Package: confoundvis (via r-universe)

November 23, 2026

Type Package

Title A Common Interface for Visualizing and Reporting Sensitivity Analyses for Unmeasured Confounding

Version 0.4.0

Description A common interface to established sensitivity analyses for unmeasured confounding in observational studies. Results of impact threshold and robustness of inference to replacement analysis (Frank, 2000, <doi:10.1177/0049124100029002001>; Frank et al., 2013, <doi:10.3102/0162373713493129>), partial R-squared omitted-variable bias analysis (Cinelli and Hazlett, 2020, <doi:10.1111/rssb.12348>), E-values (VanderWeele and Ding, 2017, <doi:10.7326/M16-2607>), and Rosenbaum bounds for matched pairs (Rosenbaum, 2002, <doi:10.1007/978-1-4757-3692-2>), the sensitivity of instrumental variable estimates (Cinelli and Hazlett, 2025, <doi:10.1093/biomet/asaf004>), and omitted-variable bias bounds for double machine learning (Chernozhukov et al., 2024, <doi:10.3386/w30302>), and the risk difference of a binary outcome under a binary unmeasured confounder (Lin, Psaty, and Kronmal, 1998, <doi:10.2307/2533848>) are stored in one object that keeps each framework's own parameter. Robustness curves, summary tables, and plain-language reports are available for every framework; two-parameter surfaces, the exact correlation contour, and observed-covariate benchmarks are provided where the corresponding framework defines them. Results can be computed from fitted models or matched-pair differences, or converted from the 'sensemakr', 'konfound', 'EValue', and 'iv.sensemakr' packages, and agree with those packages. For fits from the 'DoubleML' package, which has no sensitivity analysis in R, the bounds of the Python 'doubleml' package are computed.

License GPL-3

Encoding UTF-8

Language en-US

Depends R (>= 4.1.0)

Imports ggplot2 (>= 3.4.0), rlang, graphics, grDevices, stats

Suggests testthat (>= 3.0.0), knitr, rmarkdown, sensemakr, konfound, EValue, DOS2, sensitivity2x2xk, sensitivitymv, ggrepel, iv.sensemakr, DoubleML, mlr3, mlr3learners, data.table, lgr, marginaleffects

Config/testthat/edition 3

VignetteBuilder knitr

URL https://github.com/subirhait/confoundvis

BugReports https://github.com/subirhait/confoundvis/issues

NeedsCompilation no

Config/roxygen2/version 8.1.1

Author Subir Hait [aut, cre] (ORCID: <https://orcid.org/0009-0004-9871-9677>)

Maintainer Subir Hait <haitsubi@msu.edu>

Repository https://cran.r-universe.dev

Date/Publication 2026-11-23 07:50:38 UTC

RemoteUrl https://github.com/cran/confoundvis

RemoteRef HEAD

RemoteSha 07a75198faaaea857446fc3173986c0016d79564

Contents

as.data.frame.confoundsens . . . 3
as_confoundsens . . . 4
confoundvis . . . 5
covariate_impacts . . . 7
from_evalue . . . 9
from_iv_sensemakr . . . 9
from_konfound . . . 11
from_sensemakr . . . 12
itcv_lm . . . 13
new_confoundsens . . . 14
plot.confoundvis_profile . . . 15
plot.confoundvis_result . . . 16
plot_robustness_curve . . . 17
plot_sensitivity_contour . . . 18
plot_sensitivity_love . . . 19
plot_sensitivity_surface . . . 20
print.confoundsens . . . 21
rir_lm . . . 22
robustness_points . . . 23

sens_dml 24
sens_path_lm 26
sens_path_pairs 27
sens_profile 29
sens_report 30
sens_risk_difference 31
sens_surface 33
sens_table 35
summary.confoundsens 36

Index **37**

`as.data.frame.confoundsens` *Coerce a confoundsens object to a data frame*

Description

Coerce a confoundsens object to a data frame

Usage

```
## S3 method for class 'confoundsens'
as.data.frame(x, ...)
```

Arguments

`x`	A ‘confoundsens‘ object.
`...`	Unused; included for S3 method consistency.

Value

A data.frame with columns ‘lambda‘ and ‘theta‘, plus optional columns ‘level‘, ‘se‘, and ‘t‘ when present in ‘x‘.

Examples

```
x <- new_confoundsens(
  lambda = seq(0, 0.2, length.out = 5),
  theta  = seq(1, 0.8, length.out = 5),
  se     = rep(0.05, 5)
)
as.data.frame(x)
```

as_confoundsens *Convert a data frame to a confoundsens object*

Description

Converts a data frame containing sensitivity analysis results into a ‘confoundsens‘ object suitable for use with ‘confoundvis‘ plotting functions.

Usage

```
as_confoundsens(data, ...)

## Default S3 method:
as_confoundsens(data, ...)

## S3 method for class 'lm'
as_confoundsens(data, ...)

## S3 method for class 'sensemakr'
as_confoundsens(data, ...)

## S3 method for class 'data.frame'
as_confoundsens(
  data,
  lambda = "lambda",
  theta = "theta",
  level = NULL,
  se = NULL,
  t = NULL,
  ...
)
```

Arguments

data	A data.frame containing sensitivity analysis results, a fitted ‘lm‘ model, or a ‘sensemakr‘ object.
...	Passed to methods. For ‘lm‘ objects, ‘treatment‘ (required), ‘lambda‘, and ‘ratio‘; for ‘sensemakr‘ objects, ‘lambda‘ and ‘ratio‘.
lambda	Character string; name of the column containing lambda values. Default ‘"lambda"‘.
theta	Character string; name of the column containing theta values. Default ‘"theta"‘.
level	Optional character string; name of the column containing subgroup (stratum) labels (e.g., ‘"female"‘ / ‘"male"‘).
se	Optional character string; name of the column containing standard errors for ‘theta‘.
t	Optional character string; name of the column containing test statistics.

Details

'as_confoundsens()' is generic. The data-frame method (the 0.1.0 behaviour) maps columns to fields. Methods for fitted 'lm' models and 'sensemakr' objects compute the partial R^2 path via [sens_path_lm()] and [from_sensemakr()]; see also [from_konfound()] and [from_evalue()].

Value

A 'confoundsens' object.

Examples

```
df <- data.frame(
  lambda = seq(0, 0.2, length.out = 10),
  theta  = seq(1, 0.5, length.out = 10),
  se     = rep(0.1, 10),
  level  = rep(c("female", "male"), length.out = 10)
)
x <- as_confoundsens(df)
x

# Directly from a fitted model
fit <- lm(mpg ~ am + wt + hp, data = mtcars)
as_confoundsens(fit, treatment = "am")
```

---

confoundvis *Sensitivity analysis for unmeasured confounding in one call*

Description

The simplest way to use the package. Give it a fitted model and the name of the treatment, or a vector of matched-pair differences, and it runs every analysis the package supports for that input, compares the result with the measured covariates, and prints a short summary in plain language. The returned object can be printed, plotted, and passed to the other functions of the package.

Usage

```
confoundvis(
  x,
  treatment = NULL,
  alpha = 0.05,
  labels = NULL,
  extra = NULL,
  risk = NULL
)
```

Arguments

x	A fitted ‘lm‘ or ‘glm‘ object, a numeric vector of matched-pair differences, an ‘iv.sensemakr‘ ‘iv_fit‘ or ‘sensemakr‘ result, or a fitted ‘DoubleMLPLR‘ or ‘DoubleMLIRM‘ object.
treatment	Name of the treatment coefficient (for a model).
alpha	Significance level (two-sided for models, one-sided for matched pairs). For an ‘iv.sensemakr‘ ‘sensemakr()‘ result, the level of that object is used unless ‘alpha‘ is given, in which case the analysis is rerun at ‘alpha‘.
labels	Optional named character vector of readable covariate labels, passed to [covariate_impacts()].
extra	Optional named list of further ‘confoundsens‘ paths (for example from [from_evalue()]) to add for a linear model.
risk	For a binomial ‘glm‘: an optional list with the confounder scenario for [sens_risk_difference()], with elements ‘p1‘, ‘p0‘, and optionally ‘threshold‘ and any other argument of that function. The risk-difference analysis is then added to the RIR.

Details

What is run depends on the input:

- A linear model fitted by [stats::lm()]: the partial-R^2, ITCV, and RIR analyses ([sens_profile()]) and the measured-covariate benchmarks on both regression metrics ([covariate_impacts()]).
- A logistic regression or other [stats::glm()] fit: the RIR computed by 'konfound' (which must be installed) and imported with [from_konfound()]. konfound's RIR for such models is an approximation; see its documentation. For a binomial model with a binary treatment, give ‘risk‘ to add the risk-difference analysis of [sens_risk_difference()].
- A numeric vector: treated as treated-minus-control differences in matched pairs, analysed with Rosenbaum's sensitivity model ([sens_path_pairs()]; one-sided test).
- An instrumental-variable analysis from ‘iv.sensemakr‘ (an ‘iv_fit()‘ or ‘sensemakr()‘ result): the sensitivity of the Anderson–Rubin inference ([from_iv_sensemakr()]).
- A ‘DoubleMLPLR‘ or ‘DoubleMLIRM‘ fit from ‘DoubleML‘, fitted with ‘store_predictions = TRUE‘: the omitted-variable-bias analysis of Chernozhukov et al. ([sens_dml()]), with ‘rho = 1‘ and the one-sided level ‘1 - alpha / 2‘, so that its significance crossing matches a two-sided test at ‘alpha‘.

