Package ‘selectMeta’

October 14, 2022

Type Package

Title Estimation of Weight Functions in Meta Analysis

Version 1.0.8

Date 2015-07-03

Author Kaspar Rufibach <kaspar.rufibach@gmail.com>

Maintainer Kaspar Rufibach <kaspar.rufibach@gmail.com>

Depends DEoptim (>= 2.0-6)

Imports graphics, stats

Description Publication bias, the fact that studies identified for inclusion in a meta analysis do not represent all studies on the topic of interest, is commonly recognized as a threat to the validity of the results of a meta analysis. One way to explicitly model publication bias is via selection models or weighted probability distributions. In this package we provide implementations of several parametric and nonparametric weight functions. The novelty in Rufibach (2011) is the proposal of a non-increasing variant of the nonparametric weight function of Dear & Begg (1992). The new approach potentially offers more insight in the selection process than other methods, but is more flexible than parametric approaches. To maximize the log-likelihood function proposed by Dear & Begg (1992) under a monotonicity constraint we use a differential evolution algorithm proposed by Ardia et al (2010a, b) and implemented in Mullen et al (2009). In addition, we offer a method to compute a confidence interval for the overall effect size theta, adjusted for selection bias as well as a function that computes the simulation-based p-value to assess the null hypothesis of no selection as described in Rufibach (2011, Section 6).

URL http://www.kasparrufibach.ch

NeedsCompilation no

Repository CRAN

Date/Publication 2015-07-03 12:51:10

R topics documented:

`selectMeta-package` *Estimation of Weight Functions in Meta Analysis*

Description

Publication bias, the fact that studies identified for inclusion in a meta analysis do not represent all studies on the topic of interest, is commonly recognized as a threat to the validity of the results of a meta analysis. One way to explicitly model publication bias is via selection models or weighted probability distributions. For details we refer to Iyengar & Greenhouse (1998), Dear & Begg (1992), and Rufibach (2011). In this package we provide implementations of all the weight functions proposed in these papers. The novelty in Rufibach (2011) is the proposal of a non-increasing variant of the nonparametric weight function of Dear & Begg (1992). Since virtually all parametric weight functions proposed so far in the literature are in fact decreasing and only few studies are included in a typical meta analysis regularization by imposing monotonicity seems a sensible approach. The new approach potentially offers more insight in the selection process than other methods, but is more flexible than parametric approaches. To maximize the log-likelihood function proposed by Dear & Begg (1992) under a monotonicity constraint on w we use a differential evolution algorithm proposed by Ardia et al (2010a, b) and implemented in Mullen et al (2009).

The main functions in this package are `IyenGreen` and `DearBegg`. Using `DearBeggMonotoneCItheta` one can compute a profile likelihood confidence interval for the overall effect size θ and using `DearBeggMonotonePvalSelection` the simulation-based p-value to assess the null hypothesis of no selection, as described in Rufibach (2011, Section 6), can be computed. In addition, we provide two datasets: `education`, a dataset frequently used in illustration of meta analysis and `passive_smoking`, a second dataset that has caused some controversy about whether publication bias is present in this dataset or not.

Details

Package:	selectMeta
Type:	Package
Version:	1.0.8
Date:	2015-07-03
License:	GPL (>=2)

Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

References

Ardia, D., Boudt, K., Carl, P., Mullen, K.M., Peterson, B.G. (2010). Differential Evolution ('DEoptim') for Non-Convex Portfolio Optimization.

Ardia, D., Mullen, K.M., et.al. (2010). The 'DEoptim' Package: Differential Evolution Optimization in 'R'. Version 2.0-7.

Dear, K.B.G. and Begg, C.B. (1992). An Approach for Assessing Publication Bias Prior to Performing a Meta-Analysis. *Statist. Sci.*, **7(2)**, 237–245.

Hedges, L. and Olkin, I. (1985). Statistical Methods for Meta-Analysis. Academic, Orlando, Florida.

Iyengar, S. and Greenhouse, J.B. (1988). Selection models and the file drawer problem. *Statist. Sci.*, **3**, 109–135.

