Package ‘MetaUtility’

November 28, 2019

Type Package
Title Utility Functions for Conducting and Interpreting Meta-Analyses
Version 2.1.0
Author Maya B. Mathur, Rui Wang, Tyler J. VanderWeele
Maintainer Maya B. Mathur <mmathur@stanford.edu>
Description Contains functions to estimate the proportion of effects stronger than a threshold of scientific importance (function prop_stronger), to nonparametrically characterize the distribution of effects in a meta-analysis (calib_est, pct_pval), to make effect size conversions (r_to_d, r_to_z, z_to_r), to compute and format inference in a meta-analysis (format_CI, format_stat, tau_CI), to scrape results from existing meta-analyses for re-analysis (scrape_meta, parse_CI_string).
License GPL-2
Encoding UTF-8
Imports metafor, stats, stringr, purrr, dplyr
LazyData true
RoxygenNote 6.1.1
NeedsCompilation no
Repository CRAN
Date/Publication 2019-11-28 15:20:03 UTC

R topics documented:

  calib_est     ....................................................... 2
  format_CI    .......................................................... 3
  format_stat  ............................................................ 3
  parse_CI_string ....................................................... 4
  pct_pval      ........................................................... 4
  prop_stronger .......................................................... 5
  prop_stronger_sign .................................................... 9
  round2        ............................................................ 10
  r_to_d        ........................................................... 11
calib_est

Description

Returns estimates of the true effect in each study based on the methods of Wang & Lee (2019). Unlike the point estimates themselves, these "calibrated" estimates have been appropriately shrunk to correct the overdispersion that arises due to the studies’ finite sample sizes. By default, this function uses Dersimonian-Laird moments-based estimates of the mean and variance of the true effects, as Wang & Lee (2019) recommended.

Usage

calib_est(yi, sei, method = "DL")

Arguments

yi  Vector of study-level point estimates
sei  Vector of study-level standard errors
method  Estimation method for mean and variance of true effects (passed to metafor::rma.uni)

References


Examples

d = metafor::escalc(measure="RR", ai=tpos, bi=tneg,
               ci=cpos, di=cneg, data=metafor::dat.bcg)

# calculate calibrated estimates
d$calib = calib_est(yi = d$yi,
                    sei = sqrt(d$vi) )

# look at 5 studies with the largest calibrated estimates
d = d[ order(d$calib, decreasing = TRUE), ]
dtrial[1:5]

# look at kernel density estimate of calibrated estimates
plot(density(d$calib))
format_CI

Manuscript-friendly confidence interval formatting

Description

Formats confidence interval lower and upper bounds into a rounded string.

Usage

format_CI(lo, hi, digits = 2)

Arguments

lo                   Confidence interval lower limit (numeric)
hi                   Confidence interval upper limit (numeric)
digits               Digits for rounding

Examples

format_CI(0.36, 0.72, 3)

format_stat

Manuscript-friendly number formatting

Description

Formats a numeric result (e.g., p-value) as a manuscript-friendly string in which values below a minimum cutoff (e.g., $10^{-5}$) are reported for example as "$< 10^{-5}$", values between the minimum cutoff and a maximum cutoff (e.g., 0.01) are reported in scientific notation, and p-values above the maximum cutoff are reported simply as, for example, 0.72.

Usage

format_stat(x, digits = 2, cutoffs = c(0.01, 10^{-5}))

Arguments

x                   Numeric value to format
digits              Digits for rounding
cutoffs             A vector containing the two cutoffs

Examples

format_stat(0.735253)
format_stat(0.735253, digits = 4)
format_stat(0.0123)
format_stat(0.0001626)
format_stat(0.0001626, cutoffs = c(0.01, 10^{-3}))
parse_CI_string  

Parse a string with point estimate and confidence interval

**Description**

Given a vector of strings such as "0.65 (0.6, 0.70)", for example obtained by running optical character recognition (OCR) software on a screenshot of a published forest plot, parses the strings into a dataframe of point estimates and upper confidence interval limits. Assumes that the point estimate occurs before an opening bracket of the form "(" or "[") and that the confidence interval upper limit follows a the character sep (by default a comma, but might be a hyphen, for example). To further parse this dataframe into point estimates and variances, see MetaUtility::scrape_meta.