Each analysis keeps its own sensitivity parameter and decision rule. The numbers measure different things and must not be ranked against one another. A sensitivity analysis shows how strong an unmeasured confounder would need to be; it does not show whether one exists.

Value

An object of class ‘confoundvis_result‘: a list with ‘type‘ (‘"lm"‘, ‘"glm"‘, ‘"pairs"‘, ‘"iv"‘, or ‘"dml"‘), ‘treatment‘, ‘paths‘ (named list of ‘confoundsens‘ objects), ‘table‘ ([sens_table()] output), ‘benchmarks‘ (for ‘"lm"‘: the [covariate_impacts()] results for the two metrics), ‘profile‘ (for ‘"lm"‘), and ‘summary‘ (the plain-language lines).

See Also

[sens_profile()], [sens_report()], [covariate_impacts()]

Examples

```
fit <- lm(mpg ~ am + wt + hp, data = mtcars)
res <- confoundvis(fit, "wt")
res
plot(res)
plot(res, which = "benchmarks")

# matched pairs (Rosenbaum bounds)
d <- c(1.2, 0.8, 2.1, 0.4, 1.7, -0.3, 0.9, 1.5, 2.2, 0.6, 1.1, 0.7)
confoundvis(d)
```

---

covariate_impacts *Benchmark observed covariates against a sensitivity threshold*

Description

For each covariate term in a fitted linear model (other than the treatment and the intercept), computes how strongly it is related to the treatment and to the outcome. These are the observed benchmarks that analysts compare with a sensitivity threshold ("would an omitted variable as strong as this covariate overturn the result?").

Usage

```
covariate_impacts(
  model,
  treatment,
  metric = c("itcv", "partial_r2"),
  alpha = 0.05,
  labels = NULL
)
```

Arguments

`model`	A fitted ‘lm‘ object.
`treatment`	Character string naming the treatment coefficient.
`metric`	Which metric populates the ‘impact‘ column used by [plot_sensitivity_love()]: ‘"itcv"‘ (default) or ‘"partial_r2"‘.
`alpha`	Two-sided significance level used for the ITCV threshold.
`labels`	Optional named character vector of display labels, for example ‘c(surv2md1 = "Estimated 2-month survival")‘. Names are model term labels. The labels fill the ‘label‘ column, which the plotting functions show in place of the term names; terms without a label keep their name.

Details

Two metrics are returned:

'impact' (ITCV metric) $r_{DZ|W} \times r_{YZ|W}$, the product of partial correlations of the covariate with the treatment and with the outcome, each conditioning on the remaining covariates W (as in Frank, 2000). Compared with the ITCV. Only positive values (same sign as the estimate) push the estimate toward zero; the returned 'impact' is signed so that positive means "reduces the estimate". Defined only for single-column terms.

'bias_index' (partial R^2 metric) $\sqrt{R^2_{Y\sim Z|D,W} R^2_{D\sim Z|W}/(1 - R^2_{D\sim Z|W})}$. An omitted confounder with this pair of partial R^2 values would move the estimate by 'bias_index' $\times\, se\sqrt{df}$, so it would bring the point estimate to zero when 'bias_index' exceeds $|t|/\sqrt{df}$. Defined for multi-column terms (factors) as well.

The returned data frame carries an attribute 'threshold' for the selected 'metric', so it can be passed straight to [plot_sensitivity_love()]. Its 'r_yu', 'r_du', and 'label' columns can be passed as 'benchmarks' to [plot_sensitivity_contour()].

Covariate benchmarks describe *observed* variables. Whether an unobserved confounder could be as strong as a given covariate is a substantive judgment that no benchmark can settle.

Each model term is removed as a whole. A factor entered as one term (for example 'factor(region)') is therefore benchmarked jointly: its partial R^2 values and bias index use all of its columns, while the ITCV correlations, which need a single signed column, are 'NA'. An ordered factor is also treated as one term, through all of its contrasts, so its ordering does not change the benchmark. To use the ordering, enter a numeric score instead (one column). Dummy variables entered separately, as in a pre-expanded data set, are benchmarked one at a time, each against its omitted category.

Value

A data frame with one row per covariate term and columns 'covariate', 'term_df', 'r_du', 'r_yu', 'itcv_impact', 'r_du_zero', 'r_yu_zero', 'impact_zero', 'r2dz.x', 'r2yz.dx', 'bias_index', 'itcv_ratio', 'impact', and 'label', and attributes 'threshold', 'metric', 'itcv', and 'f_threshold'. 'r_du_zero' and 'r_yu_zero' are the zero-order (unadjusted) correlations of a single-column term with treatment and outcome, 'impact_zero' their signed product, and 'itcv_ratio' the unconditional ITCV divided by 'impact_zero' (Frank et al., 2025, eq. 10), reported for terms whose impact acts against the estimate. A ratio above 1 means an omitted variable would need a larger impact than that covariate.

References

Frank, K. A., Saw, G. K., Lin, Q., Xu, R., Rosenberg, J. M., Maroulis, S. J., & Staudt Willet, K. B. (2025). *A practical guide to impact threshold of a confounding variable (ITCV) and robustness of inference to replacement (RIR)* (Version 2). doi:10.31234/osf.io/pdgbz_v2

Examples

```
fit <- lm(mpg ~ am + wt + hp + qsec, data = mtcars)
imp <- covariate_impacts(fit, "am")
imp
plot_sensitivity_love(imp)
```

from_evalue *Convert an E-value result to a confoundsens path*

Description

Builds a bias-adjusted risk-ratio path from an ‘evalue‘ object (as returned by ‘EValue::evalues.RR()‘, ‘evalues.OR()‘, ‘evalues.HR()‘, ‘evalues.OLS()‘, and related functions). Confounder strength λ is the common risk ratio relating the confounder to both treatment and outcome; the corresponding bias factor is $B = \lambda^2/(2\lambda - 1)$ (VanderWeele and Ding, 2017). The path is on the log risk-ratio scale, so it crosses the null (0) exactly at the E-value.

Usage

```
from_evalue(x, lambda = seq(1, 5, by = 0.02))
```

Arguments

x	An object of class ‘"evalue"‘.
lambda	Numeric vector of confounder risk ratios (>= 1).

Value

A ‘confoundsens‘ object with ‘theta‘ = log of the bias-adjusted point risk ratio and, when the confidence interval is available, ‘se‘ on the log scale.

References

VanderWeele, T. J., & Ding, P. (2017). Sensitivity analysis in observational research: Introducing the E-value. *Annals of Internal Medicine*, 167(4), 268–274. doi:10.7326/M162607

Examples

```
e <- EValue::evalues.RR(est = 1.8, lo = 1.3, hi = 2.5)
path <- from_evalue(e)
robustness_points(path)
```

from_iv_sensemakr *Convert an iv.sensemakr result to a confoundsens path*

Description

Imports an instrumental-variable sensitivity analysis from ‘iv.sensemakr‘ (Cinelli and Hazlett, 2025). The confounder in this framework is an omitted variable W that makes the instrument invalid, described by two partial R^2 values: $R^2_{Z\sim W|X}$ (with the instrument) and $R^2_{Y_0\sim W|Z,X}$ (with the potential outcome under no treatment). Inference uses the Anderson–Rubin confidence set, which stays valid when the instrument is weak.

Usage

```
from_iv_sensemakr(x, lambda = NULL)
```

Arguments

x	An object of class ‘"iv.sensemakr"‘, as returned by ‘iv.sensemakr::sensemakr()‘ applied to an ‘iv_fit()‘ result.
lambda	Optional grid of equal partial R^2 values in $[0, 1)$. By default 2001 points from 0 to three times the larger robustness value (at least 0.02).

Details

The path follows the equal-strength line $\lambda = R^2_{Z\sim W|X} = R^2_{Y_0\sim W|Z,X}$, the line on which ‘iv.sensemakr‘ defines its robustness value. ‘theta‘ is the limit, nearest the null h_0, of the set of IV estimates compatible with the confounding (‘iv.sensemakr‘ at ‘alpha = 1‘); it starts at the IV point estimate and reaches h_0 at the robustness value for the point estimate. ‘meta$ci_near‘ holds the bias-adjusted Anderson–Rubin confidence limit nearest h_0 at the level of the ‘iv.sensemakr‘ object; it reaches h_0 at the robustness value $RV_{q,\alpha}$ that ‘iv.sensemakr‘ reports. [robustness_points()] uses it for the significance crossing. An infinite limit means that the adjusted confidence set is unbounded.

