Package ‘sensmediation’

July 23, 2025

Title Parametric Estimation and Sensitivity Analysis of Direct and Indirect Effects

Version 0.3.1

Date 2024-09-26

Description We implement functions to estimate and perform sensitivity analysis to unobserved confounding of direct and indirect effects introduced in Lindmark, de Luna and Eriksson (2018) <doi:10.1002/sim.7620> and Lindmark (2022) <doi:10.1007/s10260-021-00611-4>. The estimation and sensitivity analysis are parametric, based on probit and/or linear regression models. Sensitivity analysis is implemented for unobserved confounding of the exposure-mediator, mediator-outcome and exposure-outcome relationships.

Depends R (>= 3.5.0)

Imports maxLik (>= 1.3-4), mvtnorm (>= 1.0-8), stats

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

Collate 'calc.effects.R' 'coefs.sensmed.R' 'data.R' 'mediationmethods.R' 'sensmediation.R'

Suggests testthat (>= 2.1.0)

NeedsCompilation no

Author Anita Lindmark [aut, cre]

Maintainer Anita Lindmark <`anita.lindmark@umu.se`>

Repository CRAN

Date/Publication 2024-09-26 22:20:02 UTC

Contents

`calc.effects` *Function for estimation of natural direct and indirect effects and sensitivity analysis for unobserved mediator-outcome confounding*

Description

Function to estimate natural direct and indirect effect estimates and standard errors (using the delta method) based on parametric regression models and perform sensitivity analysis for unobserved confounding. Intended to be called through `sensmediation` (or `more.effects`), not on its own.

Usage

```
calc.effects(
  ML.object,
  type = "my",
  exp.name,
  med.name,
  covariates = NULL,
  alt.decomposition = FALSE,
  exp.value = 1,
  control.value = 0,
  med.model = NULL,
  out.model = NULL
)
```

Arguments

`ML.object` object from `coefs.sensmed`

`type` the type of confounding for which the sensitivity analysis is to be performed. `type = "my"`, the default, corresponds to unobserved mediator-outcome confounding, `type = "zm"` to exposure-mediator confounding and `type = "zy"` to exposure-outcome confounding.

`exp.name` A character string indicating the name of the exposure variable used in the models.

`med.name` A character string indicating the name of the mediator used in the models.

`covariates` if conditional effects are to be estimated the list of covariate values. Covariates not specified are marginalized over. For more information, see `sensmediation`.

`alt.decomposition`
logical indicating whether alternative definitions of the direct and indirect effects should be used (for more information, see `sensmediation`).

`exp.value` value of the exposure variable used as the exposure condition, default is 1.

`control.value` value of the exposure variable used as the control (unexposed) condition, default is 0.

`med.model` If `type = "zy"`, fitted `glm` model object representing the mediator model at the basis of the estimation.

`out.model` If `type = "zm"`, fitted `glm` model object representing the outcome model at the basis of the estimation.

Value

A list with elements:

`effects` A list with elements `NIE` and `NDE`, row matrices with the estimated NIE and NDE (or NIE* and NDE* if `alt.decomposition = TRUE`) for each value of the sensitivity parameter `Rho`.

`std.errs` A list with elements `se.nie` and `se.nde`, row matrices with the estimated standard errors for the natural direct and indirect effects for the different values of the sensitivity parameter `Rho`.

`betas` list of the estimated mediator model parameters over `Rho`, with

- `beta0` Intercept
- `beta1` Exposure
- `beta2` Covariates
- `beta3` Exposure-covariate interactions

Components that are not included in the input mediator model are set to 0.

`thetas` list of the estimated outcome model parameters over `Rho`, with

- `theta0` Intercept
- `theta1` Exposure
- `theta2` Mediator
- `theta3` Exposure-mediator interaction
- `theta4` Covariates
- `theta5` Exposure-covariate interactions
- `theta6` Mediator-covariate interactions
- `theta7` Exposure-mediator-covariate interactions

Components that are not included in the input outcome model are set to 0.

`part.deriv` List with the partial derivatives of the NDE (Lambda), NIE (Gamma) and TE (Eta) wrt the mediator and outcome model parameters for each value of `Rho`

`sigma.thetabeta`
: a list with the joint covariance matrix of the outcome and mediator model parameters for each value of `Rho`. Note that the covariance matrix is constructed for all estimated parameters listed in `betas` and `thetas` but that components not included in the input mediator and outcome models are set to 0.

`covariates`
: list of the covariate values that the effects are conditioned on.

Author(s)

Anita Lindmark

See Also

`sensmediation`

`coefs.sensmed` *ML estimation of regression parameters for calculation of direct and indirect effects under unobserved confounding*

Description

This function gives ML estimates of the regression parameters used to calculate mediation effects and perform sensitivity analysis. The optimization is performed using `maxLik`, see Details for more information. Called by `sensmediation`.

