Package ‘episensr’

July 22, 2025

Type Package

Title Basic Sensitivity Analysis of Epidemiological Results

Version 2.0.0

Date 2025-05-14

Description Basic sensitivity analysis of the observed relative risks adjusting for unmeasured confounding and misclassification of the exposure/outcome, or both. It follows the bias analysis methods and examples from the book by Fox M.P., MacLehose R.F., and Lash T.L. ``Applying Quantitative Bias Analysis to Epidemiologic Data, second ed.'', ('Springer', 2021).

License GPL-2

URL https://codeberg.org/dhaine/episensr,
https://dhaine.codeberg.page/episensr/

BugReports https://codeberg.org/dhaine/episensr/issues

Depends ggplot2 (>= 3.5.0), R (>= 4.3.0)

Imports actuar, boot, cli, dagitty, forcats, ggdag, magrittr, trapezoid, triangle, truncnorm, MASS, lifecycle

Suggests aplore3, covr, directlabels, knitr, lattice, rmarkdown, testthat, tidyr

VignetteBuilder knitr

Encoding UTF-8

RoxygenNote 7.3.2

NeedsCompilation no

Author Denis Haine [aut, cre] (ORCID: <https://orcid.org/0000-0002-6691-7335>)

Maintainer Denis Haine <cheval@zaclys.net>

Repository CRAN

Date/Publication 2025-05-15 14:20:02 UTC

Contents

`boot.bias` *Bootstrap resampling for selection and misclassification bias models.*

Description

Generate R bootstrap replicates of either selection or misclassification bias functions. It then generates a confidence interval of the parameter, by first order normal approximation or the bootstrap percentile interval. Replicates giving negative cell(s) in the adjusted 2-by-2 table are silently ignored.

Usage

```
boot.bias(bias_model, R = 1000, conf = 0.95, ci_type = c("norm", "perc"))
```

Arguments

`bias_model`	An object of class "episensr.boot", i.e. either selection bias function or misclassification bias function.
`R`	The number of bootstrap replicates.
`conf`	Confidence level.
`ci_type`	A character string giving the type of interval required. Values can be either "norm" or "perc", default to "norm".

Value

A list with elements:

`model`	Model ran.
`boot_mod`	Bootstrap resampled object, of class `boot`.
`nrep`	Number of replicates used.
`bias_ciRR`	Bootstrap confidence interval object for relative risk.
`bias_ciOR`	Bootstrap confidence interval object for odds ratio.
`ci`	Confidence intervals for the bias adjusted association measures.
`conf`	Confidence interval.

See Also

`selection`, `misclass`

Examples

```
misclass_eval <- misclass(matrix(c(215, 1449, 668, 4296),
dimnames = list(c("Breast cancer+", "Breast cancer-"),
c("Smoker+", "Smoker-")),
nrow = 2, byrow = TRUE),
type = "exposure",
bias_parms = c(.78, .78, .99, .99))

set.seed(123)
boot.bias(misclass_eval)
```

`confounders` *Uncontrolled confounding*

Description

`confounders()` and `probsens_conf()` allow to provide adjusted measures of association corrected for unknown or unmeasured confounding without effect modification.

Usage

```
confounders(
  case,
  exposed,
  type = c("RR", "OR", "RD"),
  bias_parms = NULL,
  alpha = 0.05
)

confounders.emm(
  case,
  exposed,
  type = c("RR", "OR", "RD"),
  bias_parms = NULL,
  alpha = 0.05
)

confounders.poly(
  case,
  exposed,
  type = c("RR", "OR", "RD"),
  bias_parms = NULL,
  alpha = 0.05
)

probsens_conf(
  case,
  exposed,
  reps = 1000,
  prev_exp = list(dist = c("constant", "uniform", "triangular", "trapezoidal", "normal",
    "beta"), parms = NULL),
  prev_nexp = list(dist = c("constant", "uniform", "triangular", "trapezoidal", "normal",
    "beta"), parms = NULL),
  risk = list(dist = c("constant", "uniform", "triangular", "trapezoidal",
    "log-logistic", "log-normal"), parms = NULL),
  corr_p = NULL,
  alpha = 0.05
)
```

Arguments

`case`	Outcome variable. If a variable, this variable is tabulated against.
`exposed`	Exposure variable.
`type`	Choice of implementation, with no effect measure modification for ratio measures (relative risk – RR; odds ratio – OR) or difference measures (risk difference – RD).
`bias_parms`	Numeric vector defining the 3, 4, or 6 necessary bias parameters.

- This vector has 3 elements for the `confounders()` function, in the following order:
  1. the association between the confounder and the outcome among those who were not exposed (RR, OR, or RD according to choice of implementation),
  2. the prevalence of the confounder among the exposed (between 0 and 1), and
  3. the prevalence of the confounder among the unexposed (between 0 and 1).
- This vector has 4 elements for the `confounders.emm()` function, in the following order:
  1. the association between the confounder and the outcome among those who were exposed,
  2. the association between the confounder and the outcome among those who were not exposed,
  3. the prevalence of the confounder among the exposed (between 0 and 1), and
  4. the prevalence of the confounder among the unexposed (between 0 and 1).
- This vector has 6 elements for the `confounders.poly()` function, in the following order:
  1. the association between the highest level confounder and the outcome,
  2. the association between the mid-level confounder and the outcome,
  3. the prevalence of the highest level confounder among the exposed (between 0 and 1),
  4. the prevalence of the highest level confounder among the unexposed (between 0 and 1),
  5. the prevalence of the mid-level confounder among the exposed (between 0 and 1), and
  6. the prevalence of the mid-level confounder among the unexposed (between 0 and 1).

`alpha` Significance level.

`reps` Number of replications to run.

`prev_exp` List defining the prevalence of exposure among the exposed. The first argument provides the probability distribution function (constant, uniform, triangular, trapezoidal, truncated normal, or beta) and the second its parameters as a vector. Lower bound of the truncated normal cannot be less than zero. Upper bound is Inf by default.

1. constant: constant value,
2. uniform: min, max,
3. triangular: lower limit, upper limit, mode,
4. trapezoidal: min, lower mode, upper mode, max.
5. normal: lower bound, upper bound, mean, sd.
6. beta: alpha, beta.

prev_nexp — List defining the prevalence of exposure among the unexposed.

risk — List defining the confounder-disease relative risk or the confounder-exposure odds ratio. The first argument provides the probability distribution function (constant, uniform, triangular, trapezoidal, log-logistic, or log-normal) and the second its parameters as a vector:

1. constant: constant value,
2. uniform: min, max,
3. triangular: lower limit, upper limit, mode,
4. trapezoidal: min, lower mode, upper mode, max.
5. log-logistic: shape, rate. Must be strictly positive,
6. log-normal: meanlog, sdlog. This is the mean and standard deviation on the log scale.

corr_p — Correlation between the exposure-specific confounder prevalences.

Details

`confounders.emm()` allows to provide for adjusted measures of association corrected for unknown or unmeasured confounding in the presence of effect modification.

`confounders.poly()` allows to provide for adjusted measures of association corrected for unknown or unmeasured polychotomous (3-level) confounding without effect modification.

Value

A list with elements:

obs_data — The analyzed 2 x 2 table from the observed data.

cfder_data — The same table for Confounder +.

cfder1.data — The same table for Mid-level Confounder + (for `confounders.poly()`).

cfder2.data — The same table for Highest-level Confounder + (for `confounders.poly()`).

nocfder_data — The same table for Confounder -.

obs_measures — A table of relative risk with confidence intervals; for Total, Confounder +, and Confounder -.

adj_measures — A table of Standardized Morbidity Ratio and Mantel-Haenszel estimates.

bias_parms — Input bias parameters.

A list with elements (for `probsens_conf()`):

obs_data — The analyzed 2 x 2 table from the observed data.

obs_measures — A table of observed relative risk and odds ratio with confidence intervals.

adj_measures — A table of corrected relative risks and odds ratios.

sim_df — Data frame of random parameters and computed values.

reps — Number of replications.

Simple bias analysis with `confounders()`

`confounders()` allows you to run a simple sensitivity analysis to correct for unknown or unmeasured confounding without effect modification. Implementation for ratio measures (relative risk – RR, or odds ratio – OR) and difference measures (risk difference – RD).

The analytic approach uses the "relative risk due to confounding" as defined by Miettinen (1972), i.e. $RR_{adj} = \frac{RR_{crude}}{RR_{conf}}$ where RR_{adj} is the standardized (adjusted) risk ratio, RR_{crude} is the crude risk ratio, and RR_{conf} is the relative risk component attributable to confounding by the stratification factors. The output provides both RR_{adj} (SMR or Mantel-Haenszel) and the RR_{conf} (i.e., RR, OR or RD due to confounding from the unmeasured confounder).

Probabilistic sensitivity analysis with `probsens_conf()`

`probsens_conf()` performs a summary-level probabilistic sensitivity analysis to correct for unknown or unmeasured confounding and random error simultaneously. It returns the Mantel-Haenszel risk ratio.

Correlations between prevalences of exposure classification among cases and controls can be specified and use the NORmal To Anything (NORTA) transformation (Li & Hammond, 1975).

Simple bias analysis with `confounders.emm()`

`confounders.emm()` allows you to run a simple sensitivity analysis to correct for unknown or unmeasured confounding in the presence of effect modification. Implementation for ratio measures (relative risk – RR, or odds ratio – OR) and difference measures (risk difference – RD).

Simple bias analysis with `confounders.poly()`

`confounders.poly()` allows you to run a simple sensitivity analysis to correct for unknown or unmeasured polychotomous (3-level) confounding without effect modification. Implementation for ratio measures (relative risk – RR, or odds ratio – OR) and difference measures (risk difference – RD).

Updated calculations

episensr 2.0.0 introduced updated calculations of probabilistic bias analyses by (1) using the NORTA transformation to define a correlation between distributions, and (2) sampling true prevalences and then sampling the adjusted cell counts rather than just using the expected cell counts from a simple quantitative bias analysis. This updated version should be preferred but if you need to run an old analysis, you can easily revert to the computation using `probsens.conf_legacy()` as follows:

```
library(episensr)
probsens.conf <- probsens.conf_legacy
```

References

Fox, M.P, MacLehose, R.F., Lash, T.L., 2021 *Applying Quantitative Bias Analysis to Epidemiologic Data*, pp.105–140, 256–262, Springer.

Miettinen, 1971. Components of the Crude Risk Ratio. *Am J Epidemiol* 96(2):168-172.

Li, S.T., Hammond, J.L., 1975. *Generation of Pseudorandom Numbers with Specified Univariate Distributions and Correlation Coefficients*. IEEE Trans Syst Man Cybern 5:557-561.