The robustness values and benchmark bounds of the ‘iv.sensemakr‘ object are kept in ‘meta‘ (‘rv‘, ‘xrv‘, ‘rv_point‘, ‘bounds‘). The significance level is fixed when the ‘iv.sensemakr‘ object is created; to use another level, call ‘iv.sensemakr::sensemakr(..., alpha =)‘ again.

Value

A ‘confoundsens‘ object with ‘meta$framework = "iv"‘.

References

Cinelli, C., & Hazlett, C. (2025). An omitted variable bias framework for sensitivity analysis of instrumental variables. *Biometrika*, 112(2), asaf004. doi:10.1093/biomet/asaf004

See Also

[sens_surface()] for the two-parameter plane.

Examples

```
data("card", package = "iv.sensemakr")
xm <- model.matrix(~ exper + expersq + black + south + smsa + reg661 +
  reg662 + reg663 + reg664 + reg665 + reg666 + reg667 + reg668 + smsa66,
  data = card)
fit <- iv.sensemakr::iv_fit(card$lwage, card$educ, card$nearc4, xm)
s <- iv.sensemakr::sensemakr(fit, benchmark_covariates = c("black", "smsa"))
path <- from_iv_sensemakr(s)
robustness_points(path)
sens_report(path)
```

`from_konfound` *Convert konfound output to a confoundsens path*

Description

Builds a path from the output of ‘konfound::konfound()‘ or ‘konfound::pkonfound()‘ called with ‘to_return = "raw_output"‘.

Usage

```
from_konfound(
  x,
  dof = NULL,
  lambda = seq(0, 0.5, by = 0.01),
  treatment = NA_character_,
  index = c("IT", "RIR"),
  alpha = 0.05
)
```

Arguments

`x`	A list returned by ‘konfound()‘/‘pkonfound()‘ with ‘to_return = "raw_output"‘.
`dof`	Optional residual degrees of freedom (ITCV only). If supplied, the path also carries adjusted t statistics.
`lambda`	Numeric vector of impacts in \[0, 1) for ‘index = "IT"‘, or of replacement proportions in \[0, 1\] for ‘index = "RIR"‘ (default ‘seq(0, 1, by = 0.001)‘ when not supplied).
`treatment`	Optional label for the treatment.
`index`	‘"IT"‘ (default) or ‘"RIR"‘.
`alpha`	The significance level used in the ‘konfound‘ call (konfound’s default is 0.05). It is recorded with the path so that [robustness_points()] and [sens_report()] report the right level; it does not change konfound’s threshold. Use ‘NA‘ if the threshold was not set from a significance level.

Details

With ‘index = "IT"‘ (default), the impact threshold (ITCV) path is built from the observed and critical correlations (‘obs_r‘, ‘critical_r‘). With ‘index = "RIR"‘, the robustness-of-inference-to-replacement path is built from ‘beta_threshold‘ and ‘perc_bias_to_change‘, and konfound’s ‘RIR_primary‘ is carried in ‘meta$rir‘; see [rir_lm()] for the path.

Value

A ‘confoundsens‘ object. For ‘index = "IT"‘ it is on the correlation scale and ‘meta$itcv‘ holds the conditional ITCV (konfound’s ‘itcvGz‘); for ‘index = "RIR"‘ it is on the scale of the estimate and ‘meta$perc_bias‘ and ‘meta$rir‘ hold konfound’s percent bias and RIR.

See Also

[itcv_lm()], [rir_lm()]

Examples

```
k <- konfound::pkonfound(est_eff = 2, std_err = 0.4, n_obs = 100,
                         n_covariates = 3, index = "IT",
                         to_return = "raw_output")
path <- from_konfound(k)
path$meta$itcv
rir <- from_konfound(k, index = "RIR")
rir$meta$rir
```

`from_sensemakr` *Convert a sensemakr result to a confoundsens path*

Description

Builds the partial R^2 sensitivity path from the summary statistics stored in a ‘sensemakr‘ object (estimate, standard error, and residual degrees of freedom), using the same bias formulas as [sens_path_lm()]. Benchmark bounds stored in the object are attached as ‘meta$bounds‘.

Usage

```
from_sensemakr(x, lambda = seq(0, 0.5, by = 0.01), ratio = 1)
```

Arguments

`x`	An object of class ‘"sensemakr"‘ (from ‘sensemakr::sensemakr()‘).
`lambda`	Numeric vector of confounder strengths in \[0, 1). Default is a grid from 0 to 0.5.
`ratio`	Positive scalar; ratio of the outcome-side to the treatment-side partial R^2.

Value

A ‘confoundsens‘ object.

See Also

[sens_path_lm()]

Examples

```
data("darfur", package = "sensemakr")
fit <- lm(peacefactor ~ directlyharmed + age + farmer_dar + herder_dar +
            pastvoted + hhsize_darfur + female + village, data = darfur)
s <- sensemakr::sensemakr(fit, treatment = "directlyharmed",
                          benchmark_covariates = "female", kd = 1:3)
path <- from_sensemakr(s)
sens_report(path)
```

`itcv_lm` *Impact threshold for a confounding variable from a fitted linear model*

Description

Computes Frank's (2000) impact threshold for a confounding variable (ITCV) for one coefficient of a linear model, together with the ITCV sensitivity path: the partial correlation between treatment and outcome after conditioning on a confounder whose impact $k = r_{DU} r_{YU}$ is split equally between the two correlations ($r_{DU} = r_{YU} = \sqrt{k}$).

Usage

```
itcv_lm(model, treatment, alpha = 0.05, lambda = seq(0, 0.5, by = 0.01))
```

Arguments

`model`	A fitted 'lm' object.
`treatment`	Character string naming the treatment coefficient.
`alpha`	Two-sided significance level used for the critical value.
`lambda`	Numeric vector of impacts in \[0, 1) at which to evaluate the path.

Details

Along this path the adjusted correlation is $(r - k)/(1 - k)$, which equals the critical correlation $r^{\#}$ exactly at $k = \text{ITCV} = (|r| - |r^{\#}|)/(1 - |r^{\#}|)$.

The conditional ITCV equals 'itcvGz' and the unconditional ITCV equals 'itcv' in the raw output of 'konfound::konfound()' (version 1.0).

Value

A list with elements 'r' (observed partial correlation), 'r_crit' (critical partial correlation), 'itcv' (the ITCV for a confounder's correlations *conditional on* the model covariates), 'dof', 'path' (a 'confoundsens' object on the correlation scale), 'r2xz' and 'r2yz' (the R^2 of the covariates with treatment and outcome), and 'itcv_unconditional' (the ITCV expressed in unconditional correlations, 'itcv * sqrt((1 - r2xz) * (1 - r2yz))'), 'benchmark_corr_product' (the impact of all covariates together, $R_{XZ} R_{YZ}$, the product of the multiple correlations), and 'itcv_ratio_to_benchmark'

('itcv_unconditional / benchmark_corr_product'). The ratio follows eq. (12) of Frank et al. (2025): values above 1 mean that an omitted variable would need a larger impact than all observed covariates together. Covariates such as pretests or fixed effects inflate the benchmark.

References

Frank, K. A. (2000). Impact of a confounding variable on a regression coefficient. *Sociological Methods & Research*, 29(2), 147–194. doi:10.1177/0049124100029002001

Frank, K. A., Maroulis, S. J., Duong, M. Q., & Kelcey, B. M. (2013). What would it take to change an inference? Using Rubin's causal model to interpret the robustness of causal inferences. *Educational Evaluation and Policy Analysis*, 35(4), 437–460. doi:10.3102/0162373713493129

Frank, K. A., Saw, G. K., Lin, Q., Xu, R., Rosenberg, J. M., Maroulis, S. J., & Staudt Willet, K. B. (2025). *A practical guide to impact threshold of a confounding variable (ITCV) and robustness of inference to replacement (RIR)* (Version 2). doi:10.31234/osf.io/pdgbz_v2

See Also

[from_konfound()], [sens_path_lm()], [rir_lm()]

Examples

```
fit <- lm(mpg ~ am + wt + hp, data = mtcars)
it <- itcv_lm(fit, "am")
it$itcv
plot_robustness_curve(it$path, points = FALSE)
```

---

new_confoundsens *Create a confoundsens object*

Description

Constructs a 'confoundsens' object — a lightweight container for storing sensitivity paths (e.g., $\theta(\lambda)$) and optional uncertainty or stratification information used by 'confoundvis' plotting functions.

