Package ‘zcurve’

October 14, 2022

Title An Implementation of Z-Curves

Version 2.1.2

Maintainer František Bartoš <f.bartos96@gmail.com>

Description An implementation of z-curves - a method for estimating expected discovery and replicability rates on the bases of test-statistics of published studies. The package provides functions for fitting the new density and EM version (Bartoš & Schimmack, 2020, <doi:10.31234/osf.io/urgtn>), censored observations, as well as the original density z-curve (Brunner & Schimmack, 2020, <doi:10.15626/MP.2018.874>).
Furthermore, the package provides summarizing and plotting functions for the fitted z-curve objects.
See the aforementioned articles for more information about the z-curves, expected discovery and replicability rates, validation studies, and limitations.

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

Imports Rcpp (>= 1.0.2), nleqslv, stats, evmix, graphics, Rdpack

LinkingTo Rcpp

Suggests spelling, testthat, vdiffr

Language en-US

RdMacros Rdpack

URL https://fbartos.github.io/zcurve/

NeedsCompilation yes

Author František Bartoš [aut, cre],
Ulrich Schimmack [aut]

Repository CRAN

Date/Publication 2022-02-07 21:00:02 UTC

R topics documented:

`control_density` *Control settings for the z-curve 2.0 density algorithm*

Description

All settings are passed to the density fitting algorithm. All unspecified settings are set to the default value. Setting `model = "KD2"` sets all settings to the default value irrespective of any other setting and fits z-curve as describe in Bartoš and Schimmack (2020). In order to fit the z-curve 1.0 density algorithm, set `model = "KD1"` and go to control_density_v1

Arguments

`version`	Which version of z-curve should be fitted. Defaults to `2` = z-curve 2.0. Set to `1` in order to fit the original version of z-curve. For its settings page go to control_density_v1.
`model`	A type of model to be fitted, defaults to `"KD2"` (another possibility is `"KD1"` for the original z-curve 1.0, see control_density_v1 for its settings)
`sig_level`	An alpha level of the test statistics, defaults to `.05`
`a`	A beginning of fitting interval, defaults to `qnorm(sig_level/2,lower.tail = F)`
`b`	An end of fitting interval, defaults to `6`
`mu`	Means of the components, defaults to `seq(0,6,1)`
`sigma`	A standard deviation of the components, "Don't touch this" \- Ulrich Schimmack, defaults to `1`

theta_min	Lower limits for weights, defaults to rep(0,length(mu))
theta_max	Upper limits for weights, defaults to rep(1,length(mu))
max_iter	A maximum number of iterations for the nlminb optimization for fitting mixture model, defaults to 150
max_eval	A maximum number of evaluation for the nlminb optimization for fitting mixture model, defaults to 1000
criterion	A criterion to terminate nlminb optimization, defaults to 1e-03
bw	A bandwidth of the kernel density estimation, defaults to .10
aug	Augment truncated kernel density, defaults to TRUE
aug.bw	A bandwidth of the augmentation, defaults to .20
n.bars	A resolution of density function, defaults to 512
density_dbc	Use bckden to estimate a truncated kernel density, defaults to FALSE, in which case density is used
compute_FDR	Whether to compute FDR, leads to noticeable increase in computation, defaults to FALSE
criterion_FDR	A criterion for estimating the maximum FDR, defaults to .02
criterion_FDR_dbc	A criterion for estimating the maximum FDR using the bckden function, defaults to .01
precision_FDR	A maximum FDR precision, defaults to .05

References

Bartoš F, Schimmack U (2020). “Z-curve. 2.0: Estimating Replication Rates and Discovery Rates.” doi: 10.31219/osf.io/wr93f, submitted for publication.

See Also

zcurve(), control_density_v1, control_EM

Examples

```
# to decrease the criterion and increase the number of iterations
ctrl <- list(
   max_iter  = 300,
   criterion = 1e-4
)
zcurve(OSC.z, method = "density", control = ctrl)
```

control_density_v1 *Control settings for the original z-curve density algorithm*

Description

All settings are passed to the density fitting algorithm. All unspecified settings are set to the default value. Setting `model = "KD1"` sets all settings to the default value irrespective of any other setting and fits z-curve as described in Brunner and Schimmack (2020).