See Also

Other confounding: `confounders.array()`, `confounders.evalue()`, `confounders.ext()`, `confounders.limit()`, `probsens.irr.conf()`

Examples

```
# The data for this example come from:
# Tyndall M.W., Ronald A.R., Agoki E., Malisa W., Bwayo J.J., Ndinya-Achola J.O.
# et al.
# Increased risk of infection with human immunodeficiency virus type 1 among
# uncircumcised men presenting with genital ulcer disease in Kenya.
# Clin Infect Dis 1996;23:449-53.
confounders(matrix(c(105, 85, 527, 93),
dimnames = list(c("HIV+", "HIV-"), c("Circ+", "Circ-")),
nrow = 2, byrow = TRUE),
type = "RR",
bias_parms = c(.63, .8, .05))

confounders(matrix(c(105, 85, 527, 93),
dimnames = list(c("HIV+", "HIV-"), c("Circ+", "Circ-")),
nrow = 2, byrow = TRUE),
type = "OR",
bias_parms = c(.63, .8, .05))

confounders(matrix(c(105, 85, 527, 93),
dimnames = list(c("HIV+", "HIV-"), c("Circ+", "Circ-")),
nrow = 2, byrow = TRUE),
type = "RD",
bias_parms = c(-.37, .8, .05))
#
confounders.emm(matrix(c(105, 85, 527, 93),
dimnames = list(c("HIV+", "HIV-"), c("Circ+", "Circ-")),
nrow = 2, byrow = TRUE),
type = "RR",
bias_parms = c(.4, .7, .8, .05))

confounders.emm(matrix(c(105, 85, 527, 93),
dimnames = list(c("HIV+", "HIV-"), c("Circ+", "Circ-")),
nrow = 2, byrow = TRUE),
type = "OR",
bias_parms = c(.4, .7, .8, .05))

confounders.emm(matrix(c(105, 85, 527, 93),
dimnames = list(c("HIV+", "HIV-"), c("Circ+", "Circ-")),
nrow = 2, byrow = TRUE),
type = "RD",
bias_parms = c(-.6, -.3, .8, .05))
#
```

```
# The data for this example come from:
# Tyndall M.W., Ronald A.R., Agoki E., Malisa W., Bwayo J.J., Ndinya-Achola J.O.
# et al.
# Increased risk of infection with human immunodeficiency virus type 1 among
# uncircumcised men presenting with genital ulcer disease in Kenya.
# Clin Infect Dis 1996;23:449-53.
confounders.poly(matrix(c(105, 85, 527, 93),
dimnames = list(c("HIV+", "HIV-"), c("Circ+", "Circ-")),
nrow = 2, byrow = TRUE),
type = "RR",
bias_parms = c(.4, .8, .6, .05, .2, .2))

confounders.poly(matrix(c(105, 85, 527, 93),
dimnames = list(c("HIV+", "HIV-"), c("Circ+", "Circ-")),
nrow = 2, byrow = TRUE),
type = "OR",
bias_parms = c(.4, .8, .6, .05, .2, .2))

confounders.poly(matrix(c(105, 85, 527, 93),
dimnames = list(c("HIV+", "HIV-"), c("Circ+", "Circ-")),
nrow = 2, byrow = TRUE),
type = "RD",
bias_parms = c(-.4, -.2, .6, .05, .2, .2))
#
# The data for this example come from:
# Tyndall M.W., Ronald A.R., Agoki E., Malisa W., Bwayo J.J., Ndinya-Achola J.O. et al.
# Increased risk of infection with human immunodeficiency virus type 1 among
# uncircumcised men presenting with genital ulcer disease in Kenya.
# Clin Infect Dis 1996;23:449-53.
tyndall <- matrix(c(105, 85, 527, 93),
dimnames = list(c("HIV+", "HIV-"), c("Circ+", "Circ-")), nrow = 2, byrow = TRUE)
set.seed(1234)
probsens_conf(tyndall, reps = 100000,
prev_exp = list("trapezoidal", c(.7, .75, .85, .9)),
prev_nexp = list("trapezoidal", c(.03, .04, .07, .1)),
risk = list("trapezoidal", c(.5, .6, .7, .8)))

set.seed(123)
probsens_conf(tyndall, reps = 20000,
prev_exp = list("beta", c(200, 56)),
prev_nexp = list("beta", c(10, 16)),
risk = list("triangular", c(.6, .7, .63)),
corr_p = .8)

set.seed(123)
probsens_conf(tyndall, reps = 20000,
prev_exp = list("normal", c(.01, .12, 0.03, 0.005)),
prev_nexp = list("normal", c(0, Inf, 0.01, 0.0001)),
risk = list("triangular", c(.6, .7, .63)), corr_p = .8)

# Fox M.P., MacLehose R.F., Lash T.L.
# SAS and R code for probabilistic quantitative bias analysis for
# misclassified binary variables and binary unmeasured confounders
```

```
# Int J Epidemiol 2023:1624-1633.
fox <- matrix(c(40, 20, 60, 80),
dimnames = list(c("Diseased", "Non-diseased"), c("Exposed", "Unexposed")),
nrow = 2, byrow = TRUE)
set.seed(1234)
probsens_conf(fox, reps = 10^5,
prev_exp = list("beta", c(10, 20)),
prev_nexp = list("beta", c(5, 20)),
risk = list("trapezoidal", c(1.5, 1.7, 2.3, 2.5)))

set.seed(1234)
probsens_conf(fox, reps = 20000,
prev_exp = list("beta", c(10, 20)),
prev_nexp = list("beta", c(5, 20)),
risk = list("log-normal", c(log(2), .23)))
```

`confounders.array` *Sensitivity analysis for unmeasured confounders based on confounding imbalance among exposed and unexposed*

Description

Sensitivity analysis to explore effect of residual confounding using simple algebraic transformation (array approach). It indicates the strength of an unmeasured confounder and the necessary imbalance among exposure categories to affect the observed (crude) relative risk.

Usage

```
confounders.array(
  crude_risk,
  type = c("binary", "continuous", "RD"),
  bias_parms = NULL
)
```

Arguments

`crude_risk` Crude (apparent or observed) relative risk between the exposure and the outcome. If type `RD`, this is the crude (observed) risk difference.

`type` Choice of implementation, for binary covariates, continuous covariates, or on risk difference scale.

`bias_parms` Numeric vector defining the necessary bias parameters. This vector has 3 elements, in the following order:

1. the association between the confounder and the outcome (RR, relative risk),
2. the prevalence of the confounder among the exposed (between 0 and 1, if type `binary`), or mean value of the confounder among the exposed (if type `continuous` or `RD`), and
3. the prevalence of the confounder among the unexposed (between 0 and 1, if type `binary`), or mean value of the confounder among the unexposed (if type `continuous` or `RD`).

Value

A list with elements:

`model`	Bias analysis performed.
`bias_parms`	Input bias parameters.
`adj_measures`	Output results, with bias as a percentage: (crude_RR - risk_adj)/risk_adj * 100.

References

Schneeweiss, S., 2006. Sensitivity analysis and external adjustment for unmeasured confounders in epidemiologic database studies of therapeutics. *Pharmacoepidemiol Drug Safety* 15: 291-303.

See Also

Other confounding: `confounders()`, `confounders.evalue()`, `confounders.ext()`, `confounders.limit()`, `probsens.irr.conf()`

Examples

```
# Example from Schneeweiss, S. Sensitivity analysis and external adjustment for
# unmeasured confounders in epidemiologic database studies of therapeutics.
# Pharmacoepidemiol Drug Safety 2006; 15: 291-303.
confounders.array(crude_risk = 1.5, type = "binary",
bias_parms = c(5.5, 0.5, 0.1))

# Examples from Patorno E., Gopalakrishnan, C., Franklin, J.M., Brodovicz, K.G.,
# Masso-Gonzalez, E., Bartels, D.B., Liu, J., and Schneeweiss, S. Claims-based
# studies of oral glucose-lowering medications can achieve balance in critical
# clinical variables only observed in electronic health records 2017; 20(4): 974-
# 984.
confounders.array(crude_risk = 1.5, type = "binary",
bias_parms = c(3.25, 0.333, 0.384))

confounders.array(crude_risk = 1.5, type = "continuous",
bias_parms = c(1.009, 7.8, 7.9))

confounders.array(crude_risk = 0.05, type = "RD", bias_parms = c(0.009, 8.5, 8))
```

`confounders.evalue` *Compute E-value to assess bias due to unmeasured confounder.*

Description

Help to quantify the evidence strength for causality in presence of unmeasured confounding. The E-value is the minimum strength of association that an unmeasured confounder would need to have with both the exposure and the outcome, conditional on the measured covariates, to fully explain away a specific exposure-outcome association.

Usage

```
confounders.evalue(
  est,
  lower_ci = NULL,
  upper_ci = NULL,
  sd = NA,
  type = c("RR", "ORc", "HRc", "diff_RR", "diff_OR"),
  true_est = 1
)
```

Arguments

`est`	Point estimate for the effect measure. For difference in continuous outcomes, it is the standardized effect size (i.e. mean of the outcome divided by its standard deviation).
`lower_ci`	Lower limit of the confidence interval for the association (relative risk, odds ratio, hazard ratio, incidence rate ratio, risk difference).
`upper_ci`	Upper limit of the confidence interval for the association (relative risk, odds ratio, hazard ratio, incidence rate ratio, risk difference).
`sd`	For difference in continuous outcomes, the standard error of the outcome divided by its standard deviation.
`type`	Choice of effect measure (relative risk, and odds ratio or hazard ratio for rare outcomes i.e. < 15% at end of follow-up – RR; odds ratio for common outcome – ORc; hazard ratio for common outcome i.e. > 15% at end of follow-up – HRc; difference in continuous outcomes, RR approximation – diff_RR; difference in continuous outcomes, OR approximation – diff_OR).
`true_est`	True estimate to assess E-value for. Default to 1 on risk scale to assess against null value. Set to a different value to assess for non-null hypotheses.

Value

A matrix with the observed point estimate and closest confidence interval to the null hypothesis, expressed as a relative risk, and their corresponding E-value.

References

VanderWeele T.J and Ding P. Sensitivity analysis in observational research: Introducing the E-value. Annals of Internal Medicine 2017;167:268-274.