Usage

```
new_confoundsens(
  lambda,
  theta,
  level = NULL,
  se = NULL,
  t = NULL,
  meta = list()
)
```

Arguments

lambda	Numeric vector of sensitivity strength values (e.g., ITCV lambda or delta). Should be nondecreasing; a warning is issued otherwise.
theta	Numeric vector of effect estimates along the sensitivity path. Must be the same length as ‘lambda‘.
level	Optional character (or coercible) vector of subgroup (stratum) labels (e.g., ‘"female"‘, ‘"male"‘). Must be the same length as ‘lambda‘.
se	Optional numeric vector of standard errors for ‘theta‘. Must be the same length as ‘lambda‘.
t	Optional numeric vector of test statistics along the path. Must be the same length as ‘lambda‘.
meta	Optional named list of metadata describing the path, for example ‘framework‘ (‘"partial_r2"‘, ‘"itcv"‘, ‘"evalue"‘, ...), ‘lambda_label‘, ‘theta_label‘, ‘null‘ (the value of ‘theta‘ that represents no effect), and ‘estimate‘. Used by printing, plotting, and [sens_report()]; ignored otherwise.

Value

A ‘confoundsens‘ object (a list with class ‘"confoundsens"‘).

Examples

```
x <- new_confoundsens(
  lambda = seq(0, 0.2, length.out = 10),
  theta  = seq(1, 0.6, length.out = 10),
  se     = rep(0.1, 10)
)
x
```

---

plot.confoundvis_profile

Plot a sensitivity profile

Description

One panel per analysis, each with its own axes. The panel title names the framework and states what the horizontal axis (x, the sensitivity parameter) and the vertical axis (y, the path value) measure in that panel. The dotted line is the null value, the dashed line is the stored threshold (when the analysis has one), and the vertical long-dashed line marks the confounder strength at which significance is lost.

Usage

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

Arguments

x	A ‘confoundvis_profile‘ object from [sens_profile()].
...	Unused.

Value

A [ggplot2::ggplot()] object.

`plot.confoundvis_result`
Plot a confoundvis result

Description

Plot a confoundvis result

Usage

```
## S3 method for class 'confoundvis_result'
plot(
  x,
  which = c("paths", "benchmarks", "benchmarks_itcv", "surface"),
  top = 15,
  ...
)
```

Arguments

x	A ‘confoundvis_result‘ from [confoundvis()].
which	‘"paths"‘ (default) for the sensitivity paths, ‘"benchmarks"‘ for the measured-covariate Love plot (partial-R^2 metric), ‘"benchmarks_itcv"‘ for the same on the ITCV metric, or ‘"surface"‘ for the two-parameter surface of the first analysis that has one.
top	Number of covariates shown in the benchmark plots.
...	Unused.

Value

A [ggplot2::ggplot()] object.

`plot_robustness_curve` *Plot a robustness curve*

Description

Plots the sensitivity path stored in a ‘confoundsens‘ object as a function of ‘lambda‘. By default, plots the effect path ‘theta(lambda)‘; optionally plots the test-statistic path ‘t(lambda)‘. Pointwise confidence bands are drawn when standard errors are available.

Usage

```
plot_robustness_curve(
  x,
  what = c("theta", "t"),
  bands = TRUE,
  points = TRUE,
  facet_level = TRUE,
  reference = TRUE
)
```

Arguments

`x`	A ‘confoundsens‘ object created by [new_confoundsens()] or [as_confoundsens()].
`what`	Character; which path to plot: ‘"theta"‘ (default) or ‘"t"‘.
`bands`	Logical; if ‘TRUE‘ and ‘x$se‘ is present, draw pointwise confidence bands (applies only when ‘what = "theta"‘), at the level the path was built with when it stores one (for example ‘sens_risk_difference(..., alpha = 0.01)‘ gives 99 otherwise. Paths from [from_iv_sensemakr()] and [sens_dml()] carry their own confidence limits (bias-adjusted Anderson–Rubin limits; ‘doubleml‘ limits of the lower and upper bounds), which are drawn instead, and the vertical range is set from the unadjusted interval because these limits can be infinite.
`points`	Logical; if ‘TRUE‘, overlay points on the line.
`facet_level`	Logical; if ‘TRUE‘ and ‘x$level‘ is present, facet the plot by level.
`reference`	Logical; if ‘TRUE‘ (default) and ‘x$meta‘ records a null value (‘meta$null‘) or a critical value (‘meta$threshold‘), draw them as horizontal reference lines (dotted and dashed respectively). Objects created by [new_confoundsens()] without ‘meta‘ are unaffected.

Value

A [ggplot2::ggplot()] object.

Examples

```
x <- new_confoundsens(
  lambda = seq(0, 0.2, length.out = 25),
  theta  = 1 - 2 * seq(0, 0.2, length.out = 25),
  se     = rep(0.05, 25),
  level  = rep(c("female", "male"), length.out = 25)
)
plot_robustness_curve(x)
plot_robustness_curve(x, bands = FALSE, facet_level = FALSE)
```

`plot_sensitivity_contour` *Sensitivity contour plot*

Description

Draws sensitivity boundaries in the plane of the two confounder correlations (r_{YU}, r_{DU}) and optionally overlays observed covariate benchmarks as labelled points.

Usage

```
plot_sensitivity_contour(
  threshold = NULL,
  grid_n = 200,
  benchmarks = NULL,
  itcv = NULL,
  limits = c(-1, 1)
)
```

Arguments

`threshold`	Positive numeric scalar; the product threshold. The product reference satisfies $\lvert r_{YU} \cdot r_{DU} \rvert =$'threshold'. May be omitted when 'itcv' is supplied, in which case the conditional ITCV is used.
`grid_n`	Integer >= 50; resolution of the curves (points per branch for the product reference; grid points per axis for the exact boundary and shading).
`benchmarks`	Optional data.frame with columns 'r_yu' and 'r_du' (numeric) and an optional 'label' column (character). Each row is plotted as a labelled benchmark point; labels repel one another when the **ggrepel** package is installed. The output of [covariate_impacts()] with 'metric = "itcv"' can be passed directly.
`itcv`	Optional output of [itcv_lm()] (a list with 'r' and 'r_crit'), or a list or named numeric vector with elements 'r' and 'r_crit'. When supplied, the exact boundary and the infeasible region are drawn.
`limits`	Numeric vector of length two; the displayed range of both axes. Narrow it (for example 'c(-0.3, 0.3)') when benchmarks lie close to the origin.

Details

Two kinds of boundary are available.

* **Product reference** (always drawn, dashed when an exact boundary is also drawn): the four branches of $|r_{YU} r_{DU}| =$ ‘threshold‘. With ‘threshold‘ equal to the ITCV, this is the impact threshold of Frank (2000). It is exact for confounders whose two correlations have equal magnitude, but for unequal correlations the product alone does not determine whether significance is lost. * **Exact boundary** (when ‘itcv‘ is supplied): the set of pairs at which the confounder-adjusted partial correlation

$$r_{\text{adj}} = \frac{r - r_{DU} r_{YU}}{\sqrt{(1 - r_{DU}^2)(1 - r_{YU}^2)}}$$

has magnitude equal to the critical correlation. The solid curve is the boundary in the direction of the original estimate; beyond it the estimate is no longer significant ("Not significant" shading). The dotted curve is the boundary in the reversed direction; beyond it the adjusted estimate is significant with the opposite sign. Both curves are computed in closed form. Pairs for which the correlation matrix of (D, Y, U) given the covariates would not be positive definite, $1 - r^2 - r_{DU}^2 - r_{YU}^2 + 2 r r_{DU} r_{YU} \leq 0$, cannot occur and are shaded as infeasible.

Value

A [ggplot2::ggplot()] object.

References

Frank, K. A. (2000). Impact of a confounding variable on a regression coefficient. *Sociological Methods & Research*, 29(2), 147–194. doi:10.1177/0049124100029002001

Examples

```
b <- data.frame(
  r_yu  = c(0.10, 0.15),
  r_du  = c(0.20, 0.12),
  label = c("SES", "BMI")
)
plot_sensitivity_contour(threshold = 0.02, benchmarks = b)

# Exact boundary and infeasible region from a fitted model
fit <- lm(mpg ~ am + wt + hp, data = mtcars)
plot_sensitivity_contour(itcv = itcv_lm(fit, "wt"))
```

`plot_sensitivity_love` *Sensitivity Love plot*

Description

Benchmarks a sensitivity threshold against the empirical distribution of observed covariate impacts in a "Love plot"-style display. Each covariate appears as a point on a horizontal impact axis; the sensitivity threshold is shown as a vertical reference line. Covariates to the left of the line are weaker than the threshold; those to the right pose a credible threat.

Usage

```
plot_sensitivity_love(
  df,
  threshold = attr(df, "threshold"),
  sort = TRUE,
  top = NULL
)
```

Arguments

df	A data.frame with at least two columns: * 'covariate' — covariate names (character or factor). * 'label' — optional display labels; used in place of 'covariate' when present (as set by the 'labels' argument of [covariate_impacts()]). * 'impact' — numeric impact values (e.g., ITCV product $\|r_{YU} \cdot r_{DU}\|$, partial R^2, or other confounding-strength metric).
threshold	Single non-missing numeric value. Defaults to the '"threshold"' attribute of 'df', which [covariate_impacts()] sets. Drawn as a vertical reference line (the sensitivity threshold). Rows with missing 'impact' are dropped.
sort	Logical; if 'TRUE' (default), sort covariates by 'impact' on the y-axis (ascending).
top	Optional positive integer. If supplied, only the 'top' covariates with the largest absolute impact are displayed.

