Package 'PublicationBias'

July 22, 2020

Type Package

Title Sensitivity Analysis for Publication Bias in Meta-Analyses

Version 2.2.0

Author Maya B. Mathur, Tyler J. VanderWeele

Maintainer Maya B. Mathur <mmathur@stanford.edu>

Description Performs sensitivity analysis for publication bias in meta-analyses (per Mathur & VanderWeele, 2020 [<https://osf.io/s9dp6>]). These analyses enable statements such as: ``For publication bias to shift the observed point estimate to the null, 'significant' results would need to be at least 30-fold more likely to be published than negative or 'nonsignificant' results." Comparable statements can be made regarding shifting to a chosen non-null value or shifting the confidence interval. Provides a worst-case meta-analytic point estimate under maximal publication bias obtained simply by conducting a standard meta-analysis of only the negative and ``nonsignificant'' studies.

License GPL-2

Encoding UTF-8

Imports metafor, stats, dplyr, robumeta, ggplot2, Rdpack, MetaUtility

RdMacros Rdpack

LazyData true

RoxygenNote 7.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2020-07-22 19:20:02 UTC

R topics documented:

`corrected_meta` *Estimate publication bias-corrected meta-analysis*

Description

For a chosen ratio of publication probabilities, `eta`, estimates a publication bias-corrected pooled point estimate and confidence interval per Mathur & VanderWeele (2020). Model options include fixed-effects (a.k.a. "common-effect"), robust independent, and robust clustered specifications.

Usage

```
corrected_meta(
  yi,
  vi,
  eta,
  clustervar = 1:length(yi),
  model,
  selection.tails = 1,
  favor.positive,
  alpha.select = 0.05,
  CI.level = 0.95,
  small = TRUE
)
```

Arguments

`yi`	A vector of point estimates to be meta-analyzed.
`vi`	A vector of estimated variances for the point estimates
`eta`	The number of times more likely an affirmative study is to be published than a nonaffirmative study; see Details
`clustervar`	A character, factor, or numeric vector with the same length as yi. Unique values should indicate unique clusters of point estimates. By default, assumes all point estimates are independent.
`model`	"fixed" for fixed-effects (a.k.a. "common-effect") or "robust" for robust random-effects
`selection.tails`	1 (for one-tailed selection, recommended for its conservatism) or 2 (for two-tailed selection)
`favor.positive`	`TRUE` if publication bias is assumed to favor positive estimates; `FALSE` if assumed to favor negative estimates. See Details.
`alpha.select`	Alpha-level at which publication probability is assumed to change
`CI.level`	Confidence interval level (as proportion) for the corrected point estimate. (The alpha level for inference on the corrected point estimate will be calculated from `CI.level`.)
`small`	Should inference allow for a small meta-analysis? We recommend always using TRUE.

Details

The ratio `eta` represents the number of times more likely affirmative studies (i.e., those with a "statistically significant" and positive estimate) are to be published than nonaffirmative studies (i.e., those with a "nonsignificant" or negative estimate).

If `favor.positive == FALSE`, such that publication bias is assumed to favor negative rather than positive estimates, the signs of `yi` will be reversed prior to performing analyses. The corrected estimate will be reported based on the recoded signs rather than the original sign convention, and accordingly the returned value `signs.recoded` will be `TRUE`.

Value

The function returns: the corrected pooled point estimate (`est`) potentially with its sign recoded as indicated by `signs.recoded`, inference on the bias-corrected estimate (`se`, `lo`, `hi`, `pval`), the user's specified `eta`, the number of affirmative and nonaffirmative studies after any needed recoding of signs (`k.affirmative` and `k.nonaffirmative`), and an indicator for whether the point estimates' signs were recoded (`signs.recoded`).

References

1. Mathur MB & VanderWeele TJ (2020). Sensitivity analysis for publication bias in meta-analyses. *Journal of the Royal Statistical Society, Series C.* Preprint available at https://osf.io/s9dp6/.