Arguments

`version`	Set to `1` to fit the original version of z-curve. Defaults to `2` = the updated version of z-curve. For its settings page go to control_density.
`model`	A type of model to be fitted, defaults to `"KD1"` (the only possibility)
`sig_level`	An alpha level of the test statistics, defaults to `.05`
`a`	A beginning of fitting interval, defaults to `qnorm(sig_level/2,lower.tail = F)`
`b`	An end of fitting interval, defaults to `6`
`K`	Number of mixture components, defaults to `3`
`max_iter`	A maximum number of iterations for the nlminb optimization for fitting mixture model, defaults to `150`
`max_eval`	A maximum number of evaluation for the nlminb optimization for fitting mixture model, defaults to `300`
`criterion`	A criterion to terminate nlminb optimization, defaults to `1e-10`
`bw`	A bandwidth of the kernel density estimation, defaults to `"nrd0"`

References

Brunner J, Schimmack U (2020). "Estimating population mean power under conditions of heterogeneity and selection for significance." *Meta-Psychology*, **4**. doi: 10.15626/MP.2018.874.

See Also

`zcurve()`, control_density, control_EM

Examples

```
# to increase the number of iterations
ctrl <- list(
   version   = 1,
   max_iter  = 300
)
zcurve(OSC.z, method = "density", control = ctrl)
```

control_EM *Control settings for the zcurve EM algorithm*

Description

All these settings are passed to the Expectation Maximization fitting algorithm. All unspecified settings are set to the default value. Setting `model = "EM"` sets all settings to the default value irrespective of any other setting and fits z-curve as described in Bartoš and Schimmack (2020)

Arguments

`model`	A type of model to be fitted, defaults to `"EM"` for a z-curve with 7 z-scores centered components.
`sig_level`	An alpha level of the test statistics, defaults to `.05`
`a`	A beginning of fitting interval, defaults to `qnorm(sig_level/2,lower.tail = F)`
`b`	An end of fitting interval, defaults to `5`
`mu`	Means of the components, defaults to `0:6`
`sigma`	A standard deviation of the components, defaults to `rep(1, length(mu))`
`theta_alpha`	A vector of alpha parameters of a Dirichlet distribution for generating random starting values for the weights, defaults to `rep(.5, length(mu))`
`theta_max`	Upper limits for weights, defaults to `rep(1,length(mu))`
`criterion`	A criterion to terminate the EM algorithm, defaults to `1e-6`
`criterion_start`	A criterion to terminate the starting phase of the EM algorithm, defaults to `1e-3`
`criterion_boot`	A criterion to terminate the bootstrapping phase of the EM algorithm, defaults to `1e-5`
`max_iter`	A maximum number of iterations of the EM algorithm (not including the starting iterations) defaults to `10000`
`max_iter_start`	A maximum number of iterations for the starting phase of EM algorithm, defaults to `100`
`max_iter_boot`	A maximum number of iterations for the booting phase of EM algorithm, defaults to `100`
`fit_reps`	A number of starting fits to get the initial position for the EM algorithm, defaults to `100`

References

Bartoš F, Schimmack U (2020). "Z-curve. 2.0: Estimating Replication Rates and Discovery Rates." doi: 10.31219/osf.io/wr93f, submitted for publication.

See Also

`zcurve()`, control_density

Examples

```
# to increase the number of starting fits
# and change the means of the mixture components

ctrl <- list(
   fit_reps  = 50,
   mu = c(0, 1.5, 3, 4.5, 6)
)
zcurve(OSC.z, method = "EM", control = ctrl)
```

---

head.zcurve_data   *Prints first few rows of a z-curve data object*

Description

Prints first few rows of a z-curve data object

Usage

```
## S3 method for class 'zcurve_data'
head(x, ...)
```

Arguments

x	z-curve data object
...	Additional arguments

See Also

zcurve_data()

---

is.zcurve   *Reports whether x is a zcurve object*

Description

Reports whether x is a zcurve object

Usage

```
is.zcurve(x)
```

Arguments

x	an object to test

`OSC.z` *Z-scores from subset of original studies featured in OSC 2015 reproducibility project*

Description

The dataset contains z-scores from subset of original studies featured in psychology reproducibility project (Collaboration and others 2015). Only z-scores from studies with unambiguous original outcomes are supplied (eliminating 7 studies with marginally significant results). The real replication rate for those studies is 35/90 (the whole project reports 36/97).