See Also

Other confounding: `confounders()`, `confounders.array()`, `confounders.ext()`, `confounders.limit()`, `probsens.irr.conf()`

Examples

```
# The data for this example come from:
# Victoria C.G., Smith P.G., Vaughan J.P., Nobre L.C., Lombardi C., Teixeira A.M.
# et al.
# Evidence for protection by breast-feeding against infant deaths from infectious
# diseases in Brazil.
# Lancet 1987;2:319-22.
confounders.evalue(est = 3.9, type = "RR")

# The data for this example come from:
# Oddy W.H, Smith G.J., Jacony P.
# A possible strategy for developing a model to account for attrition bias in a
# longitudinal cohort to investigate associations between exclusive breastfeeding and
# overweight and obesity at 20 years.
# Annals of Nutrition and Metabolism 2014;65:234-235.
confounders.evalue(est = 1.47, lower_ci = 1.12, upper_ci = 1.93, type = "ORc")

# The data for this example come from:
# Reinisch J., Sanders S., Mortensen E., Rubin D.B.
# In-utero exposure to phenobarbital and intelligence deficits in adult men.
# Journal of the American Medical Association 1995;274:1518-1525
confounders.evalue(est = -0.42, sd = 0.14, type = "diff_RR")
```

---

confounders.ext — *Sensitivity analysis for unmeasured confounders based on external adjustment*

Description

Sensitivity analysis to explore effect of residual confounding using simple algebraic transformation. It provides the relative risk adjusted for unmeasured confounders based on available external information (i.e. from the literature) on the relation between confounders and outcome.

Usage

```
confounders.ext(RR, bias_parms = NULL)
```

Arguments

`RR` "True" or fully adjusted exposure relative risk.

`bias_parms` Numeric vector defining the necessary bias parameters. This vector has 4 elements, in the following order:

1. the association between the confounder and the outcome (RR, relative risk),
2. the association between exposure category and the confounder (OR, odds ratio),
3. the prevalence of the confounder (between 0 and 1), and
4. the prevalence of the exposure (between 0 and 1).

Value

A list with elements:

`model`	Bias analysis performed.
`bias_parms`	Input bias parameters.
`adj_measures`	Output results, with bias as a percentage: (crude_RR - RR)/RR * 100.

References

Schneeweiss, S., 2006. Sensitivity analysis and external adjustment for unmeasured confounders in epidemiologic database studies of therapeutics. *Pharmacoepidemiol Drug Safety* 15: 291-303.

See Also

Other confounding: `confounders()`, `confounders.array()`, `confounders.evalue()`, `confounders.limit()`, `probsens.irr.conf()`

Examples

```
# Schneeweiss, S, Glynn, R.J., Tsai, E.H., Avorn, J., Solomon, D.H. Adjusting for
# unmeasured confounders in pharmacoepidemiologic claims data using external
# information. Epidemiology 2005; 16: 17-24.
confounders.ext(RR = 1, bias_parms = c(0.1, 0.9, 0.1, 0.4))
```

`confounders.limit` *Bounding the bias limits of unmeasured confounding.*

Description

Function to elicit the limits on measures of effect corrected for an unmeasured confounder when only some of the bias parameters are known. Crude relative risk between exposure and outcome has minimally to be provided. Up to 3 other parameters can be entered.

Usage

```
confounders.limit(p = NA, RR = NA, OR = NA, crude_RR = NULL)
```

Arguments

`p`	Proportion with the confounder among the unexposed group.
`RR`	Relative risk between the confounder and the outcome.
`OR`	Odds ratio between the confounder and the outcome.
`crude_RR`	Crude relative risk between the exposure and the outcome.

Value

A list with elements:

model	Bias analysis performed.
bias_parms	Input bias parameters.
adj_measures	Output results.

References

Fox, M.P, MacLehose, R.F., Lash, T.L., 2021 *Applying Quantitative Bias Analysis to Epidemiologic Data*, pp.129–131, Springer.

Flanders, W. Dana, Khoury, Muin J., 1990. Indirect Assessment of Confounding: Graphic Description and Limits on Effect of Adjusting for Covariates. *Epidemiology* 1(3): 239–246.

See Also

Other confounding: `confounders()`, `confounders.array()`, `confounders.evalue()`, `confounders.ext()`, `probsens.irr.conf()`

Examples

```
confounders.limit(OR = 1.65, crude_RR = 1.5)
```

---

mbias — *Sensitivity analysis to correct for selection bias caused by M bias.*

Description

Simple sensitivity analysis to correct for selection bias caused by M bias using estimates of the odds ratios relating the variables.

Usage

```
mbias(or, var = c("y", "x", "a", "b", "m"))
```

Arguments

`or` Vector defining the input bias parameters, in the following order:

1. Odds ratio between A and the exposure E,
2. Odds ratio between A and the collider M,
3. Odds ratio between B and the collider M,
4. Odds ratio between B and the outcome D,
5. Odds ratio observed between the exposure E and the outcome D.

`var` Vector defining variable names, in the following order:

1. Outcome,

2. Exposure,
3. A,
4. B,
5. Collider.

Value

A list with elements:

`model`	Bias analysis performed.
`mbias.parms`	Three maximum bias parameters: in collider-exposure relationship created by conditioning on the collider, in collider-outcome relationship created by conditioning on the collider, and in exposure-outcome relationship created by conditioning on the collider.
`adj.measures`	Selection bias corrected odds ratio.
`bias.parms`	Input bias parameters.
`labels`	Variables' labels.

References

Greenland S. Quantifying biases in causal models: classical confounding vs. collider-stratification bias. Epidemiology 2003;14:300-6.

See Also

Other selection: `selection()`

Examples

```
mbias(or = c(2, 5.4, 2.5, 1.5, 1),
var = c("HIV", "Circumcision", "Muslim", "Low CD4", "Participation"))
```

`misclass` *Misclassification of exposure or outcome*

Description

`misclass()` and `probsens()` allow to provide adjusted measures of association corrected for misclassification of the exposure or the outcome.

Usage

```
misclass(
  case,
  exposed,
  type = c("exposure", "exposure_pv", "outcome"),
  bias_parms = NULL,
  alpha = 0.05
)

probsens(
  case,
  exposed,
  type = c("exposure", "exposure_pv", "outcome"),
  reps = 1000,
  seca = list(dist = c("constant", "uniform", "triangular", "trapezoidal", "normal",
    "beta"), parms = NULL),
  seexp = NULL,
  spca = list(dist = c("constant", "uniform", "triangular", "trapezoidal", "normal",
    "beta"), parms = NULL),
  spexp = NULL,
  corr_se = NULL,
  corr_sp = NULL,
  alpha = 0.05
)
```

Arguments

`case` Outcome variable. If a variable, this variable is tabulated against.

`exposed` Exposure variable.

`type` Choice of misclassification:

1. exposure: bias analysis for exposure misclassification; corrections using sensitivity and specificity: nondifferential and independent errors,
2. exposure_pv: bias analysis for exposure misclassification; corrections using PPV/NPV: nondifferential and independent errors,
3. outcome: bias analysis for outcome misclassification.

`bias_parms` Vector defining the bias parameters. This vector has 4 elements between 0 and 1, in the following order:

1. Sensitivity of exposure (when `type = "exposure"`) or outcome (when `type = "outcome"`) classification among those with the outcome (when `type = "exposure"`) or exposure (when `type = "outcome"`),
2. Sensitivity of exposure (or outcome) classification among those without the outcome (or exposure),
3. Specificity of exposure (or outcome) classification among those with the outcome (or exposure), and
4. Specificity of exposure (or outcome) classification among those without the outcome (or exposure).

If PPV/NPV is chosen in case of exposure misclassification, this vector is the following:

1. Positive predictive value among those with the outcome,
2. Positive predictive value among those without the outcome,
3. Negative predictive value among those with the outcome,
4. Negative predictive value among those without the outcome.

`alpha` Significance level.

`reps` Number of replications to run.

`seca` List defining sensitivity among cases:

1. The sensitivity of exposure classification among those with the outcome (when `type = "exposure"`), or
2. The sensitivity of outcome classification among those with the exposure (when `type = "outcome"`).

The first argument provides the probability distribution function (constant, uniform, triangular, trapezoidal, truncated normal, or beta) and the second its parameters as a vector. Lower and upper bounds of the truncated normal have to be between 0 and 1.

1. constant: constant value,
2. uniform: min, max,
3. triangular: lower limit, upper limit, mode,
4. trapezoidal: min, lower mode, upper mode, max,
5. normal: lower bound, upper bound, mean, sd.
6. beta: alpha, beta.

If PPV/NPV is chosen in case of exposure misclassification, the same four (4) parameters `seca`, `seexp`, `spca`, `spexp` as for Se/Sp have to be used but with the following meaning, and only for a beta distributions and no correlation between distributions:

1. Positive predictive value among those with the outcome,
2. Positive predictive value among those without the outcome,
3. Negative predictive value among those with the outcome,
4. Negative predictive value among those without the outcome.

`seexp` List defining sensitivity among controls:

1. The sensitivity of exposure classification among those without the outcome (when `type = "exposure"`), or
2. The sensitivity of outcome classification among those without the exposure (when `type = "outcome"`).

`spca` List as above for `seca` but for specificity.

`spexp` List as above for `seexp` but for specificity.

`corr_se` Correlation between case and non-case sensitivities. If PPV/NPV is chosen in case of exposure misclassification, correlations are set to NULL.

`corr_sp` Correlation between case and non-case specificities.

Value

A list with elements (for `misclass()`):

`obs_data`	The analyzed 2 x 2 table from the observed data.
`corr_data`	The expected observed data given the true data assuming misclassification.
`obs_measures`	A table of observed relative risk and odds ratio with confidence intervals.
`adj_measures`	A table of corrected relative risks and odds ratios.
`bias_parms`	Input bias parameters.

A list with elements (for `probsens()`):

`obs_data`	The analyzed 2 x 2 table from the observed data.
`obs_measures`	A table of observed relative risk and odds ratio with confidence intervals.
`adj_measures`	A table of corrected relative risks and odds ratios.
`sim_df`	Data frame of random parameters and computed values.
`reps`	Number of replications.

Simple bias analysis with `misclass()`

`misclass()` allows you to run a simple sensitivity analysis for disease or exposure misclassification. Confidence interval for odds ratio adjusted using sensitivity and specificity is computed as in Chu et al. (2006), for exposure misclassification.

For exposure misclassification, bias-adjusted measures are available using sensitivity and specificity, or using predictive values.

Probabilistic sensitivity analysis with `probsens()`

`probsens()` performs a summary-level probabilistic sensitivity analysis to correct for exposure misclassification or outcome misclassification and random error. Non-differential misclassification is assumed when only the two bias parameters `seca` and `spca` are provided. Adding the 2 parameters `seexp` and `spexp` (i.e. providing the 4 bias parameters) evaluates a differential misclassification.

For exposure misclassification, bias-adjusted measures are available using sensitivity and specificity, or using predictive values. However, only a beta distribution is available for predictive values.