Mullen, K.M., Ardia, D., Gil, D.L., Windover, D., Cline, J. (2009). 'DEoptim': An 'R' Package for Global Optimization by Differential Evolution.

Rufibach, K. (2011). Selection Models with Monotone Weight Functions in Meta-Analysis. *Biom. J.*, **53**(4), 689–704.

Examples

```
# All functions in this package are illustrated
# in the help file for the function DearBegg().
```

DearBegg	*Compute the nonparametric weight function from Dear and Begg (1992)*

Description

In Dear and Begg (1992) it was proposed to nonparametrically estimate via maximum likelihood the weight function w in a selection model via pooling p-values in groups of 2 and assuming a piecewise constant w. The function `DearBegg` implements estimation of w via a coordinatewise Newton-Raphson algorithm as described in Dear and Begg (1992). In addition, the function `DearBeggMonotone` enables computation of the weight function in the same model under the constraint that it is non-increasing, see Rufibach (2011). To this end we use the differential evolution algorithm described in Ardia et al (2010a, b) and implemented in Mullen et al (2009). The functions `Hij`, `DearBeggLoglik`, and `DearBeggToMinimize` are not intended to be called by the user.

Usage

```
DearBegg(y, u, lam = 2, tolerance = 10^-10, maxiter = 1000,
    trace = TRUE)
DearBeggMonotone(y, u, lam = 2, maxiter = 1000, CR = 0.9,
    NP = NA, trace = TRUE)
Hij(theta, sigma, y, u, teststat)
DearBeggLoglik(w, theta, sigma, y, u, hij, lam)
DearBeggToMinimize(vec, y, u, lam)
```

Arguments

`y`	Normally distributed effect sizes.
`u`	Associated standard errors.
`lam`	Weight of the first entry of w in the likelihood function. Dear and Begg (1992) recommend to use `lam` = 2.
`tolerance`	Stopping criterion for Newton-Raphson.
`maxiter`	Maximal number of iterations for Newton-Raphson.
`trace`	If `TRUE`, progress of the algorithm is shown.
`CR`	Parameter that is given to `DEoptim`. See the help file of the function `DEoptim.control` for details.
`NP`	Parameter that is given to `DEoptim`. See the help file of the function `DEoptim.control` for details.
`w`	Weight function, parametrized as vector of length $1 + \lfloor (n/2) \rfloor$ where n is the number of studies, i.e. the length of y.
`theta`	Effect size estimate.
`sigma`	Random effects variance component.
`hij`	Integral of density over a constant piece of w. See Rufibach (2011, Appendix) for details.
`vec`	Vector of parameters over which we maximize.
`teststat`	Vector of test statistics, equals $|y|/u$.

Value

A list consisting of the following elements:

`w`	Vector of estimated weights.
`theta`	Estimate of the combined effect in the Dear and Begg model.
`sigma`	Estimate of the random effects component variance.
`p`	p-values computed from the inputed test statistics, ordered in decreasing order.
`y`	Effect sizes, ordered in decreasing order of p-values.
`u`	Standard errors, ordered in decreasing order of p-values.
`loglik`	Value of the log-likelihood at the maximum.
`DEoptim.res`	Only available in `DearBeggMonotone`. Provides the object that is outputted by `DEoptim`.

Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

References

Ardia, D., Boudt, K., Carl, P., Mullen, K.M., Peterson, B.G. (2010). Differential Evolution ('DEoptim') for Non-Convex Portfolio Optimization.

Ardia, D., Mullen, K.M., et.al. (2010). The 'DEoptim' Package: Differential Evolution Optimization in 'R'. Version 2.0-7.

Dear, K.B.G. and Begg, C.B. (1992). An Approach for Assessing Publication Bias Prior to Performing a Meta-Analysis. *Statist. Sci.*, **7(2)**, 237–245.

Mullen, K.M., Ardia, D., Gil, D.L., Windover, D., Cline, J. (2009). 'DEoptim': An 'R' Package for Global Optimization by Differential Evolution.

