.
l.sp1, l.sp2, l.sp3, ...	Number of smooth components in the equations.
He	Penalized -hessian/Fisher. This is the same as HeSh for unpenalized models.
HeSh	Unpenalized -hessian/Fisher.
Vb	Inverse of He. This corresponds to the Bayesian variance-covariance matrix used for confidence/credible interval calculations.
t.edf	Total degrees of freedom of the estimated bivariate model. It is calculated as sum(diag(F)).
edf1, edf2, edf3, ...	Degrees of freedom for the two equations of the fitted bivariate model (and for the third and fourth equations if present. They are calculated when splines are used.
bs.mgfit	List of values and diagnostics extracted from magic in mgcv.
conv.sp	If TRUE then the smoothing parameter selection algorithm stopped before reaching the maximum number of iterations allowed.
wor.c	Working model quantities.
p11, p10, p01, p00	Model probabilities evaluated at (y_1 = 1, y_2 = 1), (y_1 = 1, y_2 = 0), (y_1 = 0, y_2 = 1) and (y_1 = 0, y_2 = 0).
p1, p2	Marginal probabilities.
p1n, p2n	Univariate marginal probabilities. These are only provided when Method = "BSS" and are built using two separate fits.
eta1, eta2, eta3, ...	Estimated linear predictors for the two equations (as well as the third and fourth equations if present).
y1, y2	Responses of the two equations.
logLik	Value of the (unpenalized) log-likelihood evaluated at the (penalized or unpenalized) parameter estimates.
respvec	List containing response vectors.
X2s	Full design matrix of outcome equation for sample selection case.
OR, GM	Odds ratio and Gamma measure. See summary.SemiParBIVProbit for details.
tau	Kendall's tau.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

See Also

SemiParBIVProbit, plot.SemiParBIVProbit, summary.SemiParBIVProbit, predict.SemiParBIVProbit

---

SemiParTRIV *Semiparametric Trivariate Binary Models*

Description

SemiParTRIV fits flexible trivariate binary probit models with several types of covariate effects.

Usage

```
SemiParTRIV(formula, data = list(), weights = NULL, subset = NULL,
            Model = "T", penCor = "unpen", sp.penCor = 3,
            approx = FALSE, infl.fac = 1,
            gamma = 1, w.alasso = NULL, rinit = 1, rmax = 100,
            iterlimsp = 50, tolsp = 1e-07,
            gc.l = FALSE, parscale, extra.regI = "t")
```

Arguments

formula	In the basic setup this will be a list of three formulas. s terms are used to specify smooth smooth functions of predictors. See the examples below and the documentation of SemiParBIVProbit for further details.
data	An optional data frame, list or environment containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which SemiParTRIV is called.
weights	Optional vector of prior weights to be used in fitting.
subset	Optional vector specifying a subset of observations to be used in the fitting process.
Model	It indicates the type of model to be used in the analysis. Possible values are "T" (trivariate model), "TSS" (trivariate model with double sample selection).
penCor	Type of penalty for correlation coefficients. Possible values are "unpen", "lasso", "ridge", "alasso".
sp.penCor	Starting value for smoothing parameter of penCor.
approx	If TRUE then an approximation of the trivariate normal integral is employed. This may speed up computations but make them unstable at the same time (especially for highly correlated responses).
infl.fac	Inflation factor for the model degrees of freedom in the approximate AIC. Smoother models can be obtained setting this parameter to a value greater than 1.
gamma	Inflation factor used only for the alasso penalty.
w.alasso	When using the alasso penalty a weight vector made up of three values must be provided.

rinit	Starting trust region radius. The trust region radius is adjusted as the algorithm proceeds. See the documentation of trust for further details.
rmax	Maximum allowed trust region radius. This may be set very large. If set small, the algorithm traces a steepest descent path.
iterlimsp	A positive integer specifying the maximum number of loops to be performed before the smoothing parameter estimation step is terminated.
tolsp	Tolerance to use in judging convergence of the algorithm when automatic smoothing parameter estimation is used.
gc.l	This is relevant when working with big datasets. If TRUE then the garbage collector is called more often than it is usually done. This keeps the memory footprint down but it will slow down the routine.
parscale	The algorithm will operate as if optimizing objfun(x / parscale, ...) where parscale is a scalar. If missing then no rescaling is done. See the documentation of trust for more details.
extra.regI	If "t" then regularization as from trust is applied to the information matrix if needed. If different from "t" then extra regularization is applied via the options "pC" (pivoted Choleski - this will only work when the information matrix is semi-positive or positive definite) and "sED" (symmetric eigen-decomposition).