Value

A [ggplot2::ggplot()] object.

Examples

```
df <- data.frame(
  covariate = c("SES", "BMI", "Gender", "Race", "Age"),
  impact    = c(0.12, 0.05, 0.02, 0.08, 0.03)
)
plot_sensitivity_love(df, threshold = 0.10)
plot_sensitivity_love(df, threshold = 0.10, top = 3)

# From a fitted model: threshold is taken from covariate_impacts()
fit <- lm(mpg ~ am + wt + hp + qsec, data = mtcars)
plot_sensitivity_love(covariate_impacts(fit, "am"))
```

---

plot_sensitivity_surface

Plot a two-parameter sensitivity surface

Description

Draws a [sens_surface()] object: shaded zones where the conclusion changes, the boundaries (solid for loss of significance, dotted for significance in the reversed direction, dashed for the null), and optional benchmark points. Unshaded cells are those where the original conclusion holds.

Usage

```
plot_sensitivity_surface(x, benchmarks = NULL, bounds = TRUE)
```

Arguments

x	A ‘confoundsurf‘ object.
benchmarks	Optional data frame of reference points with columns ‘a‘ and ‘b‘ (and optionally ‘label‘) on the axes of the surface. For a partial-R^2 surface, the output of ‘covariate_impacts(..., metric = "partial_r2")‘ is accepted (columns ‘r2dz.x‘ and ‘r2yz.dx‘); for an ITCV surface, the output of ‘covariate_impacts(..., metric = "itcv")‘ (columns ‘r_du‘ and ‘r_yu‘). For a surface built from a ‘sensemakr‘ path, its formal bounds are added automatically unless ‘bounds = FALSE‘.
bounds	Logical; draw ‘sensemakr‘ formal bounds when present.

Value

A [ggplot2::ggplot()] object.

See Also

[sens_surface()]

Examples

```
fit <- lm(mpg ~ am + wt + hp, data = mtcars)
s <- sens_surface(sens_path_lm(fit, "wt"))
plot_sensitivity_surface(s, benchmarks = covariate_impacts(fit, "wt",
                                                           "partial_r2"))
```

---

print.confoundsens *Print method for confoundsens objects*

Description

Prints a concise summary of the key fields in a ‘confoundsens‘ object.

Usage

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

Arguments

x	A ‘confoundsens‘ object.
...	Unused; included for S3 method consistency.

Value

‘x‘, invisibly.

Examples

```
x <- new_confoundsens(
  lambda = seq(0, 0.2, length.out = 5),
  theta  = seq(1, 0.8, length.out = 5)
)
print(x)
```

---

rir_lm *Robustness of inference to replacement (RIR) from a linear model*

Description

Computes the robustness of inference to replacement of Frank et al. (2013) for a coefficient in a fitted linear model, and the corresponding path. Suppose a proportion λ of the cases were replaced by cases in which the treatment has no effect. The estimate would then be $(1-\lambda)\hat{\beta}$. The inference is nullified when this reaches the threshold for statistical significance $\beta^{\#} = \operatorname{sign}(\hat{\beta})\, t_{\mathrm{crit}}\, SE$, with the standard error held fixed. The crossing $1 - \beta^{\#}/\hat{\beta}$ is the proportion of the estimate that would have to be due to bias, and RIR is the number of cases this represents. The calculation follows ‘konfound::konfound()‘ with ‘index = "RIR"‘, using the residual degrees of freedom of the model.

Usage

```
rir_lm(
  model,
  treatment,
  alpha = 0.05,
  lambda = seq(0, 1, by = 0.001),
  threshold = NULL
)
```

Arguments

model	A fitted ‘lm‘ object.
treatment	Character string naming the treatment coefficient (a column of the model matrix).
alpha	Two-sided significance level.
lambda	Numeric vector of replacement proportions in \[0, 1\].

`threshold` Optional effect threshold on the scale of the estimate (for example, the smallest effect that would justify a policy). When supplied, it replaces the significance threshold, as with ‘eff_thr‘ in konfound; its sign is taken from the estimate.

Value

A list with ‘estimate‘, ‘beta_threshold‘, ‘perc_bias‘ (percent of the estimate that would have to be due to bias), ‘rir‘ (number of cases), ‘n_obs‘, and ‘path‘, a ‘confoundsens‘ object whose ‘lambda_sig‘ in [robustness_points()] equals ‘perc_bias / 100‘. For an estimate that is not significant, ‘perc_bias‘ and ‘rir‘ are ‘NA‘; ‘konfound‘ reports the corresponding replacement needed to sustain an inference.

References

Frank, K. A., Maroulis, S. J., Duong, M. Q., & Kelcey, B. M. (2013). What would it take to change an inference? Using Rubin’s causal model to interpret the robustness of causal inferences. *Educational Evaluation and Policy Analysis*, 35(4), 437–460. doi:10.3102/0162373713493129

Frank, K. A., Saw, G. K., Lin, Q., Xu, R., Rosenberg, J. M., Maroulis, S. J., & Staudt Willet, K. B. (2025). *A practical guide to impact threshold of a confounding variable (ITCV) and robustness of inference to replacement (RIR)* (Version 2). doi:10.31234/osf.io/pdgbz_v2

See Also

[itcv_lm()], [from_konfound()]

Examples

```
fit <- lm(mpg ~ am + wt + hp, data = mtcars)
rr <- rir_lm(fit, "wt")
rr[c("beta_threshold", "perc_bias", "rir")]
# a substantive threshold instead of statistical significance
rir_lm(fit, "wt", threshold = 2)$rir
sens_report(rr$path)
```

`robustness_points` *Locate robustness thresholds along a sensitivity path*

Description

Finds, for each level of a ‘confoundsens‘ object, the confounder strength λ at which (i) the point estimate reaches the null value and (ii) statistical significance is lost. Crossings are located by linear interpolation between grid points, so their precision depends on the spacing of ‘lambda‘.

Usage

```
robustness_points(x, alpha = NULL)
```

Arguments

x	A ‘confoundsens‘ object. Paths with a stored threshold (ITCV, RIR, and Rosenbaum paths) fix the significance level when they are built, because the threshold is computed from it. For these paths ‘alpha‘ defaults to the stored level. Asking for a different level is an error: rebuild the path at the new level instead (for example ‘itcv_lm(model, treatment, alpha = 0.01)‘). Instrumental variable paths are fixed at the level of the ‘iv.sensemakr‘ object in the same way. For other paths the default is the level stored in ‘meta$default_alpha‘ ([sens_dml()] stores the two-sided level that matches the one-sided ‘doubleml‘ level), or 0.05. The level actually used is returned in column ‘alpha‘.
alpha	Significance level, or ‘NULL‘ (default) to use the level stored in the path, if any, and 0.05 otherwise. Two-sided, except for Rosenbaum paths, which use the one-sided level of Rosenbaum’s test.

Details

The significance criterion is chosen from what the object carries, in this order: bias-adjusted confidence limits in ‘meta$ci_near‘ (instrumental variable paths from [from_iv_sensemakr()]); a critical value on the ‘theta‘ scale in ‘meta$threshold‘ (ITCV paths); adjusted t statistics with residual degrees of freedom in ‘meta$dof‘ (partial R^2 paths; critical value from a t distribution with ‘dof - 1‘ degrees of freedom, as in Cinelli and Hazlett, 2020); standard errors (normal critical value); or t statistics without degrees of freedom (normal critical value).

Value

A data frame with columns ‘level‘, ‘theta_start‘, ‘lambda_null‘, ‘lambda_sig‘, ‘sig_rule‘, and ‘alpha‘. ‘NA‘ means the crossing is not reached within the supplied ‘lambda‘ range.

Examples

```
fit <- lm(mpg ~ am + wt + hp, data = mtcars)
robustness_points(sens_path_lm(fit, "am", lambda = seq(0, 0.9, by = 0.001)))
```

---

sens_dml *Sensitivity analysis for a DoubleML fit*

Description

Computes the omitted-variable-bias sensitivity analysis of Chernozhukov et al. (2024) for a partially linear (‘DoubleMLPLR‘) or interactive (‘DoubleMLIRM‘, average treatment effect) model fitted with the R package ‘DoubleML‘, and returns it as a ‘confoundsens‘ path. The R package does not provide this analysis; the Python package ‘doubleml‘ does, and ‘sens_dml()‘ reproduces its ‘sensitivity_analysis()‘ from the stored cross-fitted predictions, following its formulas and its aggregation over repeated sample splits.

Usage

```
sens_dml(x, lambda = NULL, rho = 1, level = 0.95, null = 0, treatment = NULL)
```

Arguments

x	A fitted ‘DoubleMLPLR‘ or ‘DoubleMLIRM‘ object.
lambda	Optional grid of equal ‘cf_y = cf_d‘ values in $[0, 1)$. By default 2001 points from 0 to 1.5 times the larger robustness value (at least 0.05, at most 0.99).
rho	Correlation between the bias components, in $[-1, 1]$.
level	One-sided confidence level, as in ‘doubleml‘.
null	Null value of the effect.
treatment	Treatment column, by name or position, when the model has several treatments.