**Usage**

```r
parse_CI_string(string, sep = ",")
```

**Arguments**

- **string**: A vector of strings to be parsed.
- **sep**: The character (not including whitespaces) separating the lower from the upper limits.

**Examples**

```r
# messy string of confidence intervals
mystring = c("0.65 [0.6, 0.7]", "0.8(0.5, 0.9]", "1.2 [0.3, 1.5]"
parse_CI_string(mystring)

# now with a hyphen separator
mystring = c("0.65 [0.6- 0.7]", "0.8(0.5 - 0.9]", "1.2 [0.3 -1.5]"
parse_CI_string(mystring, sep="-")
```

pct_pval  

Return sign test p-value for meta-analysis percentile

**Description**

Returns a p-value for testing the hypothesis that $\mu$ is the $pct^{th}$ percentile of the true effect distribution based on the nonparametric sign test method of Wang et al. (2010). This function is also called by prop_stronger when using the sign test method.

**Usage**

```r
pct_pval(yi, sei, mu, pct, R = 2000)
```
prop_stronger

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>yi</td>
<td>Vector of study-level point estimates</td>
</tr>
<tr>
<td>sei</td>
<td>Vector of study-level standard errors</td>
</tr>
<tr>
<td>mu</td>
<td>The effect size to test as the pct(^{th}) percentile</td>
</tr>
<tr>
<td>pct</td>
<td>The percentile of interest (e.g., 0.50 for the median)</td>
</tr>
<tr>
<td>R</td>
<td>Number of simulation iterates to use when estimating null distribution of the test statistic.</td>
</tr>
</tbody>
</table>

References


Examples

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

# test H0: the median is -0.3
# using only R = 100 for speed, but should be much larger (e.g., 2000) in practice
pct_pval( yi = d$yi,
          sei = sqrt(d$vi),
          mu = -0.3,
          pct = 0.5,
          R = 100 )
```

---

**prop_stronger**

*Estimate proportion of true effect sizes above or below a threshold*

Description

Estimates the proportion of true (i.e., population parameter) effect sizes in a meta-analysis that are above or below a specified threshold of scientific importance based on the methods of Mathur & VanderWeele (2018) and Mathur & VanderWeele (2020).

Usage