Details

This function fits trivariate binary models.

For sample selection models, if there are factors in the model, before fitting, the user has to ensure that the numbers of factor variables' levels in the selected sample are the same as those in the complete dataset. Even if a model could be fitted in such a situation, the model may produce fits which are not coherent with the nature of the correction sought. For more details see ?SemiParBIVProbit.

Value

The function returns an object of class SemiParTRIV as described in SemiParTRIVObject.

WARNINGS

Convergence failure may sometimes occur. Convergence can be checked using conv.check which provides some information about the score and information matrix associated with the fitted model. The former should be close to 0 and the latter positive definite. SemiParTRIV() will produce some warnings if there is a convergence issue.

In such a situation, the user may use some extra regularisation (see extra.regI) and/or rescaling (see parscale). Penalising the correlations using argument penCor may help a lot as in our experience in hard situations the correlation coefficients are typically the most difficult to estimate.

The above suggestions may help, especially the latter option. However, the user should also consider looking into the proportions of 1 and 0 available for each event of the trivariate model. It may be the case that certain events do not have many observations associated with them, in which case estimation may be more challenging. As usual, model complexity plays a role.

Author(s)

Authors: Panagiota Filippou and Giampiero Marra

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

References

Filippou P., Marra G. and Radice R. (submitted), A Penalised Estimation Approach to Additive Trivariate Probit Regression.

See Also

SemiParBIVProbit, copulaReg, copulaSampleSel, SemiParBIVProbit-package, SemiParTRIVObject, conv.check, summary.SemiParTRIV

Examples

```
## Not run:

library(SemiParBIVProbit)

############
## EXAMPLE 1
############
## Generate data
## Correlation between the two equations 0.5 - Sample size 400

set.seed(0)

n <- 400

Sigma <- matrix(0.5, 3, 3); diag(Sigma) <- 1
u     <- rMVN(n, rep(0,3), Sigma)

x1 <- round(runif(n)); x2 <- runif(n); x3 <- runif(n)

f1   <- function(x) cos(pi*2*x) + sin(pi*x)
f2   <- function(x) x+exp(-30*(x-0.5)^2)

y1 <- ifelse(-1.55 +    2*x1 - f1(x2) + u[,1] > 0, 1, 0)
y2 <- ifelse(-0.25 - 1.25*x1 + f2(x2) + u[,2] > 0, 1, 0)
y3 <- ifelse(-0.75 + 0.25*x1          + u[,3] > 0, 1, 0)

dataSim <- data.frame(y1, y2, y3, x1, x2)

out  <- SemiParTRIV(list(y1 ~ x1 + s(x2),
                         y2 ~ x1 + s(x2),
                         y3 ~ x1),
                         data = dataSim)
conv.check(out)
summary(out)
```

```
plot(out, eq = 1)
plot(out, eq = 2)
AIC(out)
BIC(out)

############
## EXAMPLE 2
############
## Generate data
## with double sample selection

set.seed(0)

n <- 5000

Sigma <- matrix(c(1,   0.5, 0.4,
                  0.5,   1, 0.6,
                  0.4, 0.6,   1 ), 3, 3)

u <- rMVN(n, rep(0,3), Sigma)

f1   <- function(x) cos(pi*2*x) + sin(pi*x)
f2   <- function(x) x+exp(-30*(x-0.5)^2)

x1 <- runif(n)
x2 <- runif(n)
x3 <- runif(n)
x4 <- runif(n)

y1 <-  1    + 1.5*x1 -     x2 + 0.8*x3 - f1(x4) + u[, 1] > 0
y2 <-  1    - 2.5*x1 + 1.2*x2 +     x3          + u[, 2] > 0
y3 <-  1.58 + 1.5*x1 - f2(x2)                   + u[, 3] > 0

dataSim <- data.frame(y1, y2, y3, x1, x2, x3, x4)

f.l <- list(y1 ~ x1 + x2 + x3 + s(x4),
            y2 ~ x1 + x2 + x3,
            y3 ~ x1 + s(x2))

out <- SemiParTRIV(f.l, data = dataSim, Model = "TSS")
conv.check(out)
summary(out)
plot(out, eq = 1)
plot(out, eq = 3)
prev(out)
prev(out, type = "univariate")
prev(out, type = "naive")


## End(Not run)
```