Usage

```
coefs.sensmed(model.expl, model.resp, Rho, progress = TRUE, ...)
```

Arguments

`model.expl`
: Fitted `glm` model object. If sensitivity analysis to mediator-outcome confounding the mediator model. Otherwise the exposure model.

`model.resp`
: Fitted `glm` model object. If sensitivity analysis to exposure-mediator confounding the mediator model. Otherwise the outcome model.

`Rho`
: The sensitivity parameter vector. If `type="my"` the correlation between the error terms in the mediator and outcome models. If `type="zm"` the correlation between the error terms in the exposure and mediator models. If `type="zy"` the correlation between the error terms in the exposure and outcome models.

`progress`
: Logical, indicating whether or not the progress (i.e. the `proc.time` for each `Rho`) of the optimization will be output

`...`
: Additional arguments to be passed on to the `maxLik` function. Can be used to set the `method` and `control` arguments of the `maxLik` function.

Details

The maximization of the log-likelihood is performed using `maxLik`, the default is to use the Newton-Raphson method and an analytic gradient and Hessian.

Value

`coefs.sensmed` returns a list with elements:

`call` The matched call

`coef` A matrix with the estimated regression parameters for `model.resp` over the range of `Rho`. One column per value of `Rho`.

`sigma.res.resp` If `model.resp` is a linear regression model, the estimated standard deviation of the error term for each `Rho`.

`sigma.res.expl` If `model.expl` is a linear regression model, the estimated standard deviation of the error term for each `Rho`.

`Rho` The sensitivity parameter vector.

`expl.coef` A matrix with the estimated regression parameters for `model.expl` over the range of `Rho`. One column per value of `Rho`.

`model.expl` the original fitted `glm` object of `model.expl`.

`model.resp` the original fitted `glm` object of `model.resp`.

`X.expl` The model matrix (see `model.matrix`) of `model.expl`

`X.resp` The model matrix (see `model.matrix`) of `model.resp`

`outc.resp` The outcome variable of `model.resp`.

`outc.expl` The outcome variable of `model.expl`.

`sigmas` A list with the estimated covariance matrices for the regression parameters of `model.resp` and `model.expl` over `Rho`.

`max.info` Information about the maximization (whether or not the convergence was successful, `message`, `method` and number of iterations) for each `Rho`, see `maxLik` for more information.

`value` The values of the loglikelihood function for the best set of regression parameters from the optimization for each `Rho`, see `maxLik`.

Author(s)

Anita Lindmark

References

Henningsen, A., Toomet, O. (2011). maxLik: A Package for Maximum Likelihood Estimation in R, *Computational Statistics*, **26(3)**, pp. 443–458.

See Also

`sensmediation`

Examples

```
## Not run:
# Example with data from Riksstroke (the Swedish stroke register)

data(RSdata)

# Probit mediator and outcome models:
m.model <- glm(lowered.consc ~ AF + age.cat + sex, data = RSdata,
   family = binomial(link = 'probit'))
o.model <- glm(cf.3mo ~ AF + lowered.consc + age.cat + sex, data = RSdata,
   family = binomial(link = 'probit'))

# Estimation of regression coefficients under different values of Rho
# Rho = correlation between error terms in mediator and outcome model:
coefs.MY <- coefs.sensmed(model.expl = m.model, model.resp = o.model, Rho = seq(0, 0.5, 0.1))
# Outcome model regression coefficients:
coefs.MY$coef

## End(Not run)
```

`effects` *Functions to calculate natural direct and indirect effects.*

Description

Functions used to calculate natural direct and indirect effects based on the estimated regression parameters. Called by `calc.effects`. The functions are named according to the convention `eff."mediator model type""outcome model type"` where `b` stands for binary probit regression and `c` stands for linear regression.

Usage

```
eff.bb(
  Rho,
  betas,
  thetas,
  x.med,
  x.out,
  alt.decomposition,
  exp.value,
  control.value
)

eff.bc(
  Rho,
  betas,
  thetas,
  x.med,
  x.out,
  alt.decomposition,
  exp.value,
  control.value
)

eff.cb(
  Rho,
  betas,
  thetas,
  sigma.eta,
  x.med,
  x.out,
  alt.decomposition,
  exp.value,
  control.value
)

eff.cc(
  Rho,
  betas,
  thetas,
  x.med,
  x.out,
  alt.decomposition,
  exp.value,
  control.value
)
```

Arguments

`Rho` The sensitivity parameter vector.

`betas` List of mediator regression parameters

`thetas` List of outcome regression parameters

`x.med` Mediator covariate matrix for which to calculate standard errors

`x.out` Outcome covariate matrix for which to calculate standard errors

`alt.decomposition` logical indicating whether or not alternative definitions of the direct and indirect effects should be used.

`exp.value` value of the exposure variable used as the exposure condition.

`control.value` value of the exposure variable used as the control (unexposed) condition.