Rufibach, K. (2011). Selection Models with Monotone Weight Functions in Meta-Analysis. *Biom. J.*, **53**(4), 689–704.

See Also

IyenGreen for a parametric selection model.

Examples

```
## Not run:
##------------------------------------------
## Analysis of Hedges & Olkin dataset
## re-analyzed in Iyengar & Greenhouse, Dear & Begg
##------------------------------------------
data(education)
t  <- education$t
q  <- education$q
N  <- education$N
y  <- education$theta
u  <- sqrt(2 / N)
n  <- length(y)
k  <- 1 + floor(n / 2)
lam1 <- 2

## compute p-values
p <- 2 * pnorm(-abs(t))


##------------------------------------------
## compute all weight functions available
## in this package
##------------------------------------------

## weight functions from Iyengar & Greenhouse (1988)
res1 <- IyenGreenMLE(t, q, N, type = 1)
res2 <- IyenGreenMLE(t, q, N, type = 2)
```

```
## weight function from Dear & Begg (1992)
res3 <- DearBegg(y, u, lam = lam1)

## monotone version of Dear & Begg, as introduced in Rufibach (2011)
set.seed(1977)
res4 <- DearBeggMonotone(y, u, lam = lam1, maxiter = 1000, CR = 1)

## plot
plot(0, 0, type = "n", xlim = c(0, 1), ylim = c(0, 1), xlab = "p-values",
    ylab = "estimated weight function")
ps <- seq(0, 1, by = 0.01)
rug(p, lwd = 3)
lines(ps, IyenGreenWeight(-qnorm(ps / 2), b = res1$beta, q = 50,
    type = 1, alpha = 0.05), lwd = 3, col = 2)
lines(ps, IyenGreenWeight(-qnorm(ps / 2), b = res2$beta, q = 50,
    type = 2, alpha = 0.05), lwd = 3, col = 4)
weightLine(p, w = res3$w, col0 = 3, lwd0 = 3, lty0 = 2)
weightLine(p, w = res4$w, col0 = 6, lwd0 = 2, lty0 = 1)

legend("topright", c(expression("Iyengar & Greenhouse (1988) w"[1]),
    expression("Iyengar & Greenhouse (1988) w"[2]), "Dear and Begg (1992)",
    "Rufibach (2011)"), col = c(2, 4, 3, 6), lty = c(1, 1, 2, 1),
    lwd = c(3, 3, 3, 2), bty = "n")

## compute selection bias
eta <- sqrt(res4$sigma ^ 2 + res4$u ^ 2)
bias <- effectBias(res4$y, res4$u, res4$w, res4$theta, eta)
bias


##-------------------------------------------
## Compute p-value to assess null hypothesis of no selection,
## as described in Rufibach (2011, Section 6)
## We use the package 'meta' to compute initial estimates for
## theta and sigma
##-------------------------------------------
library(meta)

## compute null parameters
meta.edu <- metagen(TE = y, seTE = u, sm = "MD", level = 0.95,
    comb.fixed = TRUE, comb.random = TRUE)
theta0 <- meta.edu$TE.random
sigma0 <- meta.edu$tau

M <- 1000
res <- DearBeggMonotonePvalSelection(y, u, theta0, sigma0, lam = lam1,
    M = M, maxiter = 1000)

## plot all the computed monotone functions
plot(0, 0, xlim = c(0, 1), ylim = c(0, 1), type = "n", xlab = "p-values",
    ylab = expression(w(p)))
abline(v = 0.05, lty = 3)
```

```
for (i in 1:M){weightLine(p, w = res$res.mono[1:k, i], col0 = grey(0.8),
    lwd0 = 1, lty0 = 1)}
rug(p, lwd = 2)
weightLine(p, w = res$mono0, col0 = 2, lwd0 = 1, lty0 = 1)


## ========================================================================


##------------------------------------------
## Analysis second-hand tobacco smoke dataset
## Rothstein et al (2005), Publication Bias in Meta-Analysis, Appendix A
##------------------------------------------
data(passive_smoking)
u <- passive_smoking$selnRR
y <- passive_smoking$lnRR
n <- length(y)
k <- 1 + floor(n / 2)
lam1 <- 2

res2 <- DearBegg(y, u, lam = lam1)
set.seed(1)
res3 <- DearBeggMonotone(y = y, u = u, lam = lam1, maxiter = 2000, CR = 1)

plot(0, 0, type = "n", xlim = c(0, 1), ylim = c(0, 1), pch = 19, col = 1,
    xlab = "p-values", ylab = "estimated weight function")
weightLine(rev(sort(res2$p)), w = res2$w, col0 = 2, lwd0 = 3, lty0 = 2)
weightLine(rev(sort(res3$p)), w = res3$w, col0 = 4, lwd0 = 2, lty0 = 1)

legend("bottomright", c("Dear and Begg (1992)", "Rufibach (2011)"), col =
    c(2, 4), lty = c(2, 1), lwd = c(3, 2), bty = "n")

## compute selection bias
eta <- sqrt(res3$sigma ^ 2 + res3$u ^ 2)
bias <- effectBias(res3$y, res3$u, res3$w, res3$theta, eta)
bias


##------------------------------------------
## Compute p-value to assess null hypothesis of no selection
##------------------------------------------
## compute null parameters
meta.toba <- metagen(TE = y, seTE = u, sm = "MD", level = 0.95,
    comb.fixed = TRUE, comb.random = TRUE)
theta0 <- meta.toba$TE.random
sigma0 <- meta.toba$tau

M <- 1000
res <- DearBeggMonotonePvalSelection(y, u, theta0, sigma0, lam = lam1,
    M = M, maxiter = 2000)

## plot all the computed monotone functions
plot(0, 0, xlim = c(0, 1), ylim = c(0, 1), type = "n", xlab = "p-values",
```

```
    ylab = expression(w(p)))
abline(v = 0.05, lty = 3)
for (i in 1:M){weightLine(p, w = res$res.mono[1:k, i], col0 = grey(0.8),
    lwd0 = 1, lty0 = 1)}
rug(p, lwd = 2)
weightLine(p, w = res$mono0, col0 = 2, lwd0 = 1, lty0 = 1)

## End(Not run)
```