SemiParTRIVObject *Fitted SemiParTRIV object*

Description

A fitted semiparametric trivariate probit object returned by function SemiParTRIV and of class "SemiParTRIV".

Value

fit	List of values and diagnostics extracted from the output of the algorithm. For instance, fit$gradient, fit$Fisher and fit$S.h return the gradient vector, Fisher information (when used) and overall penalty matrix scaled by its smoothing parameters, for the fitted bivariate probit model. See the documentation of trust for details on the diagnostics provided.
gam1	Univariate fit for equation 1. See the documentation of mgcv for full details.
gam2, gam3, ...	Univariate fit for equation 2 and equations 3 and 4 (these are available when the dispersion and association parameters are modelled as functions of covariates).
coefficients	The coefficients of the fitted model. They are given in the following order: parametric and regression spline (if present) coefficients for the first equation, parametric and regression spline coefficients for the second equation, and dispersion parameter (or coefficients for the third equation) and association coefficient (or coefficients for the fourth equation).
weights	Prior weights used during model fitting.
sp	Estimated smoothing parameters of the smooth components.
iter.sp	Number of iterations performed for the smoothing parameter estimation step.
iter.if	Number of iterations performed in the initial step of the algorithm.
iter.inner	Number of iterations performed within the smoothing parameter estimation step.
theta	Estimated dependence parameter linking the two equations.
n	Sample size.
X1, X2, X3, ...	Design matrices associated with the linear predictors.
X1.d2, X2.d2, X3.d2, ...	Number of columns of X1, X2, X3, etc.
l.sp1, l.sp2, l.sp3, ...	Number of smooth components in the equations.
He	Penalized -hessian/Fisher. This is the same as HeSh for unpenalized models.
HeSh	Unpenalized -hessian/Fisher.
Vb	Inverse of He. This corresponds to the Bayesian variance-covariance matrix used for confidence/credible interval calculations.

t.edf	Total degrees of freedom of the estimated bivariate model. It is calculated as sum(diag(F)).
edf1, edf2, edf3, ...	Degrees of freedom for the two equations of the fitted bivariate model (and for the third and fourth equations if present. They are calculated when splines are used.
bs.mgfit	List of values and diagnostics extracted from magic in mgcv.
conv.sp	If TRUE then the smoothing parameter selection algorithm stopped before reaching the maximum number of iterations allowed.
wor.c	Working model quantities.
p111, p110,...	Model probabilities.
eta1, eta2, eta3, ...	Estimated linear predictors for the two equations (as well as the third and fourth equations if present).
y1, y2, y3	Responses of the two equations.
logLik	Value of the (unpenalized) log-likelihood evaluated at the (penalized or unpenalized) parameter estimates.
respvec	List containing response vectors.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

See Also

SemiParTRIV, plot.SemiParBIVProbit, summary.SemiParTRIV, predict.SemiParBIVProbit

---

summary.copulaReg *copulaReg summary*

Description

It takes a fitted copulaReg object produced by copulaReg() and produces some summaries from it.

Usage

```
## S3 method for class 'copulaReg'
summary(object, n.sim = 100, prob.lev = 0.05, cm.plot = FALSE,
        ylab = "Margin 2", xlab = "Margin 1", n.grid = 1000, n.dig = 2, ...)



## S3 method for class 'summary.copulaReg'
print(x, digits = max(3, getOption("digits") - 3),
           signif.stars = getOption("show.signif.stars"), ...)
```