```r
prop_stronger(q, M = NA, t2 = NA, se.M = NA, se.t2 = NA,
               ci.level = 0.95, tail = NA, estimate.method = "calibrated",
               ci.method = "calibrated", calib.est.method = "DL", dat = NULL,
               R = 2000, bootstrap = "ifneeded", yi.name = "yi", vi.name = "vi")
```
prop_stronger

Arguments

q  True effect size that is the threshold for "scientific importance"
M  Pooled point estimate from meta-analysis (required only for parametric estimation/inference and for Shapiro p-value)
t2  Estimated heterogeneity (\(\tau^2\)) from meta-analysis (required only for parametric estimation/inference and for Shapiro p-value)
se.M  Estimated standard error of pooled point estimate from meta-analysis (required only for parametric inference)
se.t2  Estimated standard error of \(\tau^2\) from meta-analysis (required only for parametric inference)
ci.level  Confidence level as a proportion (e.g., 0.95 for a 95% confidence interval)
tail  "above" for the proportion of effects above q; "below" for the proportion of effects below q.
estimate.method  Method for point estimation of the proportion ("calibrated" or "parametric"). See Details.
ci.method  Method for confidence interval estimation ("calibrated", "parametric", or "sign.test"). See Details.
calib.est.method  Method for estimating the mean and variance of the true effects when computing calibrated estimates. See Details.
dat  Dataset of point estimates (with names equal to the passed yi.name) and their variances (with names equal to the passed vi.name). Not required if using ci.method = "parametric" and bootstrapping is not needed.
R  Number of bootstrap or simulation iterates (depending on the methods chosen). Not required if using ci.method = "parametric" and bootstrapping is not needed.
bootstrap  Only used when ci.method = "parametric". In that case, if bootstrap = "ifneeded", bootstraps if estimated proportion is less than 0.15 or more than 0.85. If equal to "never", instead does not return inference in the above edge cases.
yi.name  Name of the variable in dat containing the study-level point estimates. Used for bootstrapping and conducting Shapiro test.
vi.name  Name of the variable in dat containing the study-level variances. Used for bootstrapping and conducting Shapiro test.

Details

These methods perform well only in meta-analyses with at least 10 studies; we do not recommend reporting them in smaller meta-analyses. By default, prop_stronger performs estimation using a "calibrated" method (Mathur & VanderWeele, 2020) that extends work by Wang et al. (2019). This method makes no assumptions about the distribution of true effects and performs well in meta-analyses with as few as 10 studies. Calculating the calibrated estimates involves first estimating the meta-analytic mean and variance, which, by default, is done using the moments-based
Dersimonian-Laird estimator as in Wang et al. (2019). To use a different method, which will be passed to `metafor::rma.uni`, change the argument `calib.est.method` based on the documentation for `metafor::rma.uni`. For inference, the calibrated method uses bias-corrected and accelerated bootstrapping. The bootstrapping may fail to converge for some small meta-analyses for which the threshold is distant from the mean of the true effects. In these cases, you can try choosing a threshold closer to the pooled point estimate of your meta-analysis. The mean of the bootstrap estimates of the proportion is returned as a diagnostic for potential bias in the estimated proportion.

The parametric method assumes that the true effects are approximately normal and that the number of studies is large. When these conditions hold and the proportion being estimated is not extreme (between 0.15 and 0.85), the parametric method may be more precise than the calibrated method. to improve precision. When using the parametric method and the estimated proportion is less than 0.15 or more than 0.85, it is best to bootstrap the confidence interval using the bias-corrected and accelerated (BCa) method (Mathur & VanderWeele, 2018); this is the default behavior of `prop_stronger`. Sometimes BCa confidence interval estimation fails, in which case `prop_stronger` instead uses the percentile method, issuing a warning if this is the case (but note that the percentile method should not be used when bootstrapping the calibrated estimates rather than the parametric estimates). We use a modified "safe" version of the boot package code for bootstrapping such that if any bootstrap iterates fail (usually because of model estimation problems), the error message is printed but the bootstrap iterate is simply discarded so that confidence interval estimation can proceed. As above, the mean of the bootstrapped estimates of the proportion is returned as a diagnostic for potential bias in the estimated proportion.

The sign test method (Mathur & VanderWeele, 2020) is an extension of work by Wang et al. (2010). This method was included in Mathur & VanderWeele’s (2020) simulation study; it performed adequately when there was high heterogeneity, but did not perform well with lower heterogeneity. However, in the absence of a clear criterion for how much heterogeneity is enough for the method to perform well, we do not in general recommend its use. Additionally, this method requires effects that are reasonably symmetric and unimodal.

Value

Returns a dataframe containing the point estimate for the proportion (`est`), its estimated standard error (`se`), lower and upper confidence interval limits (`lo` and `hi`), and, depending on the user's specifications, the mean of the bootstrap estimates of the proportion (`bt.mn`) and the p-value for a Shapiro test for normality conducted on the standardized point estimates (`shapiro.pval`).

References


```
Examples

##### Example 1: BCG Vaccine and Tuberculosis Meta-Analysis #####

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

# fit random-effects model
# note that metafor package returns on the log scale
m = metafor::rma.uni(yi=d$yi, vi=d$vi, knha=TRUE,
                     measure="RR", method="REML")

# pooled point estimate (RR scale)
exp(m$b)

# estimate the proportion of effects stronger than RR = 0.70
# as recommended, use the calibrated approach for both point estimation and CI
# bootstrap reps should be higher in practice (e.g., 1000)
# here using fewer for speed
prop_stronger(q = log(0.7),
              tail = "below",
              estimate.method = "calibrated",
              ci.method = "calibrated",
              dat = d,
              yi.name = "yi",
              vi.name = "vi",
              R = 100)

# warning goes away with more bootstrap iterates
# no Shapiro p-value because we haven't provided the dataset and its variable names