Correlations between sensitivity (or specificity) of exposure classification among cases and controls can be specified and use the NORmal To Anything (NORTA) transformation (Li & Hammond, 1975).

In case of negative (<=0) adjusted count in the 2-by-2 table following given prior Se/Sp distribution(s), draws are discarded.

Updated calculations, probabilistic bias analysis

episensr 2.0.0 introduced updated calculations of probabilistic bias analyses by (1) using the NORTA transformation to define a correlation between distributions, and (2) sampling true prevalences and then sampling the adjusted cell counts rather than just using the expected cell counts from a simple quantitative bias analysis. This updated version should be preferred but if you need to run an old analysis, you can easily revert to the computation using `probsens_legacy()` as follows:

```
library(episensr)
probsens <- probsens_legacy
```

References

Fox, M.P, MacLehose, R.F., Lash, T.L., 2021 *Applying Quantitative Bias Analysis to Epidemiologic Data*, pp.141–176, 233–256, 293–308, Springer.

Li, S.T., Hammond, J.L., 1975. *Generation of Pseudorandom Numbers with Specified Univariate Distributions and Correlation Coefficients*. IEEE Trans Syst Man Cybern 5:557-561.

Chu, H., Zhaojie, W., Cole, S.R., Greenland, S., *Sensitivity analysis of misclassification: A graphical and a Bayesian approach*, Annals of Epidemiology 2006;16:834-841.

Barros, A. & Hirakata, V.N., 2003. Alternatives for Logistic Regression in Cross-sectional Studies: An Empirical Comparison of Models that Directly Estimate the Prevalence Ratio. BMC Medical Research Methodology 3:21.

McNutt, L-A, Wu, C., Xue, X., Hafner J.P., 2003. *Estimating the Relative Risk in Cohort Studies and Clinical Trials of Common Outcomes*. American Journal of Epidemiology 157(10):940-943.

Greenland, S. (2004). Model-based Estimation of Relative Risks and Other Epidemiologic Measures in Studies of Common Outcomes and in Case-Control Studies. American Journal of Epidemiology 160(4):301-305.

Zhou, G. (2004). A Modified Poisson Regression Approach to Prospective Studies with Binary Data. American Journal of Epidemiology 159(7):702-706.

See Also

Other misclassification: `misclass_cov()`, `probsens.irr()`

Examples

```
# The data for this example come from:
# Fink, A.K., Lash,  T.L. A null association between smoking during pregnancy
# and breast cancer using Massachusetts registry data (United States).
# Cancer Causes Control 2003;14:497-503.
misclass(matrix(c(215, 1449, 668, 4296),
dimnames = list(c("Breast cancer+", "Breast cancer-"),
c("Smoker+", "Smoker-")),
nrow = 2, byrow = TRUE),
type = "exposure",
bias_parms = c(.78, .78, .99, .99))

misclass(matrix(c(4558, 3428, 46305, 46085),
dimnames = list(c("AMI death+", "AMI death-"),
c("Male+", "Male-")),
nrow = 2, byrow = TRUE),
type = "outcome",
bias_parms = c(.53, .53, .99, .99))

# The following example comes from Chu et al. Sensitivity analysis of
# misclassification: A graphical and a Bayesian approach.
# Annals of Epidemiology 2006;16:834-841.
```

```
misclass(matrix(c(126, 92, 71, 224),
dimnames = list(c("Case", "Control"), c("Smoker +", "Smoker -")),
nrow = 2, byrow = TRUE),
type = "exposure",
bias_parms = c(.94, .94, .97, .97))

# The next example, using PPV/NPV, comes from Bodnar et al. Validity of birth
# certificate-derived maternal weight data.
# Paediatric and Perinatal Epidemiology 2014;28:203-212.
misclass(matrix(c(599, 4978, 31175, 391851),
dimnames = list(c("Preterm", "Term"), c("Underweight", "Normal weight")),
nrow = 2, byrow = TRUE),
type = "exposure_pv",
bias_parms = c(0.65, 0.74, 1, 0.98))
#
# The data for this example come from:
# Greenland S., Salvan A., Wegman D.H., Hallock M.F., Smith T.J.
# A case-control study of cancer mortality at a transformer-assembly facility.
# Int Arch Occup Environ Health 1994; 66(1):49-54.
greenland <- matrix(c(45, 94, 257, 945), dimnames = list(c("BC+", "BC-"),
c("Smoke+", "Smoke-")), nrow = 2, byrow = TRUE)
set.seed(123)
# Exposure misclassification, non-differential
probsens(greenland, type = "exposure", reps = 20000,
seca = list("trapezoidal", c(.75, .85, .95, 1)),
spca = list("trapezoidal", c(.75, .85, .95, 1)))

# Exposure misclassification, differential
probsens(greenland, type = "exposure", reps = 20000,
seca = list("trapezoidal", c(.75, .85, .95, 1)),
seexp = list("trapezoidal", c(.7, .8, .9, .95)),
spca = list("trapezoidal", c(.75, .85, .95, 1)),
spexp = list("trapezoidal", c(.7, .8, .9, .95)),
corr_se = .8,
corr_sp = .8)

probsens(greenland, type = "exposure", reps = 20000,
seca = list("beta", c(908, 16)),
seexp = list("beta", c(156, 56)),
spca = list("beta", c(153, 6)),
spexp = list("beta", c(205, 18)),
corr_se = .8,
corr_sp = .8)

probsens(matrix(c(338, 490, 17984, 32024),
dimnames = list(c("BC+", "BC-"), c("Smoke+", "Smoke-")), nrow = 2, byrow = TRUE),
type = "exposure",
reps = 1000,
seca = list("trapezoidal", c(.8, .9, .9, 1)),
spca = list("trapezoidal", c(.8, .9, .9, 1)))

# Disease misclassification
probsens(matrix(c(173, 602, 134, 663),
```

```
dimnames = list(c("BC+", "BC-"), c("Smoke+", "Smoke-")), nrow = 2, byrow = TRUE),
type = "outcome",
reps = 20000,
seca = list("uniform", c(.8, 1)),
spca = list("uniform", c(.8, 1)))

probsens(matrix(c(338, 490, 17984, 32024),
dimnames = list(c("BC+", "BC-"), c("Smoke+", "Smoke-")), nrow = 2, byrow = TRUE),
type = "outcome",
reps = 20000,
seca = list("uniform", c(.2, .6)),
seexp = list("uniform", c(.1, .5)),
spca = list("uniform", c(.99, 1)),
spexp = list("uniform", c(.99, 1)),
corr_se = .8,
corr_sp = .8)

probsens(matrix(c(173, 602, 134, 663),
dimnames = list(c("BC+", "BC-"), c("Smoke+", "Smoke-")), nrow = 2, byrow = TRUE),
type = "outcome",
reps = 20000,
seca = list("beta", c(100, 5)),
seexp = list("beta", c(110, 10)),
spca = list("beta", c(120, 15)),
spexp = list("beta", c(130, 30)),
corr_se = .8,
corr_sp = .8)

# Fox M.P., MacLehose R.F., Lash T.L.
# SAS and R code for probabilistic quantitative bias analysis for
# misclassified binary variables and binary unmeasured confounders
# Int J Epidemiol 2023:1624-1633.
## Not run:
fox <- matrix(c(40, 20, 60, 80),
dimnames = list(c("Diseased", "Non-diseased"), c("Exposed", "Unexposed")),
nrow = 2, byrow = TRUE)
set.seed(1234)
probsens(fox, type = "exposure", reps = 10^6,
seca = list("beta", c(25, 3)),
spca = list("trapezoidal", c(.9, .93, .97, 1)),
seexp = list("beta", c(47, 7)),
spexp = list("trapezoidal", c(.8, .83, .87, .9)),
corr_se = .8,
corr_sp = .8)

## End(Not run)

# Using PPV/NPV, from Bodnar et al. Validity of birth certificate-derived maternal
# weight data. Paediatric and Perinatal Epidemiology 2014;28:203-212.
set.seed(1234)
probsens(matrix(c(599, 4978, 31175, 391851),
dimnames = list(c("Preterm", "Term"), c("Underweight", "Normal weight")),
nrow = 2, byrow = TRUE),
```

```
type = "exposure_pv", reps = 10^6,
seca = list("beta", c(50, 27)),  ## PPV_case
spca = list("beta", c(120, .5)),  ## NPV_case
seexp = list("beta", c(132, 47)),  ## PPV_ctrl
spexp = list("beta", c(115, 2)))  ## NPV_ctrl
```

`misclass_cov` *Covariate misclassification*

Description

`misclass_cov()` allows to provide adjusted measures of association corrected for misclassification of a covariate (a potential confounder or effect measure modifier).

Usage

```
misclass_cov(case, exposed, covariate, bias_parms = NULL, alpha = 0.05)
```

Arguments

`case`	Outcome variable. If a variable, this variable is tabulated against.
`exposed`	Exposure variable.
`covariate`	Covariate to stratify on.
`bias_parms`	Vector defining the bias parameters. This vector has 4 elements between 0 and 1, in the following order: 1. Sensitivity of confounder classification among those with the outcome, 2. Sensitivity of confounder classification among those without the outcome, 3. Specificity of confounder classification among those with the outcome,and 4. Specificity of confounder classification among those without the outcome.
`alpha`	Significance level.

Value

A list with elements (for `misclass_cov()`):

`obs_data`	The analyzed stratified 2 x 2 tables from the observed data.
`corr_data`	The expected stratified observed data given the true data assuming misclassification.
`obs_measures`	A table of observed relative risk and odds ratio with confidence intervals.
`adj_measures`	A table of adjusted relative risk and odds ratio.
`bias_parms`	Input bias parameters.

References

Fox, M.P, MacLehose, R.F., Lash, T.L., 2021 *Applying Quantitative Bias Analysis to Epidemiologic Data*, pp.176–179, Springer.

See Also

Other misclassification: misclass(), probsens.irr()

Examples

```
# The data for this example come from:
# Berry, R.J., Kihlberg, R., and Devine, O. Impact of misclassification of in vitro
# fertilisation in studies of folic acid and twinning: modelling using population
# based Swedish vital records.
# BMJ, doi:10.1136/bmj.38369.437789.82 (published 17 March 2004)
misclass_cov(array(c(1319, 38054, 5641, 405546, 565, 3583, 781, 21958,
754, 34471, 4860, 383588),
dimnames = list(c("Twins+", "Twins-"),
c("Folic acid+", "Folic acid-"), c("Total", "IVF+", "IVF-")),
dim = c(2, 2, 3)),
bias_parms = c(.6, .6, .95, .95))
```

---

multidimBias *Multidimensional sensitivity analysis for different sources of bias*

Description

Multidimensional sensitivity analysis for different sources of bias, where the bias analysis is repeated within a range of values for the bias parameter(s).