`sigma.eta` For a continuous mediator and binary outcome, matrix with the estimated residual standard deviation for the mediator model over the range of `Rho`.

grr *Analytic gradients of the loglikelihood functions for ML estimation of regression parameters*

Description

Implementation of the analytic gradients of the loglikelihood functions for ML estimation of regression parameters for different combinations of exposure, mediator and outcome models. The functions are named according to the convention `grr."model.expl type""model.resp type"` where `b` stands for binary probit regression and `c` stands for linear regression.

Usage

```
grr.bb(
  par,
  Rho,
  X.expl = X.expl,
  X.resp = X.resp,
  outc.resp = outc.resp,
  outc.expl = outc.expl
)

grr.bc(
  par,
  Rho,
  X.expl = X.expl,
  X.resp = X.resp,
  outc.resp = outc.resp,
  outc.expl = outc.expl
)

grr.cb(
  par,
  Rho,
  X.expl = X.expl,
  X.resp = X.resp,
  outc.resp = outc.resp,
  outc.expl = outc.expl
)

grr.cc(
  par,
  Rho,
  X.expl = X.expl,
  X.resp = X.resp,
  outc.resp = outc.resp,
  outc.expl = outc.expl
)
```

Arguments

par	Vector of parameter values.
Rho	The value of the sensitivity parameter.
X.expl	The model matrix (see model.matrix) of model.expl
X.resp	The model matrix (see model.matrix) of model.resp
outc.resp	The outcome of model.resp, a vector.
outc.expl	The outcome of model.expl, a column matrix.

See Also

coefs.sensmed, maxLik

hess *Analytic Hessians of the loglikelihood functions for ML estimation of regression parameters*

Description

Implementation of the analytic Hessians of the loglikelihood functions for ML estimation of regression parameters for different combinations of exposure, mediator and outcome models. The functions are named according to the convention hess."model.expl type""model.resp type" where b stands for binary probit regression and c stands for linear regression.

Usage

```
hess.bb(
  par,
  Rho,
  X.expl = X.expl,
  X.resp = X.resp,
  outc.resp = outc.resp,
  outc.expl = outc.expl
)

hess.bc(
  par,
  Rho,
  X.expl = X.expl,
  X.resp = X.resp,
  outc.resp = outc.resp,
  outc.expl = outc.expl
)
```

```
hess.cb(
  par,
  Rho,
  X.expl = X.expl,
  X.resp = X.resp,
  outc.resp = outc.resp,
  outc.expl = outc.expl
)

hess.cc(
  par,
  Rho,
  X.expl = X.expl,
  X.resp = X.resp,
  outc.resp = outc.resp,
  outc.expl = outc.expl
)
```

Arguments

`par` Vector of parameter values.

`Rho` The value of the sensitivity parameter.

`X.expl` The model matrix (see `model.matrix`) of `model.expl`

`X.resp` The model matrix (see `model.matrix`) of `model.resp`

`outc.resp` The outcome of `model.resp`, a vector.

`outc.expl` The outcome of `model.expl`, a column matrix.

See Also

`coefs.sensmed`, `maxLik`

LogL *Implementation of loglikelihood functions for ML estimation of regression parameters*

Description

Implementation of loglikelihood functions for ML estimation of regression parameters for different combinations of exposure, mediator and outcome models. The functions are named according to the convention `LogL."model.expl type""model.resp type"` where `b` stands for binary probit regression and `c` stands for linear regression.

Usage

```
LogL.bb(
  par,
  Rho,
  X.expl = X.expl,
  X.resp = X.resp,
  outc.resp = outc.resp,
  outc.expl = outc.expl
)

LogL.bc(
  par,
  Rho,
  X.expl = X.expl,
  X.resp = X.resp,
  outc.resp = outc.resp,
  outc.expl = outc.expl
)

LogL.cb(
  par,
  Rho,
  X.expl = X.expl,
  X.resp = X.resp,
  outc.resp = outc.resp,
  outc.expl = outc.expl
)

LogL.cc(
  par,
  Rho,
  X.expl = X.expl,
  X.resp = X.resp,
  outc.resp = outc.resp,
  outc.expl = outc.expl
)
```

Arguments

`par`	Vector of parameter values.
`Rho`	The value of the sensitivity parameter.
`X.expl`	The model matrix (see `model.matrix`) of `model.expl`
`X.resp`	The model matrix (see `model.matrix`) of `model.resp`
`outc.resp`	The outcome of `model.resp`, a vector.
`outc.expl`	The outcome of `model.expl`, a column matrix.

See Also

coefs.sensmed, maxLik

ML — *Functions for ML estimation of regression parameters for sensitivity analysis*

Description

Functions for ML estimation of regression parameters for sensitivity analysis for different combinations of exposure, mediator and outcome models. The functions are named according to the convention ML."model.expl type""model.resp type" where b stands for binary probit regression and c stands for linear regression. The optimization is performed using maxLik. The functions are intended to be called through coefs.sensmed, not on their own.