Examples

```
# calculate effect sizes from example dataset in metafor
require(metafor)
dat = metafor::escalc(measure="RR", ai=tpos, bi=tneg, ci=cpos, di=cneg, data=dat.bcg)

# first fit fixed-effects model without any bias correction
# since the point estimate is negative here, we'll assume publication bias favors negative
#  log-RRs rather than positive ones
rma( yi, vi, data = dat, method = "FE" )

# warmup
# note that passing eta = 1 (no publication bias) yields the naive point estimate
#  from rma above, which makes sense
corrected_meta( yi = dat$yi,
                vi = dat$vi,
                eta = 1,
                model = "fixed",
                favor.positive = FALSE )

# assume a known selection ratio of 5
# i.e., affirmative results are 5x more likely to be published
#  than nonaffirmative
corrected_meta( yi = dat$yi,
                vi = dat$vi,
                eta = 5,
                favor.positive = FALSE,
                model = "fixed" )
```

```
# same selection ratio, but now account for heterogeneity
# and clustering via robust specification
corrected_meta( yi = dat$yi,
                vi = dat$vi,
                eta = 5,
                favor.positive = FALSE,
                clustervar = dat$author,
                model = "robust" )

##### Make sensitivity plot as in Mathur & VanderWeele (2020) #####
# range of parameters to try (more dense at the very small ones)
eta.list = as.list( c( 200, 150, 100, 50, 40, 30, 20, rev( seq(1,15,1) ) ) )
res.list = lapply( eta.list, function(x) {
                   cat("\n Working on eta = ", x)
                   return( corrected_meta( yi = dat$yi,
                                           vi = dat$vi,
                                           eta = x,
                                           model = "robust",
                                           favor.positive = FALSE,
                                           clustervar = dat$author ) )
                                       }
                     )

# put results for each eta in a dataframe
res.df = as.data.frame( do.call( "rbind", res.list ) )

require(ggplot2)
ggplot( data = res.df, aes( x = eta, y = est ) ) +

  geom_ribbon( data = res.df, aes( x = eta, ymin = lo, ymax = hi ), fill = "gray" ) +

  geom_line( lwd = 1.2 ) +
  xlab( bquote( eta ) ) +
  ylab( bquote( hat(mu)[eta] ) ) +

  theme_classic()
```

`pval_plot` *Plot one-tailed p-values*

Description

Plots the one-tailed p-values. The leftmost red line indicates the cutoff for one-tailed p-values less than 0.025 (corresponding to "affirmative" studies; i.e., those with a positive point estimate and a two-tailed p-value less than 0.05). The rightmost red line indicates one-tailed p-values greater than 0.975 (i.e., studies with a negative point estimate and a two-tailed p-value less than 0.05). If there is a substantial point mass of p-values to the right of the rightmost red line, this suggests that selection may be two-tailed rather than one-tailed.

Usage

```
pval_plot(yi, vi, alpha.select = 0.05)
```

Arguments

yi	A vector of point estimates to be meta-analyzed. The signs of the estimates should be chosen such that publication bias is assumed to operate in favor of positive estimates.
vi	A vector of estimated variances for the point estimates
alpha.select	Alpha-level at which publication probability is assumed to change

References

1. Mathur MB & VanderWeele TJ (2020). Sensitivity analysis for publication bias in meta-analyses. *Journal of the Royal Statistical Society, Series C.* Preprint available at https://osf.io/s9dp6/.

Examples

```
# compute meta-analytic effect sizes
require(metafor)
dat = metafor::escalc(measure="RR", ai=tpos, bi=tneg, ci=cpos, di=cneg, data=dat.bcg)

# flip signs since we think publication bias operates in favor of negative effects
dat$yi = -dat$yi

pval_plot( yi = dat$yi,
           vi = dat$vi )
```

---

significance_funnel *Make significance funnel plot*

Description

Creates a modified funnel plot that distinguishes between affirmative and nonaffirmative studies, helping to detect the extent to which the nonaffirmative studies' point estimates are systematically smaller than the entire set of point estimates. The estimate among only nonaffirmative studies (gray diamond) represents a corrected estimate under worst-case publication bias. If the gray diamond represents a negligible effect size or if it is much smaller than the pooled estimate among all studies (black diamond), this suggests that the meta-analysis may not be robust to extreme publication bias. Numerical sensitivity analyses (via `PublicationBias::svalue`) should still be carried out for more precise quantitative conclusions.