Arguments

object	A fitted copulaReg object as produced by copulaReg().
x	summary.copulaReg object produced by summary.copulaReg().
n.sim	The number of simulated coefficient vectors from the posterior distribution of the estimated model parameters. This is used to calculate intervals for the association parameter, dispersion coefficient etc. It may be increased if more precision is required.
prob.lev	Probability of the left and right tails of the posterior distribution used for interval calculations.
cm.plot	If TRUE then a filled bivariate contour meta plot corresponding to the assumed (estimated) bivariate model is produced.
xlab, ylab	Margin labels.
n.grid	Number of grid points used in contour plot construction. This is relevant for continuous margins.
n.dig	Number of digit points used in rounding bivariate pdf values for contour plot construction. This is relevant for continuous margins.
digits	Number of digits printed in output.
signif.stars	By default significance stars are printed alongside output.
...	Other graphics parameters to pass on to plotting commands. These are used only when cm.plot=TRUE.

Details

This function is very similar to summary.SemiParBIVProbit().

print.summary.copulaReg prints model term summaries.

Value

tableP1	Table containing parametric estimates, their standard errors, z-values and p-values for equation 1.
tableP2,tableP3, ...	As above but for equation 2 and equations 3 and 4 if present.
tableNP1	Table of nonparametric summaries for each smooth component including effective degrees of freedom, estimated rank, approximate Wald statistic for testing the null hypothesis that the smooth term is zero and corresponding p-value, for equation 1.
tableNP2,tableNP3, ...	As above but for equation 2 and equations 3 and 4 if present.
n	Sample size.
theta	Estimated dependence parameter linking the two equations.
sigma21,sigma22	Estimated distribution specific parameters for equations 1 and 2.
nu1,nu2	Estimated distribution specific parameters for equations 1 and 2.

`formula1,formula2,formula3, ...`
: Formulas used for the model equations.

`l.sp1,l.sp2,l.sp3, ...`
: Number of smooth components in model equations.

`t.edf`
: Total degrees of freedom of the estimated bivariate model.

`CItheta`
: Interval(s) for θ.

`CIsig21,CIsig22,CInu1,CInu2`
: Intervals for distribution specific parameters

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

See Also

`plot.SemiParBIVProbit`, `predict.SemiParBIVProbit`

Examples

```
## see examples for copulaReg
```

`summary.copulaSampleSel` *copulaSampleSel summary*

Description

It takes a fitted `copulaSampleSel` object produced by `copulaSampleSel()` and produces some summaries from it.

Usage

```
## S3 method for class 'copulaSampleSel'
summary(object, n.sim = 100, prob.lev = 0.05, cm.plot = FALSE,
        xlim = c(-3, 3), ylim = c(-3, 3),
        ylab = "Margin 2", xlab = "Margin 1",
        n.grid = 1000, n.dig = 2, ...)

## S3 method for class 'summary.copulaSampleSel'
print(x, digits = max(3, getOption("digits") - 3),
          signif.stars = getOption("show.signif.stars"), ...)
```

Arguments

object	A fitted copulaSampleSel object as produced by copulaSampleSel().
x	summary.copulaSampleSel object produced by summary.copulaSampleSel().
n.sim	The number of simulated coefficient vectors from the posterior distribution of the estimated model parameters. This is used to calculate intervals for the association parameter, dispersion coefficient, for instance It may be increased if more precision is required.
prob.lev	Probability of the left and right tails of the posterior distribution used for interval calculations.
cm.plot	If TRUE then a filled bivariate contour meta plot corresponding to the assumed (estimated) bivariate model is produced.
xlim, ylim	Limits of the bivariate contour meta plot.
ylab, xlab	Labels for the bivariate contour meta plot.
n.grid	Number of grid points used in contour plot construction. This is relevant for the continuous margin.
n.dig	Number of digit points used in rounding bivariate pdf values for contour plot construction. This is relevant for the continuous margin.
digits	Number of digits printed in output.
signif.stars	By default significance stars are printed alongside output.
...	Other graphics parameters to pass on to plotting commands. These are used only when cm.plot=TRUE.

Details

This function is very similar to summary.SemiParBIVProbit().

print.summary.copulaSampleSel prints model term summaries.

Value

Very similar to what obtained when using summary.SemiParBIVProbit().