Usage

```
multidimBias(
  case,
  exposed,
  type = c("exposure", "outcome", "confounder", "selection"),
  se = NULL,
  sp = NULL,
  bias_parms = NULL,
  OR_sel = NULL,
  alpha = 0.05,
  dec = 4,
  print = TRUE
)
```

Arguments

case	Outcome variable. If a variable, this variable is tabulated against.
exposed	Exposure variable.
type	Implement analysis for exposure misclassification, outcome misclassification, unmeasured confounder, or selection bias.

se	Numeric vector of sensitivities. Parameter used with exposure or outcome misclassification.
sp	Numeric vector of specificities. Parameter used with exposure or outcome misclassification. Should be the same length as se.
bias_parms	List of bias parameters used with unmeasured confounder. The list is made of 3 vectors of the same length: 1. Prevalence of Confounder in Exposure+ population, 2. Prevalence of Confounder in Exposure- population, and 3. Relative risk between Confounder and Outcome.
OR_sel	Selection odds ratios, for selection bias implementation.
alpha	Significance level.
dec	Number of decimals in the printout.
print	A logical scalar. Should the results be printed?

Value

A list with elements:

obs_data	The analyzed 2 x 2 table from the observed data.
obs_measures	A table of odds ratios and relative risk with confidence intervals.
adj_measures	Multidimensional corrected relative risk and/or odds ratio data.
bias_parms	Bias parameters.

Examples

```
multidimBias(matrix(c(45, 94, 257, 945),
dimnames = list(c("HIV+", "HIV-"), c("Circ+", "Circ-")),
nrow = 2, byrow = TRUE),
type = "exposure",
se = c(1, 1, 1, .9, .9, .9, .8, .8, .8),
sp = c(1, .9, .8, 1, .9, .8, 1, .9, .8))

multidimBias(matrix(c(45, 94, 257, 945),
dimnames = list(c("HIV+", "HIV-"), c("Circ+", "Circ-")),
nrow = 2, byrow = TRUE),
type = "outcome",
se = c(1, 1, 1, .9, .9, .9, .8, .8, .8),
sp = c(1, .9, .8, 1, .9, .8, 1, .9, .8))

multidimBias(matrix(c(105, 85, 527, 93),
dimnames = list(c("HIV+", "HIV-"), c("Circ+", "Circ-")),
nrow = 2, byrow = TRUE),
type = "confounder",
bias_parms = list(seq(.72, .92, by = .02),
seq(.01, .11, by = .01), seq(.13, 1.13, by = .1)))

multidimBias(matrix(c(136, 107, 297, 165),
dimnames = list(c("Uveal Melanoma+", "Uveal Melanoma-"),
```

```
c("Mobile Use+", "Mobile Use -")),
nrow = 2, byrow = TRUE),
type = "selection",
OR_sel = seq(1.5, 6.5, by = .5))
```

plot.episensr.booted *Plot of bootstrap simulation output for selection and misclassification bias*

Description

This takes an episensr bootstrap object and produces the plot of bootstrap replicates for selection or misclassification bias of the variable of interest, either relative risk or odds ratio. It also draws the confidence interval.

Usage

```
## S3 method for class 'episensr.booted'
plot(x, association = c("rr", "or"), ...)
```

Arguments

`x`	An object of class "episensr.booted" returned from the episensr bootstrap generation function.
`association`	Choice between bias adjusted relative risk (rr) and odds ratio (or).
`...`	Other unused arguments.

See Also

`boot.bias`, `selection`, `misclass`

Other visualization: `plot.episensr.probsens()`, `plot.mbias()`

Examples

```
misclass_eval <- misclass(matrix(c(215, 1449, 668, 4296),
dimnames = list(c("Breast cancer+", "Breast cancer-"),
c("Smoker+", "Smoker-")),
nrow = 2, byrow = TRUE),
type = "exposure",
bias_parms = c(.78, .78, .99, .99))

set.seed(123)
misclass_boot <- boot.bias(misclass_eval)
plot(misclass_boot, association = "rr")
```

plot.episensr.probsens

Plot(s) of probabilistic bias analyses

Description

This takes a probsens-family object and produces the distribution plot of chosen bias parameters, as well as distribution of adjusted measures (with confidence interval). It can also produce a forest plot of relative risks or odds ratios (with probsens(), probsens_conf(), or probsens.sel())

Usage

```
## S3 method for class 'episensr.probsens'
plot(
  x,
 parms = c("rr", "or", "rr_tot", "or_tot", "forest_rr", "forest_or", "irr", "irr_tot",
    "seca", "seexp", "spca", "spexp", "prev_exp", "prev_nexp", "risk"),
  ...
)
```

Arguments

x	An object of class "episensr.probsens" returned from the episensr probsens, probsens.sel, probsens_conf, probsens.irr, probsens.irr.conf functions.
parms	Choice between adjusted relative risk (rr) and odds ratio (or), total error relative risk and odds ratio (rr_tot and or_tot), forest plots (forest_rr and forest_or), seca, seexp, spca, and spexp, prev.exp, prev.nexp and risk, irr and irr_tot.
...	Other unused arguments.

See Also

probsens, probsens.sel, probsens_conf, probsens.irr, probsens.irr.conf

Other visualization: plot.episensr.booted(), plot.mbias()

Examples

```
set.seed(123)
risk <- probsens(matrix(c(45, 94, 257, 945),
dimnames = list(c("BC+", "BC-"), c("Smoke+", "Smoke-")), nrow = 2, byrow = TRUE),
type = "exposure", reps = 20000,
seca = list("trapezoidal", c(.75, .85, .95, 1)),
spca = list("trapezoidal", c(.75, .85, .95, 1)))
plot(risk, "rr")

set.seed(123)
odds <- probsens(matrix(c(45, 94, 257, 945),
```

```
dimnames = list(c("BC+", "BC-"), c("Smoke+", "Smoke-")), nrow = 2, byrow = TRUE),
type = "exposure", reps = 20000,
seca = list("beta", c(908, 16)),
seexp = list("beta", c(156, 56)),
spca = list("beta", c(153, 6)),
spexp = list("beta", c(205, 18)),
corr_se = .8,
corr_sp = .8)
plot(odds, "seca")

set.seed(123)
smoke <- probsens(matrix(c(215, 1449, 668, 4296),
dimnames = list(c("BC+", "BC-"), c("Smoke+", "Smoke-")), nrow = 2, byrow = TRUE),
type = "exposure", reps = 20000,
seca = list("uniform", c(.7, .95)),
spca = list("uniform", c(.9, .99)))
plot(smoke, "forest_or")

set.seed(123)
conf <- probsens_conf(matrix(c(105, 85, 527, 93),
dimnames = list(c("HIV+", "HIV-"), c("Circ+", "Circ-")), nrow = 2, byrow = TRUE),
reps = 20000,
prev_exp = list("triangular", c(.7, .9, .8)),
prev_nexp = list("trapezoidal", c(.03, .04, .05, .06)),
risk = list("triangular", c(.6, .7, .63)),
corr_p = .8)
plot(conf, "prev_exp")

set.seed(123)
inc1 <- probsens.irr(matrix(c(2, 67232, 58, 10539000),
dimnames = list(c("GBS+", "Person-time"), c("HPV+", "HPV-")), ncol = 2),
reps = 20000,
seca = list("trapezoidal", c(.4, .45, .55, .6)),
spca = list("constant", 1))
plot(inc1, "irr")

set.seed(123)
inc2 <- probsens.irr.conf(matrix(c(77, 10000, 87, 10000),
dimnames = list(c("D+", "Person-time"), c("E+", "E-")), ncol = 2),
reps = 20000,
prev_exp = list("trapezoidal", c(.01, .2, .3, .51)),
prev_nexp = list("trapezoidal", c(.09, .27, .35, .59)),
risk = list("trapezoidal", c(2, 2.5, 3.5, 4.5)),
corr_p = .8)
plot(inc2, "risk")
```

`plot.mbias` *Plot DAGs before and after conditioning on collider (M bias)*

Description

Create two Directed Acyclic Graphs (DAGs), before and after conditioning on the collider M, for selection bias caused by M bias, using 'ggdag'.

Usage

```
## S3 method for class 'mbias'
plot(x, type = c("before", "after"), dec = 2, ...)
```

Arguments

`x`	'mbias' object to plot.
`type`	DAG before or after conditioning on M.
`dec`	Number of digits displayed.
`...`	Other unused arguments.

Value

A DAG for selection bias caused by M bias.

See Also

`mbias`

Other visualization: `plot.episensr.booted()`, `plot.episensr.probsens()`

Examples

```
plot(mbias(or = c(2, 5.4, 2.5, 1.5, 1),
var = c("HIV", "Circumcision", "Muslim", "Low CD4", "Participation")))
```

`print.episensr` *Print associations for episensr class*

Description

Print associations for episensr objects.

Usage

```
## S3 method for class 'episensr'
print(x, digits = getOption("digits"), ...)
```

Arguments

`x`	An object of class 'episensr'.
`digits`	Minimal number of *significant* digits, see 'print.default'.
`...`	Other unused arguments.

Value

Print the observed and adjusted measures of association.

---

print.episensr.booted — *Print bootstrapped confidence intervals*

Description

Print bootstrap-ed confidence intervals for selection and misclassification bias functions.

Usage

```
## S3 method for class 'episensr.booted'
print(x, digits = getOption("digits"), ...)
```

Arguments

`x`	An object of class 'episensr.booted'.
`digits`	Minimal number of *significant* digits, see 'print.default'.
`...`	Other unused arguments.

Value

Print the confidence interval of the adjusted measures of association.

---

print.mbias — *Print association corrected for M bias*

Description

Print association corrected for M bias.

Usage

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

Arguments

`x`	An object of class 'mbias'.
`...`	Other unused arguments.

Value

Print the observed and adjusted measures of association.

probsens.conf_legacy *Legacy version of* `probsens.conf()`.