Details

The confounder is described by two parameters: ‘cf_y‘, the share of the residual variation of the outcome explained by the omitted confounder, and ‘cf_d‘, the relative gain in the variation of the Riesz representer, together with the correlation ‘rho‘ between the two bias components ($|\rho| = 1$ is the worst case). The bias bound is $|\rho| \sqrt{c_y c_d / (1 - c_d)}\, S$, where $S^2 = \sigma^2 \nu^2$ is estimated from the cross-fitted residuals.

The path follows the equal-strength line $\lambda = c_y = c_d$, the line on which ‘doubleml‘ defines its robustness values. ‘theta‘ is the bound nearest the null (the lower bound when the estimate is above ‘null‘); it reaches ‘null‘ at the robustness value RV. ‘se‘ is the standard error of that bound, defined so that ‘theta - qnorm(level) * se‘ equals the ‘doubleml‘ confidence limit. ‘doubleml‘ uses a one-sided limit at confidence ‘level‘; this corresponds to a two-sided level $\alpha = 2(1 - \text{level})$, which is stored as the default significance level of the path (0.10 for ‘level = 0.95‘), so that [robustness_points()] reproduces the robustness value RVa. Supply ‘alpha‘ to [robustness_points()] to use another level; with several repeated sample splits the result is then an approximation, because ‘doubleml‘ takes medians of the confidence limits.

The fit must have been made with ‘fit(store_predictions = TRUE)‘. Clustered data, user-supplied score functions, and the ATTE score are not supported. For ‘DoubleMLIRM‘, propensity scores are truncated with the fit’s own ‘trimming_threshold‘, as in its score. The analysis depends on the learners and the sample split, like the estimate itself.

Value

A ‘confoundsens‘ object with ‘meta$framework = "dml"‘, carrying the robustness values ‘rv‘ and ‘rva‘, the bias scale ‘max_bias‘ (S), the confidence limits (‘ci_lower‘, ‘ci_upper‘), and the elements needed by [sens_surface()].

References

Chernozhukov, V., Cinelli, C., Newey, W., Sharma, A., & Syrgkanis, V. (2024). Long story short: Omitted variable bias in causal machine learning. NBER Working Paper 30302. doi:10.3386/w30302

Bach, P., Chernozhukov, V., Kurz, M. S., & Spindler, M. (2022). DoubleML: An object-oriented implementation of double machine learning in Python. *Journal of Machine Learning Research*, 23(53), 1–6.

See Also

[sens_surface()] for the plane of ‘cf_d‘ and ‘cf_y‘.

Examples

```
library(DoubleML)
set.seed(1)
dat <- make_plr_CCDDHNR2018(n_obs = 500, alpha = 0.5, return_type = "data.table")
obj <- DoubleMLData$new(dat, y_col = "y", d_cols = "d")
plr <- DoubleMLPLR$new(obj, ml_l = mlr3::lrn("regr.lm"),
                       ml_m = mlr3::lrn("regr.lm"), n_folds = 5)
plr$fit(store_predictions = TRUE)
path <- sens_dml(plr)
robustness_points(path)
sens_report(path)
```

`sens_path_lm` *Sensitivity path from a fitted linear model (partial R-squared framework)*

Description

Computes the bias-adjusted treatment coefficient as a function of the strength of a hypothetical omitted confounder, using the omitted-variable bias formulas of Cinelli and Hazlett (2020). Confounder strength is indexed by λ, the partial R^2 of the confounder with the treatment ($R^2_{D\sim Z|X}$); the partial R^2 with the outcome is $R^2_{Y\sim Z|D,X} = \min(\texttt{ratio} \times \lambda, 1)$. The default ‘ratio = 1‘ gives the equal-strength path along which the robustness value is defined.

Usage

```
sens_path_lm(model, treatment, lambda = seq(0, 0.5, by = 0.01), ratio = 1)
```

Arguments

`model`	A fitted ‘lm‘ object.
`treatment`	Character string naming the treatment coefficient (a column of the model matrix).
`lambda`	Numeric vector of confounder strengths in \[0, 1). Default is a grid from 0 to 0.5.
`ratio`	Positive scalar; ratio of the outcome-side to the treatment-side partial R^2.

Details

The confounder is assumed to act in the direction that shrinks the estimate toward zero (the conventional worst case for a reported effect).

Value

A ‘confoundsens‘ object with ‘theta‘ (adjusted estimate), ‘se‘ (adjusted standard error), and ‘t‘ (adjusted t statistic), plus ‘meta‘ recording the original estimate, degrees of freedom, and robustness value.

References

Cinelli, C., & Hazlett, C. (2020). Making sense of sensitivity: Extending omitted variable bias. *Journal of the Royal Statistical Society: Series B*, 82(1), 39–67. doi:10.1111/rssb.12348

See Also

[from_sensemakr()], [itcv_lm()], [covariate_impacts()]

Examples

```
fit <- lm(mpg ~ am + wt + hp, data = mtcars)
path <- sens_path_lm(fit, treatment = "am")
path
plot_robustness_curve(path, points = FALSE)
```

`sens_path_pairs` *Rosenbaum sensitivity path for matched pairs*

Description

Computes a sensitivity path for a matched-pair observational study under the sensitivity model of Rosenbaum (2002): within a pair, the odds of treatment for the two units may differ by at most a factor $\Gamma \geq 1$. For each value of Γ, the function computes the sharp upper bound on the one-sided P-value of a randomization test of no effect and stores it as an equivalent standard normal deviate, so that the path can be drawn, located, tabulated, and reported with the same functions as the other frameworks.

Usage

```
sens_path_pairs(
  d,
  gamma = seq(1, 3, by = 0.01),
  method = c("wilcoxon", "binary"),
  alpha = 0.05
)
```

Arguments

d	Numeric vector of treated-minus-control differences, one per matched pair. For ‘method = "binary"‘, values must be in {-1, 0, 1}.
gamma	Numeric vector of sensitivity parameters, all ‘>= 1‘.
method	‘"wilcoxon"‘ (default) or ‘"binary"‘.
alpha	One-sided significance level used for the stored threshold and the sensitivity value.

Details

Two statistics are available. ‘method = "wilcoxon"‘ uses Wilcoxon’s signed rank statistic with the large-sample approximation of Rosenbaum (2002, Sec. 4.3.3), computed as in ‘DOS2::senWilcox()‘: all absolute differences are ranked (ties receive average ranks) and zero differences then receive rank zero. When some differences are zero, the statistic and P-values therefore differ from those of ‘wilcox.test()‘ and ‘rbounds::psens()‘, which drop zeros before ranking. ‘method = "binary"‘ is for a binary outcome (differences in {-1, 0, 1}) and uses the exact binomial bound for McNemar’s statistic (Rosenbaum 2002, Sec. 4.3.2), which agrees with ‘sensitivity2x2xk::mh()‘ applied to the pairs as strata.

The test is one-sided in the direction of ‘d > 0‘ (treated minus control). Supply ‘-d‘ to test the opposite direction.

Value

A ‘confoundsens‘ object with ‘lambda‘ = Γ and ‘theta‘ = the standard normal deviate equivalent to the upper-bound P-value. ‘meta$threshold‘ is ‘qnorm(1 - alpha)‘, so [robustness_points()] reports as ‘lambda_sig‘ the value of Γ at which the upper bound reaches ‘alpha‘, and as ‘lambda_null‘ the value at which it reaches 0.5. ‘meta$sens_value‘ holds the sensitivity value computed by root finding rather than on the grid, and ‘meta$pvalue‘ the upper-bound P-values along the grid.

References

Rosenbaum, P. R. (2002). *Observational Studies* (2nd ed.). Springer. doi:10.1007/97814757-36922

See Also

[robustness_points()], [sens_table()]

Examples

```
set.seed(1)
d <- rnorm(60, mean = 0.5)
p <- sens_path_pairs(d, gamma = seq(1, 4, by = 0.01))
robustness_points(p)
sens_report(p)

# Binary outcome: 30 discordant pairs, 22 with the event in the treated unit
db <- c(rep(1, 22), rep(-1, 8), rep(0, 50))
robustness_points(sens_path_pairs(db, seq(1, 3, by = 0.01), method = "binary"))
```

`sens_profile` *Run every applicable sensitivity analysis for one estimate*

Description

One call that computes the partial-R^2, impact-threshold (ITCV), and replacement (RIR) analyses for the coefficient of ‘treatment‘ in a fitted linear model, adds any further paths the user supplies (for example an E-value path from [from_evalue()] or a Rosenbaum path from [sens_path_pairs()]), and collects them in one object. The object prints as a side-by-side table ([sens_table()]) and plots as one figure with one panel per analysis. Each analysis keeps its own sensitivity parameter, its own scale, and its own decision rule; the panels do not share an axis, and the values in different rows are not to be compared or ranked.