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

See Also

plot.SemiParBIVProbit, predict.SemiParBIVProbit

Examples

```
## see examples for copulaSampleSel
```

summary.SemiParBIVProbit *SemiParBIVProbit summary*

Description

It takes a fitted SemiParBIVProbit object produced by SemiParBIVProbit() and produces some summaries from it.

Usage

```
## S3 method for class 'SemiParBIVProbit'
summary(object, n.sim = 100, prob.lev = 0.05, cm.plot = FALSE,
        xlim = c(-3, 3), ylim = c(-3, 3),
        ylab = "Margin 2", xlab = "Margin 1",
        gm = FALSE, n.grid = 1000, n.dig = 2, ...)

## S3 method for class 'summary.SemiParBIVProbit'
print(x, digits = max(3, getOption("digits") - 3),
          signif.stars = getOption("show.signif.stars"), ...)
```

Arguments

object	A fitted SemiParBIVProbit object as produced by SemiParBIVProbit().
x	summary.SemiParBIVProbit object produced by summary.SemiParBIVProbit().
n.sim	The number of simulated coefficient vectors from the posterior distribution of the estimated model parameters. This is used to calculate intervals for the association parameter, dispersion coefficient and other measures (e.g., gamma measure). It may be increased if more precision is required.
prob.lev	Probability of the left and right tails of the posterior distribution used for interval calculations.
cm.plot	If TRUE then a filled bivariate contour meta plot corresponding to the assumed (estimated) bivariate model is produced.
xlim, ylim	Limits of the bivariate contour meta plot.
ylab, xlab	Labels for the bivariate contour meta plot.
gm	If TRUE then intervals for the gamma measure and odds ratio are calculated.
n.grid	Number of grid points used in contour plot construction. This is relevant for continuous margins.
n.dig	Number of digit points used in rounding bivariate pdf values for contour plot construction. This is relevant for continuous margins.
digits	Number of digits printed in output.
signif.stars	By default significance stars are printed alongside output.

... Other graphics parameters to pass on to plotting commands. These are used only when cm.plot=TRUE.

Details

Using some low level functions in mgcv, based on the results of Marra and Wood (2012), 'Bayesian p-values' are returned for the smooth terms. These have better frequentist performance than their frequentist counterpart. See the help file of summary.gam in mgcv for further details. Covariate selection can also be achieved using a single penalty shrinkage approach as shown in Marra and Wood (2011).

Posterior simulation is used to obtain intervals of nonlinear functions of parameters, such as the association and dispersion parameters as well as the odds ratio and gamma measure discussed by Tajar et al. (2001) if gm = TRUE.

The bivariate contour meta plot has been introduced to provide the user with a pictorial representation of the latent distribution of the model errors.

print.summary.SemiParBIVProbit prints model term summaries.

Value

tableP1 Table containing parametric estimates, their standard errors, z-values and p-values for equation 1.

tableP2,tableP3, ...
As above but for equation 2 and equations 3 and 4 if present.

tableNP1 Table of nonparametric summaries for each smooth component including effective degrees of freedom, estimated rank, approximate Wald statistic for testing the null hypothesis that the smooth term is zero and corresponding p-value, for equation 1.

tableNP2,tableNP3, ...
As above but for equation 2 and equations 3 and 4 if present.

n Sample size.

theta Estimated dependence parameter linking the two equations.

formula1,formula2,formula3, ...
Formulas used for the model equations.

l.sp1,l.sp2,l.sp3, ...
Number of smooth components in model equations.

t.edf Total degrees of freedom of the estimated bivariate model.

CItheta Interval(s) for θ.

n.sel Number of selected observations in the sample selection case.

OR, CIor Odds ratio and related CI. The odds ratio is a measure of association between binary random variables and is defined as p00p11/p10p01. In the case of independence this ratio is equal to 1. It can take values in the range (-Inf, Inf) and it does not depend on the marginal probabilities (Tajar et al., 2001). Interval is calculated using posterior simulation.

GM, CIgm	Gamma measure and related CI. This measure of association was proposed by Goodman and Kruskal (1954). It is defined as (OR - 1)/(OR + 1), can take values in the range (-1, 1) and does not depend on the marginal probabilities. Interval is calculated using posterior simulation.
tau, CIkt	Kendall's tau and respective intervals.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

References

Marra G. and Wood S.N. (2011), Practical Variable Selection for Generalized Additive Models. *Computational Statistics and Data Analysis*, 55(7), 2372-2387.