Description

[Superseded]

episensr 2.0.0 introduced breaking changes in probabilistic bias analyses by (1) using the NORTA transformation to define a correlation between distributions, and (2) sampling true prevalences and then sampling the adjusted cell counts rather than just using the expected cell counts from a simple quantitative bias analysis. This updated version should be preferred and this legacy version will be deprecated in future versions. However, if you need to quickly roll back to the previous calculations, this function provides the previous interface. To make old code work as is, add the following code to the top of your script:

```
library(episensr)
probsens.conf <- probsens.conf_legacy
```

Usage

```
probsens.conf_legacy(
  case,
  exposed,
  reps = 1000,
  prev.exp = list(dist = c("constant", "uniform", "triangular", "trapezoidal",
    "logit-logistic", "logit-normal", "beta"), parms = NULL),
  prev.nexp = list(dist = c("constant", "uniform", "triangular", "trapezoidal",
    "logit-logistic", "logit-normal", "beta"), parms = NULL),
  risk = list(dist = c("constant", "uniform", "triangular", "trapezoidal",
    "log-logistic", "log-normal"), parms = NULL),
  corr.p = NULL,
  discard = TRUE,
  alpha = 0.05
)
```

Arguments

`case`	Outcome variable. If a variable, this variable is tabulated against.
`exposed`	Exposure variable.
`reps`	Number of replications to run.
`prev.exp`	List defining the prevalence of exposure among the exposed. The first argument provides the probability distribution function (constant, uniform, triangular, trapezoidal, logit-logistic, logit-normal, or beta) and the second its parameters as a vector. Logit-logistic and logit-normal distributions can be shifted by providing lower and upper bounds. Avoid providing these values if a non-shifted distribution is desired.

1. constant: constant value,
2. uniform: min, max,
3. triangular: lower limit, upper limit, mode,
4. trapezoidal: min, lower mode, upper mode, max.
5. logit-logistic: location, scale, lower bound shift, upper bound shift,
6. logit-normal: location, scale, lower bound shift, upper bound shift.
7. beta: alpha, beta.

`prev.nexp` List defining the prevalence of exposure among the unexposed.

`risk` List defining the confounder-disease relative risk or the confounder-exposure odds ratio. The first argument provides the probability distribution function (constant, uniform, triangular, trapezoidal, log-logistic, or log-normal) and the second its parameters as a vector:

1. constant: constant value,
2. uniform: min, max,
3. triangular: lower limit, upper limit, mode,
4. trapezoidal: min, lower mode, upper mode, max.
5. log-logistic: shape, rate. Must be strictly positive,
6. log-normal: meanlog, sdlog. This is the mean and standard deviation on the log scale.

`corr.p` Correlation between the exposure-specific confounder prevalences.

`discard` A logical scalar. In case of negative adjusted count, should the draws be discarded? If set to FALSE, negative counts are set to zero.

`alpha` Significance level.

Value

A list with elements:

`obs.data` The analyzed 2 x 2 table from the observed data.

`obs.measures` A table of observed relative risk and odds ratio with confidence intervals.

`adj.measures` A table of corrected relative risks and odds ratios.

`sim.df` Data frame of random parameters and computed values.

`reps` Number of replications.

References

Lash, T.L., Fox, M.P, Fink, A.K., 2009 *Applying Quantitative Bias Analysis to Epidemiologic Data*, pp.117–150, Springer.

Examples

```
# The data for this example come from:
# Tyndall M.W., Ronald A.R., Agoki E., Malisa W., Bwayo J.J., Ndinya-Achola J.O. et al.
# Increased risk of infection with human immunodeficiency virus type 1 among
# uncircumcised men presenting with genital ulcer disease in Kenya.
```

```
# Clin Infect Dis 1996;23:449-53.
## Not run:
set.seed(123)
probsens.conf(matrix(c(105, 85, 527, 93),
dimnames = list(c("HIV+", "HIV-"), c("Circ+", "Circ-")), nrow = 2, byrow = TRUE),
reps = 20000,
prev.exp = list("triangular", c(.7, .9, .8)),
prev.nexp = list("trapezoidal", c(.03, .04, .05, .06)),
risk = list("triangular", c(.6, .7, .63)),
corr.p = .8)

set.seed(123)
probsens.conf(matrix(c(105, 85, 527, 93),
dimnames = list(c("HIV+", "HIV-"), c("Circ+", "Circ-")), nrow = 2, byrow = TRUE),
reps = 20000,
prev.exp = list("beta", c(200, 56)),
prev.nexp = list("beta", c(10, 16)),
risk = list("triangular", c(.6, .7, .63)),
corr.p = .8)

## End(Not run)
```

`probsens.irr` *Probabilistic sensitivity analysis for exposure misclassification of person-time data and random error.*

Description

Probabilistic sensitivity analysis to correct for exposure misclassification when person-time data has been collected. Non-differential misclassification is assumed when only the two bias parameters seca and spca are provided. Adding the 2 parameters seexp and spexp (i.e. providing the 4 bias parameters) evaluates a differential misclassification.

Usage

```
probsens.irr(
  counts,
  pt = NULL,
  reps = 1000,
  seca = list(dist = c("constant", "uniform", "triangular", "trapezoidal", "normal",
    "beta"), parms = NULL),
  seexp = NULL,
  spca = list(dist = c("constant", "uniform", "triangular", "trapezoidal", "normal",
    "beta"), parms = NULL),
  spexp = NULL,
  corr_se = NULL,
  corr_sp = NULL,
  alpha = 0.05
)
```

Arguments

`counts` A table or matrix where first row contains disease counts and second row contains person-time at risk, and first and second columns are exposed and unexposed observations, as:

	Exposed	Unexposed
Cases	a	b
Person-time	N1	N0

`pt` A numeric vector of person-time at risk. If provided, `counts` must be a numeric vector of disease counts.

`reps` Number of replications to run.

`seca` List defining the sensitivity of exposure classification among those with the outcome. The first argument provides the probability distribution function (uniform, triangular, trapezoidal, truncated normal, or beta) and the second its parameters as a vector. Lower and upper bounds of the truncated normal have to be between 0 and 1.

1. constant: constant value,
2. uniform: min, max,
3. triangular: lower limit, upper limit, mode,
4. trapezoidal: min, lower mode, upper mode, max,
5. normal: lower bound, upper bound, mean, sd,
6. beta: alpha, beta.

`seexp` List defining the sensitivity of exposure classification among those without the outcome.

`spca` List defining the specificity of exposure classification among those with the outcome.

`spexp` List defining the specificity of exposure classification among those without the outcome.

`corr_se` Correlation between case and non-case sensitivities.

`corr_sp` Correlation between case and non-case specificities.

`alpha` Significance level.

Details

Correlations between sensitivity (or specificity) of exposure classification among cases and controls can be specified and use the NORmal To Anything (NORTA) transformation (Li & Hammond, 1975).

Value

A list with elements:

`obs_data` The analyzed 2 x 2 table from the observed data.

`obs_measures` A table of observed incidence rate ratio with exact confidence interval.

`adj_measures` A table of corrected incidence rate ratios.

`sim_df` Data frame of random parameters and computed values.

Updated calculations

episensr 2.0.0 introduced updated calculations of probabilistic bias analyses by (1) using the NORTA transformation to define a correlation between distributions, and (2) sampling true prevalences and then sampling the adjusted cell counts rather than just using the expected cell counts from a simple quantitative bias analysis. This updated version should be preferred but if you need to run an old analysis, you can easily revert to the computation using `probsens.irr_legacy()` as follows:

```
library(episensr)
probsens.irr <- probsens.irr_legacy
```

References

Li, S.T., Hammond, J.L., 1975. *Generation of Pseudorandom Numbers with Specified Univariate Distributions and Correlation Coefficients*. IEEE Trans Syst Man Cybern 5:557-561.

See Also

Other misclassification: `misclass()`, `misclass_cov()`

Examples

```
set.seed(123)
# Exposure misclassification, non-differential
probsens.irr(matrix(c(2, 67232, 58, 10539000),
dimnames = list(c("GBS+", "Person-time"), c("HPV+", "HPV-")), ncol = 2),
reps = 20000,
seca = list("trapezoidal", c(.4, .45, .55, .6)),
spca = list("constant", 1))
```

---

probsens.irr.conf — *Probabilistic sensitivity analysis for unmeasured confounding of person-time data and random error.*

Description

Probabilistic sensitivity analysis to correct for unmeasured confounding when person-time data has been collected.

Usage

```
probsens.irr.conf(
  counts,
  pt = NULL,
  reps = 1000,
 prev_exp = list(dist = c("constant", "uniform", "triangular", "trapezoidal", "normal",
    "beta"), parms = NULL),
 prev_nexp = list(dist = c("constant", "uniform", "triangular", "trapezoidal", "normal",
```

```
    "beta"), parms = NULL),
  risk = list(dist = c("constant", "uniform", "triangular", "trapezoidal",
    "log-logistic", "log-normal"), parms = NULL),
  corr_p = NULL,
  alpha = 0.05
)
```

Arguments

`counts` A table or matrix where first row contains disease counts and second row contains person-time at risk, and first and second columns are exposed and unexposed observations, as:

	Exposed	Unexposed
Cases	a	b
Person-time	N1	N0

`pt` A numeric vector of person-time at risk. If provided, `counts` must be a numeric vector of disease counts.

`reps` Number of replications to run.

`prev_exp` List defining the prevalence of exposure among the exposed. The first argument provides the probability distribution function (constant, uniform, triangular, trapezoidal, truncated normal, or beta) and the second its parameters as a vector. Lower and upper bounds for truncated normal distribution cannot be les than zero.

1. constant; value,
2. uniform: min, max,
3. triangular: lower limit, upper limit, mode,
4. trapezoidal: min, lower mode, upper mode, max.
5. normal: lower bound, upper bound, mean, sd,
6. beta: alpha, beta.

`prev_nexp` List defining the prevalence of exposure among the unexposed.

`risk` List defining the confounder-disease relative risk or the confounder-exposure odds ratio. The first argument provides the probability distribution function (constant,uniform, triangular, trapezoidal, log-logistic, or log-normal) and the second its parameters as a vector:

1. constant: value,
2. uniform: min, max,
3. triangular: lower limit, upper limit, mode,
4. trapezoidal: min, lower mode, upper mode, max.
5. log-logistic: shape, rate. Must be strictly positive,
6. log-normal: meanlog, sdlog. This is the mean and standard deviation on the log scale.

`corr_p` Correlation between the exposure-specific confounder prevalences.

`alpha` Significance level.

Details

Correlations between prevalences of exposure classification among cases and controls can be specified and use the NORmal To Anything (NORTA) transformation (Li & Hammond, 1975).

Value

A list with elements:

`obs_data`	The analyzed 2 x 2 table from the observed data.
`obs_measures`	A table of observed incidence rate ratio with exact confidence interval.
`adj_measures`	A table of corrected incidence rate ratios.
`sim_df`	Data frame of random parameters and computed values.

Updated calculations

episensr 2.0.0 introduced updated calculations of probabilistic bias analyses by (1) using the NORTA transformation to define a correlation between distributions, and (2) sampling true prevalences and then sampling the adjusted cell counts rather than just using the expected cell counts from a simple quantitative bias analysis. This updated version should be preferred but if you need to run an old analysis, you can easily revert to the computation using `probsens.irr.conf_legacy()` as follows:

```
library(episensr)
probsens.irr.conf <- probsens.irr.conf_legacy
```

References

Li, S.T., Hammond, J.L., 1975. *Generation of Pseudorandom Numbers with Specified Univariate Distributions and Correlation Coefficients*. IEEE Trans Syst Man Cybern 5:557-561.