Usage

```
ML.bb(model.expl, model.resp, Rho, progress = TRUE, ...)

ML.bc(model.expl, model.resp, Rho, progress = TRUE, ...)

ML.cb(model.expl, model.resp, Rho, progress = TRUE, ...)

ML.cc(model.expl, model.resp, Rho, progress = TRUE, ...)
```

Arguments

model.expl	Fitted glm model object (probit or linear). If sensitivity analysis to mediator-outcome confounding the mediator model. Otherwise the exposure model.
model.resp	Fitted glm model object (probit or linear). If sensitivity analysis to exposure-mediator confounding the mediator model. Otherwise the outcome model.
Rho	The sensitivity parameter vector. If type="my" the correlation between the error terms in the mediator and outcome models. If type="zm" the correlation between the error terms in the exposure and mediator models. If type="zy" the correlation between the error terms in the exposure and outcome models.
progress	Logical, indicating whether or not the progress (i.e. the proc.time for each Rho) of the optimization will be output
...	Additional arguments to be passed on to the maxLik function. Can be used to set the method and control arguments of the maxLik function.

Value

A list with elements:

coef	A matrix with the estimated regression parameters for model.resp over the range of Rho. One column per value of Rho.

`Rho` The sensitivity parameter vector.

`expl.coef` A matrix with the estimated regression parameters for `model.expl` over the range of `Rho`. One column per value of `Rho`.

`model.expl` the original fitted `glm` object of `model.expl`.

`model.resp` the original fitted `glm` object of `model.resp`.

`X.expl` The model matrix (see `model.matrix`) of `model.expl`

`X.resp` The model matrix (see `model.matrix`) of `model.resp`

`outc.resp` The outcome variable of `model.resp`.

`outc.expl` The outcome variable of `model.expl`.

`sigma.res.expl` If `model.expl` is linear, a column matrix with the estimated residual standard deviation for `model.expl` over the range of `Rho`.

`sigma.res.resp` If `model.resp` is linear, a column matrix with the estimated residual standard deviation for `model.resp` over the range of `Rho`.

`value` The values of the -loglikelihood function for the best set of regression parameters from the optimization for each `Rho`.

`sigmas` A list with the covariance matrices for the model parameters in `model.expl` and `model.resp` for each `Rho`.

`max.info` Information about the maximization (whether or not the convergence was successful, `message`, `method` and number of iterations) for each `Rho`, see `maxLik` for more information.

Author(s)

Anita Lindmark

See Also

`coefs.sensmed`, `maxLik`

`more.effects` *Estimate additional natural direct and indirect effects based on an* `"effectsMed"` *object*

Description

Takes an `"effectsMed"` object and estimates additional natural direct and indirect effects, with a sensitivity analysis using the same sensitivity parameter as in the original analysis, without having to redo the optimization to find the estimated regression coefficients. The effects to be estimated are regulated through the arguments `covariates`, `alt.decomposition`, `exp.value` and `control.value` as described in the documentation for `sensmediation`. The confidence level used is regulated through the argument `conf.level`.

Usage

```
more.effects(
  sensmed.object,
  conf.level = 0.95,
  covariates = NULL,
  alt.decomposition = FALSE,
  exp.value = NULL,
  control.value = NULL
)
```

Arguments

`sensmed.object` an object of class "effectsMed" for which additional effects are to be calculated.

`conf.level` the confidence level to be used for confidence intervals and uncertainty intervals.

`covariates` if conditional effects are to be estimated the list of covariate values (see `sensmediation`). Covariates not specified are marginalized over.

`alt.decomposition` logical indicating whether alternative definitions of the direct and indirect effects should be used (see `sensmediation`).

`exp.value` value of the exposure variable used as the exposure condition, default is to take the value stored in `sensmed.object`.

`control.value` value of the exposure variable used as the control (unexposed) condition, default is to take the value stored in `sensmed.object`.

Value

`more.effects` returns an object of class `"effectsMed"`, see the documentation for `sensmediation` for information.

Author(s)

Anita Lindmark

See Also

`sensmediation`

Examples

```
## Not run:

# Example with data from Riksstroke (the Swedish stroke register)

data(RSdata)

# Probit mediator and outcome models:
med.model <- glm(lowered.consc ~ AF + age.cat + sex, data = RSdata,
family = binomial(link = 'probit'))
```

```
out.model <- glm(cf.3mo ~ AF + lowered.consc + age.cat + sex, data = RSdata,
family = binomial(link = 'probit'))

# First we estimate marginal NIE, NDE with sensitivity analyses to mediator-outcome
# confounding:
sensmed <- sensmediation(med.model, out.model, exp.name = "AF1", med.name = "lowered.consc",
Rho = seq(0, 0.5, 0.1))

# Then we also estimate NIE, NDE conditional on male sex without reestimating the regression
# coefficients:
sensmed.cond <- more.effects(sensmed.object = sensmed, covariates = list(sex = 1))
summary(sensmed.cond)
plot(sensmed.cond)

## End(Not run)
```

partdevs *Implementations of the partial derivatives (gradients) of the expressions for the direct, indirect and total effects. Used to calculate standard errors (delta method).*

Description

Functions implementing the partial derivatives (gradients) of the expressions for the direct, indirect and total effects. These are then used to calculate standard errors of the effects using the delta method. Called by the `stderrs` functions. The functions are named according to the convention `partdevs."mediator model type""outcome model type"` where bstands for binary probit regression and `c` stands for linear regression.