Usage

```
sens_profile(model, treatment, alpha = 0.05, extra = NULL, n_grid = 2001)
```

Arguments

`model`	A fitted ‘lm‘ object.
`treatment`	Character string naming the treatment coefficient.
`alpha`	Two-sided significance level for all three regression analyses.
`extra`	Optional named list of further ‘confoundsens‘ paths to add to the profile.
`n_grid`	Number of grid points for each default path.

Details

The default grids are chosen from the closed-form values (robustness value, ITCV, RIR proportion), so that each crossing falls well inside the grid. They use 2,001 points, which places the interpolated crossings within about 10^{-6} of the closed-form values in typical cases.

Value

An object of class ‘confoundvis_profile‘: a list with ‘paths‘ (a named list of ‘confoundsens‘ objects), ‘table‘ (the [sens_table()] output), ‘itcv‘ (the full [itcv_lm()] result), and ‘rir‘ (the full [rir_lm()] result).

See Also

[sens_table()], [sens_report()], [plot_robustness_curve()]

Examples

```
fit <- lm(mpg ~ am + wt + hp, data = mtcars)
pr <- sens_profile(fit, "wt")
pr
plot(pr)
```

`sens_report` *Plain-language sensitivity report*

Description

Summarises a ‘confoundsens‘ path in a few sentences: the starting estimate, the framework-specific summary (robustness value, ITCV, or E-value) when available, and the confounder strengths at which the point estimate reaches the null and significance is lost. The report closes with a statement of what the analysis cannot establish.

Usage

```
sens_report(x, alpha = NULL, digits = 3)
```

Arguments

`x`	A ‘confoundsens‘ object. Paths with a stored threshold (ITCV, RIR, and Rosenbaum paths) fix the significance level when they are built, because the threshold is computed from it. For these paths ‘alpha‘ defaults to the stored level. Asking for a different level is an error: rebuild the path at the new level instead (for example ‘itcv_lm(model, treatment, alpha = 0.01)‘). Instrumental variable paths are fixed at the level of the ‘iv.sensemakr‘ object in the same way. For other paths the default is the level stored in ‘meta$default_alpha‘ ([sens_dml()] stores the two-sided level that matches the one-sided ‘doubleml‘ level), or 0.05. The level actually used is returned in column ‘alpha‘.
`alpha`	Significance level, or ‘NULL‘ (default) to use the level stored in the path, if any, and 0.05 otherwise. Two-sided, except for Rosenbaum paths, which use the one-sided level of Rosenbaum’s test.
`digits`	Number of significant digits.

Value

An object of class ‘confoundvis_report‘: a list with ‘text‘ (a character vector) and ‘points‘ (the output of [robustness_points()]). It prints as text.

Examples

```
fit <- lm(mpg ~ am + wt + hp, data = mtcars)
sens_report(sens_path_lm(fit, "am", lambda = seq(0, 0.9, by = 0.001)))
```

`sens_risk_difference` *Sensitivity of a risk difference to a binary unmeasured confounder*

Description

For a binary outcome and a binary treatment analysed with a logistic (or other binomial) regression, computes the standardized risk difference that would be obtained after adjusting for a hypothetical binary confounder U, over a range of confounder strengths. The confounder is described as in Lin, Psaty, and Kronmal (1998): its prevalence in the treated (‘p1‘) and untreated (‘p0‘) groups, conditional on the measured covariates, and its association with the outcome, either an odds ratio (the default) or a risk difference (VanderWeele and Arah, 2011). The answer is on the absolute-risk scale, so it can be compared with a practically important difference given in ‘threshold‘.

Usage

```
sens_risk_difference(
  model,
  treatment,
  p1,
  p0,
  threshold = 0,
  confounder_effect = c("odds_ratio", "risk_difference"),
  target = c("ATE", "ATT"),
  lambda = NULL,
  alpha = 0.05,
  propensity = NULL
)
```

Arguments

`model`	A fitted [stats::glm()] with ‘family = binomial‘, fitted with a ‘data‘ argument and a binary 0/1 (or two-level factor) outcome.
`treatment`	Name of the binary treatment variable in the model data (numeric 0/1, logical, or a two-level factor whose second level is the treated level).
`p1, p0`	Prevalence of the confounder in the treated and untreated groups, conditional on the measured covariates.
`threshold`	A practically important risk difference, on the scale of the estimate and with its sign (for example ‘0.02‘ for two extra events per 100 people). Default 0.
`confounder_effect`	‘"odds_ratio"‘ (default) or ‘"risk_difference"‘: the scale of the confounder-outcome association.
`target`	‘"ATE"‘ (default; standardized to everyone in the model data) or ‘"ATT"‘ (standardized to the treated).
`lambda`	Optional grid of strengths: odds ratios (>= 1) or risk differences (in \[0, 1)). By default from no confounding to 1.5 times the strength that removes the estimate (at most an odds ratio of 50).

`alpha` Two-sided significance level for the confidence limits in ‘meta‘ (default 0.05).

`propensity` Optional vector of propensity scores for the target ‘"ATE"‘, used for the distribution of U in the population. By default a logistic regression of the treatment on the other terms of the model.

For the risk-difference model, δ is the average confounder-outcome risk difference in the target population; it may vary with the covariates. The correction $RD - \delta(p_1 - p_0)$ is exact for any such pattern with average δ. A value is used only if some pattern with that average keeps every person's risks in both strata of U within $[0, 1]$; the largest such value is the average of the person-level limits, ‘meta$max_feasible‘. The default grid stops there, and infeasible scenarios in a user grid are ‘NA‘. The odds-ratio model never leaves $[0, 1]$. Offsets in the model are kept. ‘threshold‘ may lie on either side of zero; the crossing is the strength at which the adjusted risk difference reaches it.

Details

The fitted model gives each person's risk with the treatment set to each level. Under the confounder model these risks are mixtures over the two strata of U; the function solves for the stratum risks (in closed form) and averages them over the distribution of U in the target population (g-computation). With no confounding the result is the usual standardized risk difference, the same as ‘marginaleffects::avg_comparisons()‘ for the treatment. Assumptions: the prevalence of U depends on the treatment but not on the measured covariates, U does not modify the effect of treatment, and for the odds-ratio model the fitted model is the outcome model in each stratum of U up to the term for U. For the risk-difference model the adjustment is exact and equals $RD - \delta(p_1 - p_0)$.

The sensitivity parameter λ is the strength of the confounder-outcome association, oriented so that larger values move the estimate toward zero: for the odds-ratio model the odds ratio is λ or $1/\lambda$, and for the risk-difference model the risk difference is $\pm\lambda$, with the sign chosen from the signs of the estimate and of $p_1 - p_0$.

Two kinds of uncertainty are kept apart. The path over λ describes the range across confounding scenarios; it is not sampling error. Standard errors (delta method over the model coefficients, with the covariates held fixed, as in ‘marginaleffects‘) describe sampling error for each scenario. [robustness_points()] uses them for the significance crossing, and ‘meta‘ also gives the strength at which the estimate reaches ‘threshold‘ and at which the confidence limit does.

Value

A ‘confoundsens‘ object with ‘meta$framework = "risk_difference"‘. ‘theta‘ is the adjusted risk difference and ‘se‘ its standard error; ‘meta$risks‘ holds the adjusted risks in both groups; ‘meta$lambda_zero‘ and ‘meta$lambda_threshold‘ are the strengths at which the estimate reaches 0 and ‘threshold‘, and ‘meta$lambda_threshold_ci‘ the strength at which the confidence limit nearest zero reaches ‘threshold‘. ‘NA‘ means not reached.

References

Lin, D. Y., Psaty, B. M., & Kronmal, R. A. (1998). Assessing the sensitivity of regression results to unmeasured confounders in observational studies. *Biometrics*, 54(3), 948–963. doi:10.2307/2533848

VanderWeele, T. J., & Arah, O. A. (2011). Bias formulas for sensitivity analysis of unmeasured confounding for general outcomes, treatments, and confounders. *Epidemiology*, 22(1), 42–52. doi:10.1097/EDE.0b013e3181f74493

See Also

[sens_surface()] for the plane of ‘p1‘ and the confounder strength.