See Also

Other confounding: `confounders()`, `confounders.array()`, `confounders.evalue()`, `confounders.ext()`, `confounders.limit()`

Examples

```
set.seed(123)
# Unmeasured confounding
probsens.irr.conf(matrix(c(77, 10000, 87, 10000),
dimnames = list(c("D+", "Person-time"), c("E+", "E-")), ncol = 2),
reps = 20000,
prev_exp = list("trapezoidal", c(.01, .2, .3, .51)),
prev_nexp = list("trapezoidal", c(.09, .27, .35, .59)),
risk = list("trapezoidal", c(2, 2.5, 3.5, 4.5)),
corr_p = .8)
```

`probsens.irr.conf_legacy` *Legacy version of* `probsens.irr.conf()`.

Description

[Superseded]

episensr 2.0.0 introduced breaking changes in probabilistic bias analyses by (1) using the NORTA transformation to define a correlation between distributions, and (2) sampling true prevalences and then sampling the adjusted cell counts rather than just using the expected cell counts from a simple quantitative bias analysis. This updated version should be preferred and this legacy version will be deprecated in future versions. However, if you need to quickly roll back to the previous calculations, this function provides the previous interface. To make old code work as is, add the following code to the top of your script:

```
library(episensr)
probsens.irr.conf <- probsens.irr.conf_legacy
```

Usage

```
probsens.irr.conf_legacy(
  counts,
  pt = NULL,
  reps = 1000,
  prev.exp = list(dist = c("constant", "uniform", "triangular", "trapezoidal",
    "logit-logistic", "logit-normal", "beta"), parms = NULL),
  prev.nexp = list(dist = c("constant", "uniform", "triangular", "trapezoidal",
    "logit-logistic", "logit-normal", "beta"), parms = NULL),
  risk = list(dist = c("constant", "uniform", "triangular", "trapezoidal",
    "log-logistic", "log-normal"), parms = NULL),
  corr.p = NULL,
  discard = TRUE,
  alpha = 0.05
)
```

Arguments

`counts` A table or matrix where first row contains disease counts and second row contains person-time at risk, and first and second columns are exposed and unexposed observations, as:

	Exposed	Unexposed
Cases	a	b
Person-time	N1	N0

`pt` A numeric vector of person-time at risk. If provided, `counts` must be a numeric vector of disease counts.

`reps` Number of replications to run.

`prev.exp` List defining the prevalence of exposure among the exposed. The first argument provides the probability distribution function (constant,uniform, triangular, trapezoidal, logit-logistic, logit-normal, or beta) and the second its parameters as a vector. Logit-logistic and logit-normal distributions can be shifted by providing lower and upper bounds. Avoid providing these values if a non-shifted distribution is desired.

1. constant; value,
2. uniform: min, max,
3. triangular: lower limit, upper limit, mode,
4. trapezoidal: min, lower mode, upper mode, max.
5. logit-logistic: location, scale, lower bound shift, upper bound shift,
6. logit-normal: location, scale, lower bound shift, upper bound shift,
7. beta: alpha, beta.

`prev.nexp` List defining the prevalence of exposure among the unexposed.

`risk` List defining the confounder-disease relative risk or the confounder-exposure odds ratio. The first argument provides the probability distribution function (constant,uniform, triangular, trapezoidal, log-logistic, or log-normal) and the second its parameters as a vector:

1. constant: value,
2. uniform: min, max,
3. triangular: lower limit, upper limit, mode,
4. trapezoidal: min, lower mode, upper mode, max.
5. log-logistic: shape, rate. Must be strictly positive,
6. log-normal: meanlog, sdlog. This is the mean and standard deviation on the log scale.

`corr.p` Correlation between the exposure-specific confounder prevalences.

`discard` A logical scalar. In case of negative adjusted count, should the draws be discarded? If set to FALSE, negative counts are set to zero.

`alpha` Significance level.

Value

A list with elements:

`obs.data` The analyzed 2 x 2 table from the observed data.

`obs.measures` A table of observed incidence rate ratio with exact confidence interval.

`adj.measures` A table of corrected incidence rate ratios.

`sim.df` Data frame of random parameters and computed values.

References

Lash, T.L., Fox, M.P, Fink, A.K., 2009 *Applying Quantitative Bias Analysis to Epidemiologic Data*, pp.117–150, Springer.

Examples

```
## Not run:
set.seed(123)
# Unmeasured confounding
probsens.irr.conf(matrix(c(77, 10000, 87, 10000),
dimnames = list(c("D+", "Person-time"), c("E+", "E-")), ncol = 2),
reps = 20000,
prev.exp = list("trapezoidal", c(.01, .2, .3, .51)),
prev.nexp = list("trapezoidal", c(.09, .27, .35, .59)),
risk = list("trapezoidal", c(2, 2.5, 3.5, 4.5)),
corr.p = .8)

## End(Not run)
```

---

probsens.irr_legacy *Legacy version of* probsens.irr().

Description

[Superseded]

episensr 2.0.0 introduced breaking changes in probabilistic bias analyses by (1) using the NORTA transformation to define a correlation between distributions, and (2) sampling true prevalences and then sampling the adjusted cell counts rather than just using the expected cell counts from a simple quantitative bias analysis. This updated version should be preferred and this legacy version will be deprecated in future versions. However, if you need to quickly roll back to the previous calculations, this function provides the previous interface. To make old code work as is, add the following code to the top of your script:

```
library(episensr)
probsens.irr <- probsens.irr_legacy
```

Usage

```
probsens.irr_legacy(
  counts,
  pt = NULL,
  reps = 1000,
  seca.parms = list(dist = c("constant", "uniform", "triangular", "trapezoidal",
    "logit-logistic", "logit-normal", "beta"), parms = NULL),
  seexp.parms = NULL,
  spca.parms = list(dist = c("constant", "uniform", "triangular", "trapezoidal",
    "logit-logistic", "logit-normal", "beta"), parms = NULL),
  spexp.parms = NULL,
  corr.se = NULL,
  corr.sp = NULL,
  discard = TRUE,
  alpha = 0.05
)
```

Arguments

`counts` A table or matrix where first row contains disease counts and second row contains person-time at risk, and first and second columns are exposed and unexposed observations, as:

	Exposed	Unexposed
Cases	a	b
Person-time	N1	N0

`pt` A numeric vector of person-time at risk. If provided, `counts` must be a numeric vector of disease counts.

`reps` Number of replications to run.

`seca.parms` List defining the sensitivity of exposure classification among those with the outcome. The first argument provides the probability distribution function (uniform, triangular, trapezoidal, logit-logistic, logit-normal, or beta) and the second its parameters as a vector. Logit-logistic and logit-normal distributions can be shifted by providing lower and upper bounds. Avoid providing these values if a non-shifted distribution is desired.

1. constant: constant value,
2. uniform: min, max,
3. triangular: lower limit, upper limit, mode,
4. trapezoidal: min, lower mode, upper mode, max,
5. logit-logistic: location, scale, lower bound shift, upper bound shift,
6. logit-normal: location, scale, lower bound shift, upper bound shift,
7. beta: alpha, beta.

`seexp.parms` List defining the sensitivity of exposure classification among those without the outcome.

`spca.parms` List defining the specificity of exposure classification among those with the outcome.

`spexp.parms` List defining the specificity of exposure classification among those without the outcome.

`corr.se` Correlation between case and non-case sensitivities.

`corr.sp` Correlation between case and non-case specificities.

`discard` A logical scalar. In case of negative adjusted count, should the draws be discarded? If set to FALSE, negative counts are set to zero.

`alpha` Significance level.

Value

A list with elements:

`obs.data` The analyzed 2 x 2 table from the observed data.

`obs.measures` A table of observed incidence rate ratio with exact confidence interval.

`adj.measures` A table of corrected incidence rate ratios.

`sim.df` Data frame of random parameters and computed values.

References

Lash, T.L., Fox, M.P, Fink, A.K., 2009 *Applying Quantitative Bias Analysis to Epidemiologic Data*, pp.117–150, Springer.

Examples

```
## Not run:
set.seed(123)
# Exposure misclassification, non-differential
probsens.irr(matrix(c(2, 67232, 58, 10539000),
dimnames = list(c("GBS+", "Person-time"), c("HPV+", "HPV-")), ncol = 2),
reps = 20000,
seca.parms = list("trapezoidal", c(.4, .45, .55, .6)),
spca.parms = list("constant", 1))

## End(Not run)
```

`probsens_legacy` *Legacy version of* `probsens()`.

Description

[Superseded]

episensr 2.0.0 introduced breaking changes in probabilistic bias analyses by (1) using the NORTA transformation to define a correlation between distributions, and (2) sampling true prevalences and then sampling the adjusted cell counts rather than just using the expected cell counts from a simple quantitative bias analysis. This updated version should be preferred and this legacy version will be deprecated in future versions. However, if you need to quickly roll back to the previous calculations, this function provides the previous interface. To make old code work as is, add the following code to the top of your script:

```
library(episensr)
probsens <- probsens_legacy
```

Usage

```
probsens_legacy(
  case,
  exposed,
  type = c("exposure", "outcome"),
  reps = 1000,
  seca.parms = list(dist = c("constant", "uniform", "triangular", "trapezoidal",
    "logit-logistic", "logit-normal", "beta"), parms = NULL),
  seexp.parms = NULL,
  spca.parms = list(dist = c("constant", "uniform", "triangular", "trapezoidal",
    "logit-logistic", "logit-normal", "beta"), parms = NULL),
  spexp.parms = NULL,
```

```
  corr.se = NULL,
  corr.sp = NULL,
  discard = TRUE,
  alpha = 0.05
)
```

Arguments

`case` Outcome variable. If a variable, this variable is tabulated against.

`exposed` Exposure variable.

`type` Choice of correction for exposure or outcome misclassification.

`reps` Number of replications to run.

`seca.parms` List defining:

1. The sensitivity of exposure classification among those with the outcome (when `type = "exposure"`), or
2. The sensitivity of outcome classification among those with the exposure (when `type = "outcome"`).

The first argument provides the probability distribution function (constant, uniform, triangular, trapezoidal, logit-logistic, logit-normal, or beta) and the second its parameters as a vector. Logit-logistic and logit-normal distributions can be shifted by providing lower and upper bounds. Avoid providing these values if a non-shifted distribution is desired.