Usage

```
partdevs.bb(
  beta0,
  beta1,
  beta2,
  beta3,
  theta0,
  theta1,
  theta2,
  theta3,
  theta4,
  theta5,
  theta6,
  theta7,
  x.med,
  x.out,
  t.de,
  t.ie,
  exp.value,
  control.value
)

partdevs.bc(
  beta0,
  beta1,
  beta2,
  beta3,
  theta2,
  theta3,
  theta6,
  theta7,
  x.med,
  x.out,
  t.de,
  t.ie,
  exp.value,
  control.value
)

partdevs.cb(
  beta0,
  beta1,
  beta2,
  beta3,
  theta0,
  theta1,
  theta2,
  theta3,
  theta4,
  theta5,
  theta6,
  theta7,
  sigma.eta,
  x.med,
  x.out,
  t.de,
  t.ie,
  exp.value,
  control.value
)

partdevs.cc(
  beta0,
  beta1,
  beta2,
  beta3,
  theta2,
  theta3,
  theta6,
  theta7,
  exp.value,
  control.value,
  x.med,
  x.out,
  t.de,
  t.ie
)
```

Arguments

`beta0`, `beta1` Vectors of mediator regression parameters (intercept and exposure) over `Rho`

`beta2`, `beta3` Matrices of mediator regression parameters (covariate main effects and exposure-covariate interactions) over `Rho`

`theta0`, `theta1`, `theta2`, `theta3`
Vectors of outcome regression parameters (intercept, exposure, mediator, exposure-mediator interaction) over `Rho`

`theta4`, `theta5`, `theta6`, `theta7`
Matrices of outcome regression parameters (covariate main effects, exposure-covariate, mediator-covariate and exposure-mediator-covariate interactions) over `Rho`

`x.med` Mediator covariate matrix for which to calculate standard errors

`x.out` Outcome covariate matrix for which to calculate standard errors

`t.de`, `t.ie` exposure values used to calculate the direct and indirect effects depending on the desired decomposition (see the Details section of `sensmediation` for more information). If `alt.decomposition` = `TRUE` then `t.de` = `exp.value` and `t.ie` = `control.value`, otherwise `t.de` = `control.value` and `t.ie` = `exp.value`.

`exp.value` value of the exposure variable used as the exposure condition.

`control.value` value of the exposure variable used as the control (unexposed) condition.

`sigma.eta` For a continuous mediator and binary outcome, matrix with the estimated residual standard deviation for the mediator model over the range of `Rho`.

`plot.effectsMed` *Plot function for objects of class* `"effectsMed"`

Description

Plots the estimated natural indirect or direct effects with confidence intervals over the range of the sensitivity parameter `Rho`.

Usage

```
## S3 method for class 'effectsMed'
plot(
  x,
  effect = "indirect",
  xlab = NULL,
  ylab = NULL,
  xlim = NULL,
  ylim = NULL,
  main = NULL,
  lwd = graphics::par("lwd"),
  ...
)
```

Arguments

x	object of class "effectsMed"
effect	which effect to plot results for ("indirect" or "direct")
xlab	a title for the x axis, see title. Default is expression(rho).
ylab	a title for the y axis, see title. Default is NIE (NIE* if alt.decomposition = TRUE) or NDE (NDE*)
xlim	the x limits (x1, x2) of the plot, see plot.default. Default is c(min(x$Rho),max(x$Rho))
ylim	the y limits of the plot. Default is c(min(x$CI$CI.nie[,1]),max(x$CI$CI.nie[,2]))
main	a main title for the plot, see title
lwd	line widths for the lines of the plot, see par
...	additional graphical parameters to be passed to plotting functions, see par

RSdata *Example data for the functions in sensmediation*

Description

The data are a subsample of 1000 observations from Riksstroke, the Swedish Stroke Register. The original data consisted of over 50 000 patients with first time ischemic stroke during the years 2009-2012. The data are limited to patients over the age of 44 and its purpose is to illustrate the functioning of the functions in the package.

Usage

```
data(RSdata)
```

Format

A data frame with 1000 observations on the following 5 variables.

`cf.3mo` Outcome: case fatality within 3 months after stroke, 1 = deceased, 0 = not deceased.

`lowered.consc` Mediator: level of consciousness upon arrival to hospital. 1 = lowered consciousness, 0 = fully alert.

`AF` Exposure: atrial fibrillation. Factor with levels, "1" = atrial fibrillation, "0" = no atrial fibrillation.

`age.cat` Age at time of stroke. Factor with levels, "45-69", "70-79", "80-89" and "90-".