Usage

```
significance_funnel(
  yi,
  vi,
  xmin = min(yi),
  xmax = max(yi),
  ymin = 0,
  ymax = max(sqrt(vi)),
  xlab = "Point estimate",
  ylab = "Estimated standard error",
  favor.positive = NA,
  est.all = NA,
  est.N = NA,
  alpha.select = 0.05,
  plot.pooled = TRUE
)
```

Arguments

yi	A vector of point estimates to be meta-analyzed.
vi	A vector of estimated variances for the point estimates
xmin	x-axis (point estimate) lower limit for plot
xmax	x-axis (point estimate) upper limit for plot
ymin	y-axis (standard error) lower limit for plot
ymax	y-axis (standard error) upper limit for plot
xlab	Label for x-axis (point estimate)
ylab	Label for y-axis (standard error)
favor.positive	TRUE if publication bias is assumed to favor positive estimates; FALSE if assumed to favor negative estimates.
est.all	Regular meta-analytic estimate among all studies (optional)
est.N	Worst-case meta-analytic estimate among only nonaffirmative studies (optional)
alpha.select	Alpha-level at which publication probability is assumed to change
plot.pooled	Should the pooled estimates within all studies and within only the nonaffirmative studies be plotted as well?

Details

By default (plot.pooled = TRUE), also plots the pooled point estimate within all studies, supplied by the user as est.all (black diamond), and within only the nonaffirmative studies, supplied by the user as est.N (grey diamond). The user can calculate est.all and est.N using their choice of meta-analysis model. If instead these are not supplied but plot.pooled = TRUE, these pooled estimates will be automatically calculated using a fixed-effects (a.k.a. "common-effect") model.

References

1. Mathur MB & VanderWeele TJ (2020). Sensitivity analysis for publication bias in meta-analyses. *Journal of the Royal Statistical Society, Series C.* Preprint available at https://osf.io/s9dp6/.

Examples

```
##### Make Significance Funnel with User-Specified Pooled Estimates #####

# compute meta-analytic effect sizes for an example dataset
require(metafor)
dat = metafor::escalc(measure="RR", ai=tpos, bi=tneg, ci=cpos, di=cneg, data=dat.bcg)

# flip signs since we think publication bias operates in favor of negative effects
# alternatively, if not flipping signs, could pass favor.positive = FALSE to
#  significance_funnel
dat$yi = -dat$yi

# optional: regular meta-analysis of all studies (for the black diamond)
# for flexibility, you can use any choice of meta-analysis model here
# in this case, we'll use the robust independent specification since the point estimates
#  seem to be from unique papers
# thus, each study gets its own studynum
require(robumeta)
meta.all =  robu( yi ~ 1,
                  studynum = 1:nrow(dat),
                  data = dat,
                  var.eff.size = vi,
                  small = TRUE )

# optional: calculate worst-case estimate (for the gray diamond)
#  by analyzing only the nonaffirmative studies
dat$pval = 2 * ( 1 - pnorm( abs( dat$yi / sqrt(dat$vi) ) ) )  # two-tailed p-value
dat$affirm = (dat$yi > 0) & (dat$pval < 0.05)  # is study affirmative?
meta.worst =  robu( yi ~ 1,
                    studynum = 1:nrow( dat[ dat$affirm == TRUE, ] ),
                    data = dat[ dat$affirm == TRUE, ],
                    var.eff.size = vi,
                    small = TRUE )

##### Make Significance Funnel with Alpha = 0.50 and Default Pooled Estimates #####
# change alpha to 0.50 just for illustration
# now the pooled estimates are from the fixed-effect specification because they are
#  not provided by the user
significance_funnel( yi = dat$yi,
                     vi = dat$vi,
                     favor.positive = TRUE,
                     alpha.select = 0.50,
                     plot.pooled = TRUE )
```

svalue *Severity of publication bias needed to "explain away" results*

Description

Estimates the S-value, defined as the severity of publication bias (i.e., the ratio by which affirmative studies are more likely to be published than nonaffirmative studies) that would be required to shift the pooled point estimate or its confidence interval limit to the value q.