Marra G. and Wood S.N. (2012), Coverage Properties of Confidence Intervals for Generalized Additive Model Components. *Scandinavian Journal of Statistics*, 39(1), 53-74.

Tajar M., Denuit M. and Lambert P. (2001), Copula-Type Representation for Random Couples with Bernoulli Margins. Discussion Papaer 0118, Universite Catholique De Louvain.

See Also

AT, prev, SemiParBIVProbitObject, plot.SemiParBIVProbit, predict.SemiParBIVProbit

Examples

```
## see examples for SemiParBIVProbit
```

---

summary.SemiParTRIV *SemiParTRIV summary*

Description

It takes a fitted SemiParTRIV object produced by SemiParTRIV() and produces some summaries from it.

Usage

```
## S3 method for class 'SemiParTRIV'
summary(object, n.sim = 100, prob.lev = 0.05, ...)

## S3 method for class 'summary.SemiParTRIV'
print(x, digits = max(3, getOption("digits") - 3),
           signif.stars = getOption("show.signif.stars"), ...)
```

Arguments

object	A fitted SemiParTRIV object as produced by SemiParTRIV().
x	summary.SemiParTRIV object produced by summary.SemiParTRIV().
n.sim	The number of simulated coefficient vectors from the posterior distribution of the estimated model parameters. This is used to calculate intervals for the association parameter and other measures. It may be increased if more precision is required.
prob.lev	Probability of the left and right tails of the posterior distribution used for interval calculations.
digits	Number of digits printed in output.
signif.stars	By default significance stars are printed alongside output.
...	Other arguments.

Details

This function is very similar to summary.SemiParBIVProbit().

print.summary.SemiParTRIV prints model term summaries.

Value

Very similar to what obtained when using summary.SemiParBIVProbit().

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

See Also

plot.SemiParBIVProbit, predict.SemiParBIVProbit

Examples

```
## see examples for SemiParTRIV
```

TRIapprox *Internal Function*

Description

It approximates the trivariate normal integral.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

triprobgHs *Internal Function*

Description

It provides score and Hessian for trivariate binary models.

Author(s)

Author: Panagiota Filippou

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

VuongClarke *Vuong and Clarke tests*

Description

The Vuong and Clarke tests are likelihood-ratio-based tests that can be used for choosing between two non-nested models.

Usage

```
VuongClarke(obj1, obj2, sig.lev = 0.05)
```

Arguments

obj1,obj2	Objects of the two fitted bivariate non-nested models.
sig.lev	Significance level used for testing.

Details

The Vuong (1989) and Clarke (2007) tests are likelihood-ratio-based tests for model selection that use the Kullback-Leibler information criterion. The implemented tests can be used for choosing between two bivariate models which are non-nested.

In the Vuong test, the null hypothesis is that the two models are equally close to the actual model, whereas the alternative is that one model is closer. The test follows asymptotically a standard normal distribution under the null. Assume that the critical region is $(-c, c)$, where c is typically set to 1.96. If the value of the test is higher than c then we reject the null hypothesis that the models are equivalent in favor of model obj1. Viceversa if the value is smaller than c. If the value falls in $[-c, c]$ then we cannot discriminate between the two competing models given the data.

In the Clarke test, if the two models are statistically equivalent then the log-likelihood ratios of the observations should be evenly distributed around zero and around half of the ratios should be

larger than zero. The test follows asymptotically a binomial distribution with parameters n and 0.5. Critical values can be obtained as shown in Clarke (2007). Intuitively, model `obj1` is preferred over `obj2` if the value of the test is significantly larger than its expected value under the null hypothesis $(n/2)$, and vice versa. If the value is not significantly different from $n/2$ then `obj1` can be thought of as equivalent to `obj2`.

Value

It returns two decisions based on the tests and criteria discussed above.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

References

Clarke K. (2007), A Simple Distribution-Free Test for Non-Nested Model Selection. *Political Analysis*, 15, 347-363.

Vuong Q.H. (1989), Likelihood Ratio Tests for Model Selection and Non-Nested Hypotheses. *Econometrica*, 57(2), 307-333.