Usage

```
OSC.z
```

Format

A vector with 90 observations

References

Collaboration OS, others (2015). "Estimating the reproducibility of psychological science." *Science*, **349**(6251). doi: 10.1126/science.aac4716.

`plot.zcurve` *Plot fitted z-curve object*

Description

Plot fitted z-curve object

Usage

```
## S3 method for class 'zcurve'
plot(
  x,
  annotation = FALSE,
  CI = FALSE,
  extrapolate = FALSE,
  y.anno = c(0.95, 0.88, 0.78, 0.71, 0.61, 0.53, 0.43, 0.35),
  x.anno = 0.6,
  cex.anno = 1,
  ...
)
```

Arguments

`x`	Fitted z-curve object
`annotation`	Add annotation to the plot. Defaults to `FALSE`.
`CI`	Plot confidence intervals for the estimated z-curve. Defaults to `FALSE`.
`extrapolate`	Scale the chart to the extrapolated area. Defaults to `FALSE`.
`y.anno`	A vector of length 8 specifying the y-positions of the individual annotation lines relative to the figure's height. Defaults to `c(.95, .88, .78, .71, .61, .53, .43, .35)`
`x.anno`	A number specifying the x-position of the block of annotations relative to the figure's width.
`cex.anno`	A number specifying the size of the annotation text.
`...`	Additional arguments including `main`, `xlab`, `ylab`, `xlim`, `ylim`, `cex.axis`, `cex.lab`

See Also

`zcurve()`

Examples

```
# simulate some z-statistics and fit a z-curve
z <- abs(rnorm(300,3))
m.EM <- zcurve(z, method = "EM", bootstrap = 100)

# plot the z-curve
plot(m.EM)

# add annotation text and model fit CI
plot(m.EM, annotation = TRUE, CI = TRUE)

# change the location of the annotation to the left
plot(m.EM, annotation = TRUE, CI = TRUE, x_text = 0)
```

`power_to_z` *Compute z-score corresponding to a power*

Description

A function for computing z-scores of two-sided tests corresponding to power `power` for a given significance level alpha `alpha` (or corresponding cut-off z-statistic `a`).

Usage

```
power_to_z(
  power,
  alpha = 0.05,
  a = stats::qnorm(alpha/2, lower.tail = FALSE),
  two.sided = TRUE,
  nleqslv_control = list(xtol = 1e-15, maxit = 300, stepmax = 0.5)
)
```

Arguments

`power`	A vector of powers
`alpha`	Level of significance alpha
`a`	Or, alternatively a z-score corresponding to `alpha`
`two.sided`	Whether directionality of the effect size should be taken into account.
`nleqslv_control`	A named list of control parameters passed to the nleqslv function used for solving the inverse of z_to_power function.

Examples

```
# z-scores corresponding to the (aproximate) power of components of EM2
power_to_z(c(0.05, 0.20, 0.40, 0.60, 0.80, 0.974, 0.999), alpha = .05)
```

---

print.estimates.zcurve

Prints estimates from z-curve object

Description

Prints estimates from z-curve object

Usage

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

Arguments

`x`	Estimate of a z-curve object
`...`	Additional arguments

See Also

`zcurve()`

print.summary.zcurve *Prints summary object for z-curve method*

Description

Prints summary object for z-curve method

Usage

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

Arguments

x	Summary of a z-curve object
...	Additional arguments

See Also

zcurve()

print.zcurve *Prints a fitted z-curve object*

Description

Prints a fitted z-curve object

Usage

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

Arguments

x	Fitted z-curve object
...	Additional arguments

See Also

zcurve()

print.zcurve_data *Prints a z-curve data object*

Description

Prints a z-curve data object

Usage

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

Arguments

x	z-curve data object
...	Additional arguments

See Also

zcurve_data()

summary.zcurve *Summarize fitted z-curve object*

Description

Summarize fitted z-curve object

Usage

```
## S3 method for class 'zcurve'
summary(
  object,
  type = "results",
  all = FALSE,
  ERR.adj = 0.03,
  EDR.adj = 0.05,
  round.coef = 3,
  ...
)
```