DearBeggMonotoneCItheta *Compute an approximate profile likelihood ratio confidence interval for effect estimate*

Description

Under some assumptions on the true underlying p-value density the usual likelihood ratio theory for the finite-dimensional parameter of interest, θ, holds although we estimate the infinite-dimensional nuisance parameter w, see Murphy and van der Vaart (2000). These functions implement such a confidence interval. To this end we compute the set

$$\{\theta : \tilde{l}(\theta, \hat{\sigma}(\theta), \hat{w}(\theta)) \ge c\}$$

where $c = -0.5 \cdot \chi^2_{1-\alpha}(1)$ and $\tilde{l}$ is the relative profile log-likelihood function.

The functions DearBeggProfileLL and DearBeggToMinimizeProfile are not intended to be called by the user directly.

Usage

```
DearBeggMonotoneCItheta(res, lam = 2, conf.level = 0.95, maxiter = 500)
DearBeggProfileLL(z, res0, lam, conf.level = 0.95, maxiter = 500)
DearBeggToMinimizeProfile(vec, theta, y, u, lam)
```

Arguments

res	Output from function DearBeggMonotone.
lam	Weight of the first entry of w in the likelihood function. Should be the same as used to generate res.
conf.level	Confidence level of confidence interval.
maxiter	Maximum number of iterations of differential evolution algorithm used in computation of confidence limits. Increase this number to get higher accuracy.
z	Variable to maximize over, corresponds to θ.
res0	Output from DearBeggMonotone, contains initial estimates.
vec	Vector of parameters over which we maximize.
theta	Current θ.
y	Normally distributed effect sizes.
u	Associated standard errors.

Value

A list with the element

ci.theta — that contains the profile likelihood confidence interval for θ.

Note

Since we have to numerically find zeros of a suitable function, via uniroot, to get the limits and each iteration involves computation of $w(\theta)$ via a variant of DearBeggMonotone, computation of a confidence interval may take some time (typically seconds to minutes).

Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

References

Murphy, S. and van der Vaart, A. (2000). On profile likelihood. *J. Amer. Statist. Assoc.*, **95**, 449–485.

Rufibach, K. (2011). Selection Models with Monotone Weight Functions in Meta-Analysis. *Biom. J.*, **53**(4), 689–704.

See Also

The estimate under a monotone selection function can be computed using *DearBeggMonotone*.

Examples

```
## Not run:
## compute confidence interval for theta in the education dataset
data(education)
N  <- education$N
y  <- education$theta
u  <- sqrt(2 / N)
lam1 <- 2
res.edu <- DearBeggMonotone(y, u, lam = lam1, maxiter = 1000,
    CR = 1, trace = FALSE)
r1 <- DearBeggMonotoneCItheta(res.edu, lam = 2, conf.level = 0.95)
res.edu$theta
r1$ci.theta

## compute confidence interval for theta in the passive smoking dataset
data(passive_smoking)
u <- passive_smoking$selnRR
y <- passive_smoking$lnRR
lam1 <- 2
res.toba <- DearBeggMonotone(y, u, lam = lam1, maxiter = 1000,
    CR = 1, trace = FALSE)
r2 <- DearBeggMonotoneCItheta(res.toba, lam = 2, conf.level = 0.95)
res.toba$theta
```

```
r2$ci.theta

## End(Not run)
```

DearBeggMonotonePvalSelection
Compute simulation-based p-value to assess null hypothesis of no selection

---

Description

This function computes a simulation-based p-value to assess the null hypothesis of no selection. For details we refer to Rufibach (2011, Section 6).

Usage

```
DearBeggMonotonePvalSelection(y, u, theta0, sigma0, lam = 2, M = 1000,
    maxiter = 1000, test.stat = function(x){return(min(x))})
```

Arguments

`y`	Normally distributed effect sizes.
`u`	Associated standard errors.
`theta0`	Initial estimate for θ.
`sigma0`	Initial estimate for σ.
`lam`	Weight of the first entry of w in the likelihood function. Should be the same as used to generate `res`.
`M`	Number of runs to compute p-value.
`maxiter`	Maximum number of iterations of differential evolution algorithm. Increase this number to get higher accuracy.
`test.stat`	A function that takes as argument a vector and returns a number. Defines the test statistic to be used on the estimated selection function w.

Value

`pval`	The computed p-value.
`res.mono`	The monotone estimates for each simulation run.
`mono0`	The monotone estimates for the original data.
`Ti`	The test statistics for each simulation run.
`T0`	The test statistic for the original data.
`ran.num`	Matrix that contains the generated p-values.

Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

References

Rufibach, K. (2011). Selection Models with Monotone Weight Functions in Meta-Analysis. *Biom. J.*, **53**(4), 689–704.

See Also

This function is illustrated in the help file for DearBegg.

`education` *Dataset open vs. traditional education on creativity*

Description

Dataset of studies of effect of open vs. traditional education on creativity. Standard dataset to illustrate selection models in meta-analysis.

Usage

```
data(education)
```

Format

A data frame with 10 observations on the following 5 variables.

`i` Study number.

`N` Sample size of study.

`theta` Estimated effect size.

`t` t test statistic, $t = \theta/\sqrt{2/N}$.

`q` Defrees of freedom, $q = 2N - 2$.

References

Dear, Keith B.G. and Begg, Colin B. (1992). An Approach for Assessing Publication Bias Prior to Performing a Meta-Analysis. *Statist. Sci.*, **7(2)**, 237–245.

Hedges, L. and Olkin, I. (1985). Statistical Methods for Meta-Analysis. Academic, Orlando, Florida.

Iyengar, S. and Greenhouse, J.B. (1988). Selection models and the file drawer problem (including rejoinder). *Statist. Sci.*, **3**, 109–135.