Examples

```
## see examples for SemiParBIVProbit
```

---

war *Civil war data*

Description

Civil war data from Fearon and Laitin (2003).

Usage

```
data(war)
```

Format

`war` is a 6326 row data frame with the following columns

onset equal to 1 for all country-years in which a civil war started.

instab equal to 1 if unstable government.

oil equal to 1 for oil exporter country.

warl equal to 1 if the country had a distinct civil war ongoing in the previous year.

gdpenl GDP per capita (measured as thousands of 1985 U.S. dollars) lagged one year.

ncontig equal to 1 for non-contiguous state.

nwstate equal to 1 for new state.

lpopl log(population size).

lmtnest log(mountainous).

ethfrac measure of ethnic fractionalization (calculated as the probability that two randomly drawn individuals from a country are not from the same ethnicity).

relfrac measure of religious fractionalization.

polity2l measure of political democracy (ranges from -10 to 10) lagged one year.

Source

Data are from:

Fearon J.D., Laitin D.D. (2003), Ethnicity, Insurgency, and Civil War. *The American Political Science Review*, 97, 75-90.

Examples

```
## Not run:

#########################################################
#########################################################

library("SemiParBIVProbit")

data("war", package = "SemiParBIVProbit")

#################################################
# Bivariate brobit model with partial observability
#################################################

reb.eq <- onset ~ instab + oil + warl + lpopl + lmtnest + ethfrac +
                  polity2l + s(gdpenl) + s(relfrac)
gov.eq <- onset ~ instab + oil + warl + ncontig + nwstate + s(gdpenl)

bpo <- SemiParBIVProbit(list(reb.eq, gov.eq), data = war, Model = "BPO")
conv.check(bpo)

# perhaps model is to complex

set.seed(1)
sbpo <- summary(bpo)
sbpo$theta; sbpo$CItheta

# let's exclude the correlation parameter in fitting

bpo0 <- SemiParBIVProbit(list(reb.eq, gov.eq), data = war, Model = "BPO0")
conv.check(bpo0)

summary(bpo0)
```

```
war.eq <- onset ~ instab + oil + warl +  ncontig + nwstate + lpopl +
                  lmtnest + ethfrac + polity2l + s(gdpenl) + s(relfrac)
Probit <- gam(war.eq, family = binomial(link = "probit"), data = war)
summary(Probit)


coef(Probit)[(which(names(coef(Probit)) == "s(gdpenl).9"))]

coef(bpo0)[(which(names(coef(bpo)) == "s(gdpenl).9"))]


probitW <- bpoW <- bpoReb <- bpoGov <- NA
gdp.grid <- seq(0, 8)

median.values <- data.frame(t(apply(war, 2, FUN = median)))

for (i in 1:length(gdp.grid)){

newd <- median.values; newd$gdpenl <- gdp.grid[i]
eta1 <- predict(bpo0, eq = 1, newd)
eta2 <- predict(bpo0, eq = 2, newd)
probitW[i] <- predict(Probit, newd, type = "response")
bpoW[i]    <- pnorm(eta1)*pnorm(eta2)
bpoReb[i]  <- pnorm(eta1)
bpoGov[i]  <- pnorm(eta2)

}


plot(gdp.grid, probitW, type = "l", ylim = c(0, 0.55), lwd = 2,
     col = "grey", xlab = "GDP per Capita (in thousands)",
     ylab = "Pr(Outcome)", main = "Probabilities for All Outcomes",
     cex.main = 1.5, cex.lab = 1.3, cex.axis = 1.3)
lines(gdp.grid, bpoW,   lwd = 2)
lines(gdp.grid, bpoReb, lwd = 2, lty = 2)
lines(gdp.grid, bpoGov, lwd = 2, lty = 3)

#dev.copy(postscript, "probWAR.eps", width = 8)
#dev.off()


## End(Not run)

#
```

`working.comp` *Internal Function*

Description

It efficiently calculates the working model quantities needed to implement the automatic multiple smoothing parameter estimation procedure by exploiting a result which leads to very fast and stable calculations.

Author(s)

Maintainer: Giampiero Marra <giampiero.marra@ucl.ac.uk>

Index