Usage

```
svalue(
  yi,
  vi,
  q,
  clustervar = 1:length(yi),
  model,
  alpha.select = 0.05,
  eta.grid.hi = 200,
  favor.positive,
  CI.level = 0.95,
  small = TRUE,
  return.worst.meta = FALSE
)
```

Arguments

yi	A vector of point estimates to be meta-analyzed. Their signs should be coded such that publication bias is assumed to favor positive, rather than negative, estimates.
vi	A vector of estimated variances for the point estimates
q	The attenuated value to which to shift the point estimate or CI. Should be specified on the same scale as yi (e.g., if yi is on the log-RR scale, then q should be as well).
clustervar	A character, factor, or numeric vector with the same length as yi. Unique values should indicate unique clusters of point estimates. If left unspecified, assumes studies are independent.
model	"fixed" for fixed-effects (a.k.a. "common-effect") or "robust" for robust random-effects
alpha.select	Alpha-level at which publication probability is assumed to change
eta.grid.hi	The largest value of eta that should be included in the grid search. This argument is only needed when model = "robust".
favor.positive	TRUE if publication bias is assumed to favor positive estimates; FALSE if assumed to favor negative estimates. See Details.
CI.level	Confidence interval level (as a proportion) for the corrected point estimate

`small` Should inference allow for a small meta-analysis? We recommend using always using `TRUE`.

`return.worst.meta` Should the worst-case meta-analysis of only the nonaffirmative studies be returned?

Details

To illustrate interpretation of the S-value, if the S-value for the point estimate is 30 with `q=0`, this indicates that affirmative studies (i.e., those with a "statistically significant" and positive estimate) would need to be 30-fold more likely to be published than nonaffirmative studies (i.e., those with a "nonsignificant" or negative estimate) to attenuate the pooled point estimate to `q`.

If `favor.positive == FALSE`, such that publication bias is assumed to favor negative rather than positive estimates, the signs of `yi` will be reversed prior to performing analyses. The returned number of affirmative and nonaffirmative studies will reflect the recoded signs, and accordingly the returned value `signs.recoded` will be `TRUE`.

Value

The function returns: the amount of publication bias required to attenuate the pooled point estimate to `q` (`sval.est`), the amount of publication bias required to attenuate the confidence interval limit of the pooled point estimate to `q` (`sval.ci`), the number of affirmative and nonaffirmative studies after any needed recoding of signs (`k.affirmative` and `k.nonaffirmative`), and an indicator for whether the point estimates' signs were recoded (`signs.recoded`).

If `return.worst.meta = TRUE`, also returns the worst-case meta-analysis of only the nonaffirmative studies. If `model = "fixed"`, the worst-case meta-analysis is fit by `metafor::rma.uni`. If `model = "robust"`, it is fit by `robumeta::robu`. Note that in the latter case, custom inverse-variance weights are used, which are the inverse of the sum of the study's variance and a heterogeneity estimate from a naive random-effects meta-analysis (Mathur & VanderWeele, 2020). This is done for consistency with the results of `corrected_meta`, which is used to determine `sval.est` and `sval.ci`. Therefore, the worst-case meta-analysis results may differ slightly from what you would obtain if you simply fit `robumeta::robu` on the nonaffirmative studies with the default weights.

References

1. Mathur MB & VanderWeele TJ (2020). Sensitivity analysis for publication bias in meta-analyses. *Journal of the Royal Statistical Society, Series C.* Preprint available at https://osf.io/s9dp6/.

Examples

```
# calculate effect sizes from example dataset in metafor
require(metafor)
dat = metafor::escalc(measure="RR", ai=tpos, bi=tneg, ci=cpos, di=cneg, data=dat.bcg)

##### Fixed-Effects Specification #####
# S-values and worst-case meta-analysis under fixed-effects specification
svals.FE.0 = svalue( yi = dat$yi,
                     vi = dat$vi,
                     q = 0,
```

```
                  favor.positive = FALSE,
                  model = "fixed" )

# publication bias required to shift point estimate to 0
svals.FE.0$sval.est

# and to shift CI to include 0
svals.FE.0$sval.ci

# now try shifting to a nonzero value (RR = 0.90)
svals.FE.q = svalue( yi = dat$yi,
                     vi = dat$vi,
                     q = log(.9),
                     favor.positive = FALSE,
                     model = "fixed" )

# publication bias required to shift point estimate to RR = 0.90
svals.FE.q$sval.est

# and to shift CI to RR = 0.90
svals.FE.q$sval.ci

##### Robust Clustered Specification #####
svalue( yi = dat$yi,
        vi = dat$vi,
        q = 0,
        favor.positive = FALSE,
        model = "robust" )
```

Index