# now use the parametric approach (Mathur & VanderWeele 2018)
# no bootstrapping will be needed for this choice of q
prop_stronger(q = log(0.7),
              M = as.numeric(m$b),
              t2 = m$tau2,
              se.M = as.numeric(m$vb),
              se.t2 = m$se.tau2,
              tail = "below",
              estimate.method = "parametric",
              ci.method = "parametric",
              bootstrap = "ifneeded")

##### Example 2: Meta-Analysis of Multisite Replication Studies #####

# replication estimates (Fisher's z scale) and SEs
# from moral credential example in reference #2
r.fis = c(0.303, 0.078, 0.113, -0.055, 0.056, 0.073,
          0.263, 0.056, 0.002, -0.106, 0.09, 0.024, 0.069, 0.074,
          0.107, 0.01, -0.089, -0.187, 0.265, 0.076, 0.082)

r.SE = c(0.111, 0.092, 0.156, 0.106, 0.105, 0.057,
prop_stronger_sign

\[0.091, 0.089, 0.081, 0.1, 0.093, 0.086, 0.076,
0.094, 0.065, 0.087, 0.108, 0.114, 0.073, 0.105, 0.04)\]

\[
d = \text{data.frame(} \ yi = \text{r.fis,} \\
\quad \ vi = \text{r.SE^2 })
\]

# meta-analyze the replications
\[m = \text{metafor::rma.uni(} \ yi = \text{r.fis,} \ vi = \text{r.SE^2, measure = "ZCOR")}\]

# probability of true effect above \(r = 0.10 = 28\%\)
# convert threshold on \(r\) scale to Fisher's \(z\)
\[q = \text{r_to_z(}0.10)\]

# bootstrap reps should be higher in practice (e.g., 1000)
# here using only 100 for speed
\[\text{prop_stronger(} q = \text{q,} \]
\quad \text{tail = "above",} \\
\quad \text{estimate.method = "calibrated",} \\
\quad \text{ci.method = "calibrated",} \\
\quad \text{dat = d,} \\
\quad \text{yi.name = "yi",} \\
\quad \text{vi.name = "vi",} \\
\quad \text{R = 100 )}\]

# probability of true effect equally strong in opposite direction
\[q.star = \text{r_to_z(-}0.10)\]
\[\text{prop_stronger(} q = \text{q.star,} \]
\quad \text{tail = "below",} \\
\quad \text{estimate.method = "calibrated",} \\
\quad \text{ci.method = "calibrated",} \\
\quad \text{dat = d,} \\
\quad \text{yi.name = "yi",} \\
\quad \text{vi.name = "vi",} \\
\quad \text{R = 100 )}\]

# BCa fails to converge here

---

**prop_stronger_sign**  
Return sign test point estimate of proportion of effects above or below threshold.

**Description**

Internal function not intended for user to call. Uses an extension of the sign test method of Wang et al. (2010) to estimate the proportion of true (i.e., population parameter) effect sizes in a meta-analysis that are above or below a specified threshold of scientific importance. See important caveats in the Details section of the documentation for the function `prop_stronger`. 
Usage

prop_stronger_sign(q, yi, vi, ci.level = 0.95, tail = NA, R = 2000,
return.vectors = FALSE)

Arguments

q True effect size that is the threshold for "scientific importance"

yi Study-level point estimates

vi study-level variances

ci.level Confidence level as a proportion

tail above for the proportion of effects above q; below for the proportion of effects below q.

R Number of simulation iterates to estimate null distribution of sign test statistic

return.vectors Should all percents and p-values from the grid search be returned?

References


---

round2 Round while keeping trailing zeroes

Description

Rounds a numeric value and formats it as a string, keeping trailing zeroes.

Usage

round2(x, digits = 2)

Arguments

x Numeric value to round

digits Digits for rounding

Examples

round2(0.03000, digits = 4)

# compare to base round, which drops trailing zeroes and returns a numeric
round(0.03000, digits = 4)
**r_to_d**  
*Convert Pearson’s r to Cohen’s d*

**Description**

Converts Pearson’s r (computed with a continuous X and Y) to Cohen’s d for use in meta-analysis. The resulting Cohen’s d represents the estimated increase in standardized Y that is associated with a delta-unit increase in X.