1. constant: constant value,
2. uniform: min, max,
3. triangular: lower limit, upper limit, mode,
4. trapezoidal: min, lower mode, upper mode, max,
5. logit-logistic: location, scale, lower bound shift, upper bound shift,
6. logit-normal: location, scale, lower bound shift, upper bound shift.
7. beta: alpha, beta.

`seexp.parms` List defining:

1. The sensitivity of exposure classification among those without the outcome (when `type = "exposure"`), or
2. The sensitivity of outcome classification among those without the exposure (when `type = "outcome"`).

`spca.parms` List as above for `seca.parms` but for specificity.

`spexp.parms` List as above for `seexp.parms` but for specificity.

`corr.se` Correlation between case and non-case sensitivities.

`corr.sp` Correlation between case and non-case specificities.

`discard` A logical scalar. In case of negative adjusted count, should the draws be discarded? If set to FALSE, negative counts are set to zero.

`alpha` Significance level.

Value

A list with elements:

obs.data	The analyzed 2 x 2 table from the observed data.
obs.measures	A table of observed relative risk and odds ratio with confidence intervals.
adj.measures	A table of corrected relative risks and odds ratios.
sim.df	Data frame of random parameters and computed values.
reps	Number of replications.

References

Lash, T.L., Fox, M.P, Fink, A.K., 2009 *Applying Quantitative Bias Analysis to Epidemiologic Data*, pp.117–150, Springer.

Examples

```
# The data for this example come from:
# Greenland S., Salvan A., Wegman D.H., Hallock M.F., Smith T.J.
# A case-control study of cancer mortality at a transformer-assembly facility.
# Int Arch Occup Environ Health 1994; 66(1):49-54.
## Not run:
set.seed(123)
# Exposure misclassification, non-differential
probsens_legacy(matrix(c(45, 94, 257, 945),
dimnames = list(c("BC+", "BC-"), c("Smoke+", "Smoke-")), nrow = 2, byrow = TRUE),
type = "exposure",
reps = 20000,
seca.parms = list("trapezoidal", c(.75, .85, .95, 1)),
spca.parms = list("trapezoidal", c(.75, .85, .95, 1)))

# Exposure misclassification, differential
probsens_legacy(matrix(c(45, 94, 257, 945),
dimnames = list(c("BC+", "BC-"), c("Smoke+", "Smoke-")), nrow = 2, byrow = TRUE),
type = "exposure",
reps = 20000,
seca.parms = list("trapezoidal", c(.75, .85, .95, 1)),
seexp.parms = list("trapezoidal", c(.7, .8, .9, .95)),
spca.parms = list("trapezoidal", c(.75, .85, .95, 1)),
spexp.parms = list("trapezoidal", c(.7, .8, .9, .95)),
corr.se = .8,
corr.sp = .8)

probsens_legacy(matrix(c(45, 94, 257, 945),
dimnames = list(c("BC+", "BC-"), c("Smoke+", "Smoke-")), nrow = 2, byrow = TRUE),
type = "exposure",
reps = 20000,
seca.parms = list("beta", c(908, 16)),
seexp.parms = list("beta", c(156, 56)),
spca.parms = list("beta", c(153, 6)),
spexp.parms = list("beta", c(205, 18)),
```

```
corr.se = .8,
corr.sp = .8)

probsens_legacy(matrix(c(338, 490, 17984, 32024),
dimnames = list(c("BC+", "BC-"), c("Smoke+", "Smoke-")), nrow = 2, byrow = TRUE),
type = "exposure",
reps = 1000,
seca.parms = list("trapezoidal", c(.8, .9, .9, 1)),
spca.parms = list("trapezoidal", c(.8, .9, .9, 1)))

# Disease misclassification
probsens_legacy(matrix(c(173, 602, 134, 663),
dimnames = list(c("BC+", "BC-"), c("Smoke+", "Smoke-")), nrow = 2, byrow = TRUE),
type = "outcome",
reps = 20000,
seca.parms = list("uniform", c(.8, 1)),
spca.parms = list("uniform", c(.8, 1)))

probsens_legacy(matrix(c(338, 490, 17984, 32024),
dimnames = list(c("BC+", "BC-"), c("Smoke+", "Smoke-")), nrow = 2, byrow = TRUE),
type = "outcome",
reps = 20000,
seca.parms = list("uniform", c(.2, .6)),
seexp.parms = list("uniform", c(.1, .5)),
spca.parms = list("uniform", c(.99, 1)),
spexp.parms = list("uniform", c(.99, 1)),
corr.se = .8,
corr.sp = .8)

probsens_legacy(matrix(c(173, 602, 134, 663),
dimnames = list(c("BC+", "BC-"), c("Smoke+", "Smoke-")), nrow = 2, byrow = TRUE),
type = "outcome",
reps = 20000,
seca.parms = list("beta", c(100, 5)),
seexp.parms = list("beta", c(110, 10)),
spca.parms = list("beta", c(120, 15)),
spexp.parms = list("beta", c(130, 30)),
corr.se = .8,
corr.sp = .8)

## End(Not run)
```

`selection` *Selection bias.*

Description

`selection()` and `probsens.sel()` allow to provide adjusted measures of association corrected for selection bias.

Usage

```
selection(case, exposed, bias_parms = NULL, alpha = 0.05)

probsens.sel(
  case,
  exposed,
  reps = 1000,
 case_exp = list(dist = c("constant", "uniform", "triangular", "trapezoidal", "normal",
    "beta"), parms = NULL),
 case_nexp = list(dist = c("constant", "uniform", "triangular", "trapezoidal", "normal",
    "beta"), parms = NULL),
 ncase_exp = list(dist = c("constant", "uniform", "triangular", "trapezoidal", "normal",
    "beta"), parms = NULL),
  ncase_nexp = list(dist = c("constant", "uniform", "triangular", "trapezoidal",
    "normal", "beta"), parms = NULL),
  alpha = 0.05
)
```

Arguments

`case` Outcome variable. If a variable, this variable is tabulated against.

`exposed` Exposure variable.

`bias_parms` Selection probabilities. Either a vector of 4 elements between 0 and 1 defining the following probabilities in this order can be provided:

1. Selection probability among cases exposed (1),
2. Selection probability among cases unexposed (2),
3. Selection probability among noncases exposed (3), and
4. Selection probability among noncases unexposed (4).

or a single positive selection-bias factor which is the ratio of the exposed versus unexposed selection probabilities comparing cases and noncases ((1*4)/(2*3) from above).

`alpha` Significance level.

`reps` Number of replications to run.

`case_exp` If or_parms not provided, defines the selection probability among case exposed. The first argument provides the probability distribution function and the second its parameters as a vector:

1. constant: constant value,
2. uniform: min, max,
3. triangular: lower limit, upper limit, mode,
4. trapezoidal: min, lower mode, upper mode, max.
5. normal: truncated normal with lower bound, upper bound, mean, sd,
6. beta: alpha, beta.

`case_nexp` Same among cases non-exposed.

`ncase_exp` Same among non-cases exposed.

`ncase_nexp` Same among non-cases non-exposed.

Value

A list with elements:

`model`	Bias analysis performed.
`obs_data`	The analyzed 2 x 2 table from the observed data.
`corr_data`	The same table corrected for selection proportions.
`obs_measures`	A table of odds ratios and relative risk with confidence intervals.
`adj_measures`	Selection bias corrected measures of outcome-exposure relationship.
`bias_parms`	Input bias parameters: selection probabilities.
`selbias_or`	Selection bias odds ratio based on the bias parameters chosen.

A list with elements (for `probsens.sel()`):

`obs_data`	The analyzed 2 x 2 table from the observed data.
`obs_measures`	A table of observed odds ratio with confidence intervals.
`adj_measures`	A table of corrected odds ratios.
`sim_df`	Data frame of random parameters and computed values.
`reps`	Number of replications.

Simple bias analysis with `selection()`

`selection()` allows you to run a simple sensitivity analysis to correct for selection bias using estimates of the selection proportions.

Probabilistic sensitivity analysis with `probsens.sel()`

`probsens.sel()` performs a summary-level probabilistic sensitivity analysis to correct for selection bias.

References

Fox, M.P, MacLehose, R.F., Lash, T.L., 2021 *Applying Quantitative Bias Analysis to Epidemiologic Data*, pp.90–91, 274–279, Springer.

See Also

Other selection: `mbias()`

Examples

```
# The data for this example come from:
# Stang A., Schmidt-Pokrzywniak A., Lehnert M., Parkin D.M., Ferlay J., Bornfeld N.
# et al.
# Population-based incidence estimates of uveal melanoma in Germany. Supplementing
# cancer registry data by case-control data.
# Eur J Cancer Prev 2006;15:165-70.
selection(matrix(c(136, 107, 297, 165),
dimnames = list(c("UM+", "UM-"), c("Mobile+", "Mobile-")),
```

```
nrow = 2, byrow = TRUE),
bias_parms = c(.94, .85, .64, .25))


selection(matrix(c(136, 107, 297, 165),
dimnames = list(c("UM+", "UM-"), c("Mobile+", "Mobile-")),
nrow = 2, byrow = TRUE),
bias_parms = 0.43)
#
# The data for this example come from:
# Stang A., Schmidt-Pokrzywniak A., Lehnert M., Parkin D.M., Ferlay J., Bornfeld N. et al.
# Population-based incidence estimates of uveal melanoma in Germany.
# Supplementing cancer registry data by case-control data.
# Eur J Cancer Prev 2006;15:165-70.
set.seed(1234)
probsens.sel(matrix(c(139, 114, 369, 377),
dimnames = list(c("Melanoma+", "Melanoma-"), c("Mobile+", "Mobile-")), nrow = 2, byrow = TRUE),
reps = 5000,
case_exp = list("beta", c(139, 5.1)),
case_nexp = list("beta", c(114, 11.9)),
ncase_exp = list("beta", c(369, 96.1)),
ncase_nexp = list("beta", c(377, 282.9)))
```

`%>%` *Pipe bias functions*

Description

episensr also uses the pipe function, `%>%` to turn function composition into a series of imperative statements.

Arguments

`lhs, rhs`	Data or bias function and a function to apply to it

Examples

```
# Instead of
misclass(matrix(c(118, 832, 103, 884),
dimnames = list(c("BC+", "BC-"), c("AD+", "AD-")), nrow = 2, byrow = TRUE),
type = "exposure", bias_parms = c(.56, .58, .99, .97))
# you can write
dat <- matrix(c(118, 832, 103, 884),
dimnames = list(c("BC+", "BC-"), c("AD+", "AD-")), nrow = 2, byrow = TRUE)
dat %>% misclass(., type = "exposure", bias_parms = c(.56, .58, .99, .97))
```

Index