`sex` Factor with levels, "1" = male, "0" = female

Examples

```
data(RSdata)
```

`sensmediation` *Estimate natural direct and indirect effects based on parametric regression models and perform sensitivity analysis*

Description

Function to estimate the natural direct and indirect effects based on parametric regression models. Standard errors for the effects are calculated using the delta method. The function also gives sensitivity analysis results for unobserved confounding. Implements methods introduced in Lindmark, de Luna and Eriksson (2018).

Usage

```
sensmediation(
  med.model,
  out.model,
  exp.model = NULL,
  exp.name = NULL,
  med.name = NULL,
  type = "my",
  Rho = 0,
  progress = TRUE,
  conf.level = 0.95,
  covariates = NULL,
  alt.decomposition = FALSE,
  control.value = 0,
  exp.value = 1,
  covariance = NULL,
  med.full = NULL,
  out.full = NULL,
  all.interactions = NULL,
  ...
)
```

Arguments

`med.model` Fitted `glm` model object representing the mediator model at the basis of the estimation (see Details for more information).

`out.model` Fitted `glm` model object representing the outcome model at the basis of the estimation (see Details for more information).

`exp.model` Fitted `glm` model object representing the exposure model. Should be provided if `type="zm"` or `type="zy"`.

`exp.name` A character string indicating the name of the exposure variable used in the models. Needs to match the name of the exposure found in the output from the fitted glm-models (this is especially important to check for exposures of class `factor`).

`med.name` A character string indicating the name of the mediator used in the models. Needs to match the name of the mediator found in the output from the outcome glm-model (this is especially important to check for mediators of class `factor`).

`type` the type of confounding for which the sensitivity analysis is to be performed. `type="my"`, the default, corresponds to unobserved mediator-outcome confounding, `type="zm"` to exposure-mediator confounding and `type="zy"` to exposure-outcome confounding.

`Rho` The sensitivity parameter vector. If `type="my"` the correlation between the error terms in the mediator and outcome models. If `type="zm"` the correlation between the error terms in the exposure and mediator models. If `type="zy"` the correlation between the error terms in the exposure and outcome models.

`progress` Logical, indicating whether or not the progress (i.e. the `proc.time` for each `Rho`) of the optimization will be output

`conf.level` the confidence level to be used for confidence intervals and uncertainty intervals.

`covariates` if conditional effects are to be estimated the named list of covariate values (see Details). Covariates not specified are marginalized over.

`alt.decomposition`
logical indicating whether or not alternative definitions of the direct and indirect effects should be used (see Details).

`control.value` value of the exposure variable used as the control (unexposed) condition, default is 0.

`exp.value` value of the exposure variable used as the exposure condition, default is 1.

`covariance`, `med.full`, `out.full`, `all.interactions`
arguments used in previous versions of the package that are now deprecated.

`...` Additional arguments to be passed on to the `maxLik` function. Can be used to set the `method` and `control` arguments of the `maxLik` function (see `coefs.sensmed`).

Details

To obtain the ML estimates of the regression parameters used to calculate mediation effects and perform sensitivity analysis `sensmediation` calls `coefs.sensmed`. The maximization of the log-likelihood is performed using `maxLik`, the default is to use the Newton-Raphson method and an analytic gradient and Hessian.

The mediator and outcome models (and exposure model for `type = "zm"` or `"zy"`) should be fitted using `glm` and can be of two types, probit models (`family = binomial(link = 'probit')`) for binary mediators or outcomes (exposures) and linear regression models (`family = gaussian`) for continuous mediators or outcomes (exposures). Note that the exposure can either be binary or continuous, categorical exposures with more than two levels are not currently supported. The outcome model may contain exposure-mediator, exposure-covariate, mediator-covariate and exposure-mediator-covariate interactions. The mediator model may contain exposure-covariate interactions. All models may also contain interactions between covariates. Note, however that interactions may not be included in a model without also including the main effects of the interacting variables. That is, interactions should be specified either as `X1*X2` or `X1 + X2 + X1:X2`, not as `X1:X2` alone.

To obtain results conditional on specific covariate values, these values should be provided through the `covariates` argument as a named list (see Examples). The effects will be averaged over covariates not specified in the list.

The total effect can be decomposed into a direct and indirect effect in different ways. Let z be the exposure value and z* the control (unexposed) value. The default is to give the decomposition into the "pure direct effect" `E(Y(z,M(z*)))-E(Y(z*,M(z*)))` (here denoted NDE) and the "total indirect effect" `E(Y(z,M(z)))-E(Y(z,M(z*)))` (denoted NIE). Setting `alt.decomposition=TRUE` instead gives the decomposition into the "total direct effect" `E(Y(z,M(z)))-E(Y(z*,M(z)))` (here denoted NDE*) and "pure indirect effect" `E(Y(z*,M(z)))-E(Y(z*,M(z*)))` (denoted NIE*).

Standard errors for the effects are calculated using the delta method. Confidence intervals (CI) for (and p-values for tests of) the natural direct and indirect effects for each value of the sensitivity parameter are constructed based on a normal approximation. Uncertainty intervals (UI) are constructed as the union of all CIs over the sensitivity parameter vector.