See Also

This dataset is analyzed in the help file for DearBegg.

effectBias *Compute bias for each effect size based on estimated weight function*

Description

Based on the estimated weight function an explicit formula for the bias of each initial effect estimate can be derived, see Rufibach (2011). This function implements computation of this bias and is called by DearBegg and DearBeggMonotone.

Usage

```
effectBias(y, u, w, theta, eta)
```

Arguments

y	Normally distributed effect sizes.
u	Associated standard errors.
w	Vector of estimated weights as computed by either DearBegg or DearBeggMonotone.
theta	Effect size estimate.
eta	Standard error of effect size estimate.

Value

A list consisting of the following elements:

dat	Matrix with columns y, u, y, p, bias, y - bias, bias / y, where the rows are provided in decreasing order of p-values.

Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

References

Rufibach, K. (2011). Selection Models with Monotone Weight Functions in Meta-Analysis. *Biom. J.*, **53**(4), 689–704.

Examples

```
# For an illustration see the help file for the function DearBegg().
```

`IyenGreen` *Compute MLE and weight functions of Iyengar and Greenhouse (1988)*

Description

Two parameteric weight functions for selection models were introduced in Iyengar and Greenhouse (1988):

$$w_1(x; \beta, q) = |x|^\beta / t(q, \alpha)$$

$$w_2(x; \gamma, q) = e^{-\gamma}$$

if $|x| \le t(q, \alpha)$ and $w_1(x; \beta, q) = w_2(x; \gamma, q) = 1$ otherwise. Here, $t(q, \alpha)$ is the α-quantile of a t distribution with q degrees of freedom. The functions w_1 and w_2 are used to model the selection process that may be present in a meta analysis, in a model where effect sizes are assumed to follow a t distribution. We have implemented estimation of the parameters in this model in `IyenGreenMLE` and plotting in `IyenGreenWeight`. The functions `normalizeT` and `IyenGreenLoglikT` are used in computation of ML estimators and not intended to be called by the user. For an example how to use `IyenGreenMLE` and `IyenGreenWeight` we refer to the help file for `DearBegg`.

Usage

```
normalizeT(s, theta, b, q, N, type = 1, alpha = 0.05)
IyenGreenLoglikT(para, t, q, N, type = 1)
IyenGreenMLE(t, q, N, type = 1, alpha = 0.05)
IyenGreenWeight(x, b, q, type = 1, alpha = 0.05)
```

Arguments

`s`	Quantile where normalizing integrand should be computed.
`theta`	Vector containing effect size estimates of the meta analysis.
`b`	Parameter that governs shape of the weight function. Equals β for w_1 and γ for w_2.
`q`	Degrees of freedom in the denominator of w_1, w_2. Must be a real number.
`N`	Number of observations in each trial.
`type`	Type of weight function in Iyengar & Greenhouse (1988). Either 1 (for w_1) or 2 (for w_2).
`alpha`	Quantile to be used in the denominator of w_1, w_2.
`para`	Vector in R^2 over which log-likelihood function is maximized.
`t`	Vector of real numbers, t test statistics.
`x`	Vector of real numbers where weight function should be computed at.

Details

Note that these weight functions operate on the scale of t statistics, not p-values.

Value

See example in DearBegg for details.

Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

References

Iyengar, S. and Greenhouse, J.B. (1988). Selection models and the file drawer problem (including rejoinder). *Statist. Sci.*, **3**, 109–135.

See Also

For nonparametric estimation of weight functions see DearBegg.

Examples

```
# For an illustration see the help file for the function DearBegg().
```

---

passive_smoking — *Dataset on the effect of environmental tobacco smoke*

Description

Effect of environmental tobacco smoke on lung-cancer in lifetime non-smokers.

Usage

```
data(passive_smoking)
```

Format

A data frame with 37 observations on the following 2 variables.

`lnRR` Log-relative risk.

`selnRR` Standard error of log-relative risk.

Details

The sample consists of lung cancer patients and controls that were lifelong non-smokers. The effect of interest is measured by the relative risk of lung cancer according to whether the spouse currently smoked or had never smoked.

References

Hackshaw, A. K., Law, M. R., and Wald, N.J. The accumulated evidence on lung cancer and environmental tobacco smoke. *BMJ*, **315**, 980–988.