Examples

```
set.seed(1)
n <- 2000
x <- rnorm(n)
a <- rbinom(n, 1, plogis(0.5 * x))
y <- rbinom(n, 1, plogis(-1.5 + 0.4 * a + 0.6 * x))
fit <- glm(y ~ a + x, family = binomial, data = data.frame(y, a, x))
rd <- sens_risk_difference(fit, "a", p1 = 0.4, p0 = 0.2, threshold = 0.02)
rd
sens_report(rd)
plot_robustness_curve(rd)
```

---

sens_surface *Two-parameter sensitivity surface*

Description

Returns the two-parameter sensitivity surface of which a ‘confoundsens‘ path is one slice. The horizontal axis is always the confounder's association with treatment and the vertical axis its association with the outcome, each in the units of the framework:

Usage

```
sens_surface(x, a = NULL, b = NULL, n = 201, alpha = 0.05)
```

Arguments

x	A ‘confoundsens‘ object produced by [sens_path_lm()], [from_sensemakr()], [itcv_lm()] (its ‘$path‘), [from_konfound()] with ‘index = "IT"‘, [from_evalue()], [sens_path_pairs()], [from_iv_sensemakr()], [sens_dml()], or [sens_risk_difference()].
a, b	Optional numeric grids for the two axes. Defaults depend on the framework and extend beyond the relevant boundary.
n	Number of grid points per axis when ‘a‘ or ‘b‘ is not supplied.
alpha	Significance level for the partial-R^2 and DoubleML boundaries (for DoubleML, the default is the level stored by [sens_dml()]). ITCV, Rosenbaum, and instrumental-variable paths use the level stored in the path (a warning is given if ‘alpha‘ is supplied); for E-values the boundary is defined by the stored confidence limit.

Details

| Framework | 'a' (treatment side) | 'b' (outcome side) | Value | |—|—|—|—|—| | 'partial_r2' | $R^2_{D\sim U|X}$ | $R^2_{Y\sim U|D,X}$ | adjusted estimate | | 'itcv' | $r_{DU|X}$ | $r_{YU|X}$ | adjusted partial correlation | | 'evalue' | RR_{UD} | RR_{UY} | bound on the risk ratio | | 'rosenbaum' | Λ | Δ | implied Γ | | 'iv' | $R^2_{Z\sim W|X}$ | $R^2_{Y_0\sim W|Z,X}$ | adjusted Anderson–Rubin limit nearest the null | | 'dml' | 'cf_d' | 'cf_y' | bias-adjusted bound nearest the null | | 'risk_difference' | prevalence of U among the treated ('p0' fixed) | confounder-outcome odds ratio or risk difference | adjusted risk difference |

Each surface carries boundaries computed from the published formula of the framework, in closed form where one exists and otherwise by root finding (the partial-R^2 significance boundaries, and the Rosenbaum Γ at which the upper-bound P-value is 0.5): the **null boundary**, where the point estimate (or, for Rosenbaum, the upper-bound P-value of 0.5) is reached, and the **significance boundary**, where the conclusion at level 'alpha' is lost. For the partial-R^2 framework, the bias and adjusted standard error of Cinelli and Hazlett (2020) are used, with $d-1$ degrees of freedom for the significance boundary. For the ITCV, the boundaries are those of [plot_sensitivity_contour()]. For E-values, the joint bounding factor $B = RR_{UD}RR_{UY}/(RR_{UD} + RR_{UY} - 1)$ of Ding and VanderWeele (2016) is used, so the null boundary passes through the point (E-value, E-value). For Rosenbaum's model, $\Gamma = (\Lambda\Delta + 1)/(\Lambda + \Delta)$ (Rosenbaum and Silber, 2009), so the significance boundary is the set of (Λ, Δ) pairs that amplify the sensitivity value.

Each surface is the two-parameter display that the source framework already uses: the partial-R^2 surface is the plane of 'sensemakr::ovb_contour_plot()', the E-value surface is the curve of equivalent confounders shown by 'EValue::bias_plot()', and the Rosenbaum surface gives the pairs returned by 'sensitivitymv::amplify()'. The ITCV surface plots the adjusted partial correlation of Frank (2000) over both correlations. The instrumental-variable surface is the plane of 'iv.sensemakr::ovb_contour_plot()' (Cinelli and Hazlett, 2025), with the benchmark bounds of the 'iv.sensemakr' object; its boundaries are where the adjusted Anderson–Rubin limit nearest the null (solid) and the bound on the estimate (dashed) reach the null. The DoubleML surface is the plane of 'doubleml''s 'sensitivity_plot()' (Chernozhukov et al., 2024). The risk-difference surface varies the prevalence of the confounder among the treated with its prevalence among the untreated held at 'p0' (Lin et al., 1998); it adds the practical threshold as a boundary and uses at most 61 grid points per axis, because each point is a g-computation. The surface only evaluates these published formulas. It reports no area or share of the plane: the size of a shaded zone depends on the chosen axis ranges and has no inferential meaning.

Value

An object of class 'confoundsurf': a list with 'grid' (a data frame with columns 'a', 'b', 'value', and 'zone'), 'boundary' (a data frame with columns 'a', 'b', 'kind', and 'group'), and 'meta' (framework, axis labels, and the quantities used). Plot it with [plot_sensitivity_surface()].

References

Cinelli, C., & Hazlett, C. (2020). Making sense of sensitivity: Extending omitted variable bias. *Journal of the Royal Statistical Society: Series B*, 82(1), 39–67. doi:10.1111/rssb.12348

Ding, P., & VanderWeele, T. J. (2016). Sensitivity analysis without assumptions. *Epidemiology*, 27(3), 368–377. doi:10.1097/EDE.0000000000000457

Frank, K. A. (2000). Impact of a confounding variable on a regression coefficient. *Sociological Methods & Research*, 29(2), 147–194. doi:10.1177/0049124100029002001

Rosenbaum, P. R., & Silber, J. H. (2009). Amplification of sensitivity analysis in matched observational studies. *Journal of the American Statistical Association*, 104(488), 1398–1405. doi:10.1198/jasa.2009.tm08470

See Also

[plot_sensitivity_surface()]

Examples

```
fit <- lm(mpg ~ am + wt + hp, data = mtcars)
s <- sens_surface(sens_path_lm(fit, "wt"))
s
plot_sensitivity_surface(s)

# E-value: every pair of risk ratios that explains away the estimate
if (requireNamespace("EValue", quietly = TRUE)) {
  e <- EValue::evalues.RR(1.8, lo = 1.3, hi = 2.5)
  plot_sensitivity_surface(sens_surface(from_evalue(e)))
}
```

---

sens_table *Summarize several sensitivity paths in one table*

Description

Collects the decision information of several ‘confoundsens‘ objects into one data frame: for each path (and each level within a path), the framework, the meaning of its sensitivity parameter, the framework's own summary statistic, and the confounder strengths at which the null and the significance boundary are reached, together with the rule used to locate the latter. The table is intended for reporting several analyses of the same study side by side. It does not convert the sensitivity parameters to a common scale: values in the ‘lambda_null‘ and ‘lambda_sig‘ columns are comparable only within a row.

Usage

```
sens_table(..., alpha = NULL)
```

Arguments

...	‘confoundsens‘ objects, or a single list of them. Names (from ‘name = object‘ arguments or list names) are used as row labels.
alpha	Significance level passed to [robustness_points()] for paths without a stored threshold (default 0.05). Paths with a stored threshold (ITCV, RIR, Rosenbaum) always use the level they were built with, which is shown in column ‘alpha‘; supply ‘alpha = NULL‘ (default) when the table mixes such paths built at different levels.

Value

A data frame of class ‘confoundvis_table‘ with columns ‘path‘, ‘level‘, ‘framework‘, ‘source‘, ‘parameter‘ (meaning of lambda), ‘summary_name‘, ‘summary_value‘, ‘theta_start‘, ‘lambda_null‘, ‘lambda_sig‘, ‘sig_rule‘, and ‘alpha‘. It prints with a reminder that rows use different scales.

See Also

[robustness_points()], [sens_report()]

Examples

```
fit <- lm(mpg ~ am + wt + hp, data = mtcars)
g <- seq(0, 0.9, by = 0.001)
sens_table(partial_r2 = sens_path_lm(fit, "wt", lambda = g),
           itcv       = itcv_lm(fit, "wt", lambda = g)$path)
```

---

summary.confoundsens *Summary method for confoundsens objects*

Description

Produces a concise numerical summary of the sensitivity path, optionally broken down by level.

Usage

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

Arguments

object	A ‘confoundsens‘ object.
...	Unused; included for S3 method consistency.

Value

An object of class ‘summary.confoundsens‘ containing a ‘table‘ data.frame with per-level summary statistics.

Examples

```
x <- new_confoundsens(
  lambda = seq(0, 0.2, length.out = 10),
  theta  = seq(1, 0.6, length.out = 10),
  se     = rep(0.08, 10),
  level  = rep(c("female", "male"), length.out = 10)
)
summary(x)
```

Index

as.data.frame.confoundsens, 3
as_confoundsens, 4

confoundvis, 5
covariate_impacts, 7

from_evalue, 9
from_iv_sensemakr, 9
from_konfound, 11
from_sensemakr, 12

itcv_lm, 13

new_confoundsens, 14

plot.confoundvis_profile, 15
plot.confoundvis_result, 16
plot_robustness_curve, 17
plot_sensitivity_contour, 18
plot_sensitivity_love, 19
plot_sensitivity_surface, 20
print.confoundsens, 21

rir_lm, 22
robustness_points, 23

sens_dml, 24
sens_path_lm, 26
sens_path_pairs, 27
sens_profile, 29
sens_report, 30
sens_risk_difference, 31
sens_surface, 33
sens_table, 35
summary.confoundsens, 36