Value

`sensmediation` returns an object of class `"effectsMed"`.

The function `summary` (`summary.effectsMed`) gives a summary of the results in the form of a table with the estimated effects and results of the sensitivity analysis. The function `plot` (`plot.effectsMed`) plots the estimated natural indirect or direct effects with confidence intervals over the range of the sensitivity parameter.

`call` The matched call

`Rho` The sensitivity parameter vector.

`type` character, the type of confounding the sensitivity analysis is performed for.

`coefs.sensmed` a list with the output from `coefs.sensmed`

`NIE` matrix with the estimated NIEs (or NIE*s if `alt.decomposition=TRUE`) over the range of the sensitivity parameter `Rho`.

`NDE` matrix with the estimated NDEs (or NDE*s if `alt.decomposition=TRUE`) over the range of the sensitivity parameter `Rho`.

`std.errs` list with the standard errors of the NIE (NIE*), NDE (NDE*) and total effect over the range of the sensitivity parameter `Rho`.

`CI` a list with the confidence intervals of the NIE (NIE*), NDE (NDE*) and total effect over the range of the sensitivity parameter `Rho`.

`UI` matrix with the uncertainty intervals for the NIE (NIE*) and NDE (NDE*) over the range of the sensitivity parameter `Rho`.

`conf.level` numeric, the confidence level used for confidence intervals and uncertainty intervals.

`covariates` list of the covariate values that the effects are conditioned on.

`exp.name` character vector containing the name of the exposure variable.

`med.name` character vector containing the name of the mediator variable.

`exp.value` value of the exposure variable used as the exposure condition.

`control.value` value of the exposure variable used as the control (unexposed) condition.

`alt.decomposition`
logical, indicating whether the alternative definitions of the direct and indirect effects have been used

`med.model` the mediator model input.

`out.model` the outcome model input.

`betas` list of the estimated mediator model parameters over `Rho`, with

- `beta0` Intercept
- `beta1` Exposure
- `beta2` Covariates
- `beta3` Exposure-covariate interactions

Components that are not included in the input mediator model are set to 0.

`thetas` list of the estimated outcome model parameters over `Rho`, with

- `theta0` Intercept
- `theta1` Exposure
- `theta2` Mediator
- `theta3` Exposure-mediator interaction
- `theta4` Covariates
- `theta5` Exposure-covariate interactions
- `theta6` Mediator-covariate interactions
- `theta7` Exposure-mediator-covariate interactions

Components that are not included in the input outcome model are set to 0.

`part.deriv` List with the partial derivatives of the NDE (Lambda), NIE (Gamma) and TE (Eta) wrt the mediator and outcome model parameters for each value of `Rho`. See `partdevs`.

`sigma.thetabeta`
a list with the joint covariance matrix of the outcome and mediator model parameters for each value of `Rho`. Note that the covariance matrix is constructed for all estimated parameters listed in `betas` and `thetas` but that components not included in the input mediator and outcome models are set to 0.

Author(s)

Anita Lindmark

References

Lindmark, A., de Luna, X., Eriksson, M. (2018) Sensitivity Analysis for Unobserved Confounding of Direct and Indirect Effects Using Uncertainty Intervals, *Statistics in Medicine*, **37(10)**, pp 1744–1762, doi:10.1002/sim.7620.

Lindmark A (2022). Sensitivity analysis for unobserved confounding in causal mediation analysis allowing for effect modification, censoring and truncation. *Statistical Methods & Applications*, **31**, pp 785–814, doi:10.1007/s10260-021-00611-4.

See Also

`more.effects` which can be used to calculate additional direct and indirect effects with sensitivity analysis using the same sensitivity parameter without running the optimization again.