**Usage**

\[
\text{r_to_d}(r, \text{sx}, \text{delta}, N = \text{NA}, Ns = N, \text{sx.known} = \text{FALSE})
\]

**Arguments**

- **r**: Pearson’s correlation
- **sx**: Sample standard deviation of X
- **delta**: Contrast in X for which to compute Cohen’s d, specified in raw units of X (not standard deviations).
- **N**: Sample size used to estimate r
- **Ns**: Sample size used to estimate sx, if different from N
- **sx.known**: Is sx known rather than estimated? (By default, assumes sx is estimated, which will almost always be the case.)

**Details**

To preserve the sign of the effect size, the code takes the absolute value of delta. The standard error estimate assumes that X is approximately normal and that N is large.

**References**


**Examples**

# d for a 1-unit vs. a 2-unit increase in X

\[
\begin{align*}
\text{r_to_d}&( r = 0.5, \\
&\quad \text{sx} = 2, \\
&\quad \text{delta} = 1, \\
&\quad N = 100 )
\end{align*}
\]

\[
\begin{align*}
\text{r_to_d}&( r = 0.5, \\
&\quad \text{sx} = 2, \\
&\quad \text{delta} = 2, \\
&\quad N = 100 )
\end{align*}
\]

# d when sx is estimated in the same vs. a smaller sample
# point estimate will be the same, but inference will be a little
# less precise in second case
r_to_d( r = -0.3,
    sx = 2,
    delta = 2,
    N = 300,
    Ns = 300 )

r_to_d( r = -0.3,
    sx = 2,
    delta = 2,
    N = 300,
    Ns = 30 )

---

**r_to_z**

*Convert Pearson’s r to Fisher’s z*

**Description**

Converts Pearson’s r to Fisher’s z for use in meta-analysis.

**Usage**

```r
r_to_z(r)
```

**Arguments**

- `r` Pearson’s correlation

**Examples**

```r
# convert a Pearson correlation of -0.8 to Fisher's z
r_to_z(-0.8)
```

---

**scrape_meta**

*Convert forest plot or summary table to meta-analytic dataset*

**Description**

Given relative risks (RR) and upper bounds of 95% confidence intervals (CI) from a forest plot or summary table, returns a dataframe ready for meta-analysis (e.g., via the metafor package) with the log-RRs and their variances. Optionally, the user may indicate studies for which the point estimate is to be interpreted as an odds ratios of a common outcome rather than a relative risk; for such studies, the function applies VanderWeele (2017)’s square-root transformation to convert the odds ratio to an approximate risk ratio.
Usage

```r
scrape_meta(type = "RR", est, hi, sqrt = FALSE)
```

Arguments

type

RR if point estimates are RRs or ORs (to be handled on log scale); raw if point estimates are raw differences, standardized mean differences, etc. (such that they can be handled with no transformations)
est

Vector of study point estimates on RR or OR scale
hi

Vector of upper bounds of 95% CIs on RRs
sqrt

Vector of booleans (TRUE/FALSE) for whether each study measured an odds ratio of a common outcome that should be approximated as a risk ratio via the square-root transformation

References


---

### tau_CI

**Return confidence interval for tau for a meta-analysis**

**Description**

Returns confidence interval lower and upper limits for tau (the estimated standard deviation of the true effects) for a meta-analysis fit in `metafor::rma`.

**Usage**

```r
tau_CI(meta, ci.level = 0.95)
```

Arguments

- **meta**: A meta-analysis object fit in `metafor::rma`
- **ci.level**: Confidence interval level as a proportion (e.g., 0.95)

**Examples**

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

# fit random-effects model
# note that metafor package returns on the log scale
m = metafor::rma.uni(yi = d$yi, vi = d$vi, knha=TRUE,
                     measure="RR", method="REML")
```


z_to_r

Description

Converts Fisher's z to Pearson's r for use in meta-analysis.

Usage

z_to_r(z)

Arguments

z

Fisher's z

Examples

# convert Fisher's z of 1.1 to Pearson's r
z_to_r(1.1)
Index

calib_est, 2
format_CI, 3
format_stat, 3
parse_CI_string, 4
pct_pval, 4
prop_stronger, 5
prop_stronger_sign, 9
r_to_d, 11
r_to_z, 12
round2, 10
scrape_meta, 12
tau_CI, 13
z_to_r, 14