See Also

This dataset is analyzed in the help file for `DearBegg`.

`pPool` *Pool p-values in pairs*

Description

To avoid unidentifiability in estimation of a selection function, Dear and Begg (1992) pool p-values in pairs.

Usage

```
pPool(p)
```

Arguments

`p`	Vector of p-values.

Value

Vector of pooled p-values.

Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

References

Dear, K.B.G. and Begg, C.B. (1992). An Approach for Assessing Publication Bias Prior to Performing a Meta-Analysis. *Statist. Sci.*, **7(2)**, 237–245.

Rufibach, K. (2011). Selection Models with Monotone Weight Functions in Meta-Analysis. *Biom. J.*, **53**(4), 689–704.

See Also

This function is used in `weightLine`.

Examples

```
# This function is used in the help file for the function DearBegg().
```

Pval *Functions for the distribution of p-values*

Description

The density of the p-value generated by a test of the hypothesis

$$H_0 : Y \sim N(0, \sigma^2) \;\; vs. \;\; H_1 : Y \sim N(\theta, \eta^2)$$

has the form

$$f(p; \theta, \sigma, \eta) = \frac{\sigma}{2\eta} \frac{\phi\Big((-\sigma\Phi^{-1}(p/2) - \theta)/\eta\Big) + \phi\Big((\sigma\Phi^{-1}(p/2) - \theta)/\eta\Big)}{\phi(\Phi^{-1}(p/2))}$$

where $\eta^2 = u^2 + \sigma^2$. We refer to Rufibach (2011) for details.

Usage

```
dPval(p, u, theta, sigma2)
pPval(q, u, theta, sigma2)
qPval(prob, u, theta, sigma2)
rPval(n, u, theta, sigma2, seed = 1)
```

Arguments

p, q	Quantile.
prob	Probability.
u	Standard error of the effect size.
theta	Effect size.
sigma2	Random effect variance component.
n	Number of random numbers to be generated.
seed	Seed to set.

Value

`dPval` gives the density, `pPval` gives the distribution function, `qPval` gives the quantile function, and `rPval` generates random deviates for the density $f(p; \theta, \sigma, \eta)$.

Author(s)

Kaspar Rufibach (maintainer), <kaspar.rufibach@gmail.com>,
http://www.kasparrufibach.ch

References

Dear, K.B.G. and Begg, C.B. (1992). An Approach for Assessing Publication Bias Prior to Performing a Meta-Analysis. *Statist. Sci.*, **7(2)**, 237–245.

Rufibach, K. (2011). Selection Models with Monotone Weight Functions in Meta-Analysis. *Biom. J.*, **53**(4), 689–704.

`weightLine` *Function to plot estimated weight functions*

Description

This function faclitiates plotting of estimated weight functions according to the method in Dear and Begg (1992) or its non-increasing version described in Rufibach (2010).

Usage

```
weightLine(p, w, col0, lwd0, lty0 = 1, type = c("pval", "empirical")[1])
```

Arguments

`p`	Vector of p-values.
`w`	Vector of estimated weights, as outputted by `DearBegg` or `DearBeggMonotone`.
`col0`	Color of line that is drawn.
`lwd0`	Line width.
`lty0`	Line type.
`type`	Should weights be drawn versus original p-values (`type == "pval"`) or versus the empirical distribution of the p-values (`type === "empirical"`).

Author(s)

Kaspar Rufibach (maintainer), `<kaspar.rufibach@gmail.com>`,
`http://www.kasparrufibach.ch`

References

Dear, K.B.G. and Begg, C.B. (1992). An Approach for Assessing Publication Bias Prior to Performing a Meta-Analysis. *Statist. Sci.*, **7(2)**, 237–245.

Rufibach, K. (2011). Selection Models with Monotone Weight Functions in Meta-Analysis. *Biom. J.*, **53**(4), 689–704.

See Also

This function is used in `weightLine`.

Examples

```
# This function is used in the help file for the function DearBegg().
```

Index