Examples

```
# Example with data from Riksstroke (the Swedish stroke register)

data(RSdata)

# Probit mediator and outcome models:
m.model <- glm(lowered.consc ~ AF + age.cat + sex, data = RSdata,
   family = binomial(link = 'probit'))
o.model <- glm(cf.3mo ~ AF + lowered.consc + age.cat + sex, data = RSdata,
   family = binomial(link = 'probit'))

# Estimation of NIE, NDE and sensitivity analyses to mediator-outcome confounding:
# (note that the name of the exposure is "AF1" to match the name in coef(out.model))
sensmed <- sensmediation(m.model, o.model, exp.name = "AF1", med.name = "lowered.consc",
   Rho = c(0, 0.1))
summary(sensmed)
plot(sensmed)
plot(sensmed, effect = "direct")

## Not run:
# Conditional effects and sensitivity analysis to mediator-outcome confounding using
# more.effects():
sensmed.cond <- more.effects(sensmed.object = sensmed,
   covariates = list(sex = 1, age.cat = "70-79"))
summary(sensmed.cond)

## End(Not run)

## Not run:
## Sensitivity analysis to exposure-mediator confounding:
  e.model <- glm(AF ~ age.cat + sex, data = RSdata,
     family = binomial(link = 'probit'))

  sensmed.zm <- sensmediation(med.model = m.model, out.model = o.model,
     exp.model = e.model, type = "zm", Rho = seq(0, 0.5, 0.1), exp.name = "AF1",
     med.name = "lowered.consc")
  summary(sensmed.zm)

## End(Not run)

## Not run:
# Additional effects using more.effects:
# Results with conf.level = 0.99:
sensmed.zm.99 <- more.effects(sensmed.object = sensmed.zm, conf.level = 0.99)
summary(sensmed.zm.99)

## End(Not run)

## Not run:
# Examples with simulated data, continuous exposure:

require(mvtnorm)

n <- 1000
set.seed(102677)

x <- rnorm(n)
z <- -0.5 + 0.1*x + rnorm(n)
R <- 0.5
Sigma <- cbind(c(1,R), c(R,1))
epsilon <- rmvnorm(n, sigma = Sigma)
m <- -1.2 + 0.8*z + 0.13*x + epsilon[,1]
y <- -1 + 0.05*z + 3*m + 0.5*x + epsilon[,2]

# Models:
z.model <- glm(z ~ x)
m.model2 <- glm(m ~ z + x)
y.model <- glm(y ~ z + m + x)

## Estimation of NIE, NDE. Note that the exposure condition is 2
## so effects are calculated for a 2 unit increase of the exposure:
eff.contz <- sensmediation(med.model = m.model2, out.model = y.model,
           exp.name = "z", med.name = "m", control.value = 0, exp.value = 2)
summary(eff.contz)

## End(Not run)
```

`stderrs` *Functions to calculate standard errors of the direct, indirect and total effects using the delta method.*

Description

Functions used to calculate standard errors of the direct, indirect and total effects using the delta method. Called by `calc.effects`. The functions are named according to the convention `stderr."mediator` model type""outcome model type" where b stands for binary probit regression and c stands for linear regression.

Usage

```
stderr.bb(
  Rho,
  betas,
  thetas,
  sigma.pars,
  x.med,
  x.out,
  alt.decomposition,
  exp.value,
  control.value
)

stderr.bc(
  Rho,
  betas,
  thetas,
  sigma.pars,
  x.med,
  x.out,
  alt.decomposition,
  exp.value,
  control.value
)

stderr.cb(
  Rho,
  betas,
  thetas,
  sigma.eta,
  sigma.pars,
  x.med,
  x.out,
  alt.decomposition,
  exp.value,
  control.value
)

stderr.cc(
  Rho,
  betas,
  thetas,
  sigma.pars,
  x.med,
  x.out,
  alt.decomposition,
  exp.value,
  control.value
)
```

Arguments

`Rho` The sensitivity parameter vector.

`betas` List of mediator regression parameters

`thetas` List of outcome regression parameters

`sigma.pars` List of covariance matrices for the mediator and outcome regression parameters

`x.med` Mediator covariate matrix for which to calculate standard errors

`x.out` Outcome covariate matrix for which to calculate standard errors

`alt.decomposition` logical indicating whether or not alternative definitions of the direct and indirect effects should be used.

`exp.value` value of the exposure variable used as the exposure condition.

`control.value` value of the exposure variable used as the control (unexposed) condition.

`sigma.eta` For a continuous mediator and binary outcome, matrix with the estimated residual standard deviation for the mediator model over the range of `Rho`.

`summary.effectsMed` *Summary function for objects of class* `"effectsMed"`

Description

Summary function for objects of class `"effectsMed"`

Usage

```
## S3 method for class 'effectsMed'
summary(object, non.sign = FALSE, ...)

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

Arguments

`object` object of class `"effectsMed"`

`non.sign` logical indicating whether sensitivity analysis results should be printed for non-significant effects.

`...` additional arguments

`x` object of class `"summaryeffectsMed"`

`digits` number of digits to be printed.

Value

A list with values:

call The matched call

Rho The sensitivity parameter vector.

type character, the type of confounding the sensitivity analysis is performed for.

conf.level numeric, the confidence level used for confidence intervals and uncertainty intervals.

UI matrix with the uncertainty intervals for the NIE (NIE*) and NDE (NDE*) over the range of the sensitivity parameter Rho.

covariates list of the covariate values that the effects are conditioned on.

exp.name character vector containing the name of the exposure.

med.name character vector containing the name of the mediator.

alt.decomposition
logical, indicating whether the alternative definitions of the direct and indirect effects have been used

non.sign logical indicating whether sensitivity analysis results are printed for non-significant effects.

effects Results of the mediation analysis. Estimated NIE and NDE with confidence intervals and p-values for Rho = 0

ns.nie values of Rho with estimated NIEs and confidence intervals where the NIE is not significant.

ns.nde values of Rho with estimated NDEs and confidence intervals where the NDE is not significant.

rev.nie values of Rho with estimated NIEs and confidence intervals where the NIE is reversed.

rev.nde values of Rho with estimated NDEs and confidence intervals where the NDE is reversed.

Index