Arguments

`object`	A fitted z-curve object.
`type`	Whether the results `"results"` or the mixture mode parameters `"parameters"` should be returned. Defaults to `"results"`.
`all`	Whether additional results, such as file drawer ration, expected and missing number of studies, and Soric FDR be returned. Defaults to `FALSE`
`ERR.adj`	Confidence intervals adjustment for ERR. Defaults to `.03` as proposed by Bartos & Schimmack (in preparation).
`EDR.adj`	Confidence intervals adjustment for EDR. Defaults to `.05` as proposed by Bartos & Schimmack (in preparation).
`round.coef`	To how many decimals should the coefficient be rounded. Defaults to `3`.
`...`	Additional arguments

Value

Summary of a z-curve object

See Also

`zcurve()`

`zcurve` *Fit a z-curve*

Description

`zcurve` is used to fit z-curve models. The function takes input of z-statistics or two-sided p-values and returns object of class `"zcurve"` that can be further interrogated by summary and plot function. It default to EM model, but different version of z-curves can be specified using the `method` and `control` arguments. See 'Examples' and 'Details' for more information.

Usage

```
zcurve(
  z,
  z.lb,
  z.ub,
  p,
  p.lb,
  p.ub,
  data,
  method = "EM",
  bootstrap = 1000,
  control = NULL
)
```

Arguments

`z`	a vector of z-scores.
`z.lb`	a vector with start of censoring intervals of censored z-scores.
`z.ub`	a vector with end of censoring intervals of censored z-scores.
`p`	a vector of two-sided p-values, internally transformed to z-scores.
`p.lb`	a vector with start of censoring intervals of censored two-sided p-values.
`p.ub`	a vector with end of censoring intervals of censored two-sided p-values.
`data`	an object created with `zcurve_data()` function.
`method`	the method to be used for fitting. Possible options are Expectation Maximization `"EM"` and density `"density"`, defaults to `"EM"`.
`bootstrap`	the number of bootstraps for estimating CI. To skip bootstrap specify `FALSE`.
`control`	additional options for the fitting algorithm more details in control EM or control density.

Details

The function returns the EM method by default and changing `method = "density"` gives the KD2 version of z-curve as outlined in Bartoš and Schimmack (2020). For the original z-curve (Brunner and Schimmack 2020), referred to as KD1, specify `'control = "density", control = list(model = "KD1")'`.

Value

The fitted z-curve object

References

Bartoš F, Schimmack U (2020). "Z-curve. 2.0: Estimating Replication Rates and Discovery Rates." doi: 10.31219/osf.io/wr93f, submitted for publication.

Brunner J, Schimmack U (2020). "Estimating population mean power under conditions of heterogeneity and selection for significance." *Meta-Psychology*, **4**. doi: 10.15626/MP.2018.874.

See Also

`summary.zcurve()`, `plot.zcurve()`, control_EM, control_density

Examples

```
# load data from OSC 2015 reproducibility project
OSC.z

# fit an EM z-curve (with disabled bootstrap due to examples times limits)
m.EM <- zcurve(OSC.z, method = "EM", bootstrap = FALSE)
# a version with 1000 boostraped samples would looked like:
m.EM <- zcurve(OSC.z, method = "EM", bootstrap = 1000)
```

```
# or KD2 z-curve (use larger bootstrap for real inference)
m.D <- zcurve(OSC.z, method = "density", bootstrap = FALSE)

# inspect the results
summary(m.EM)
summary(m.D)
# see '?summary.zcurve' for more output options

# plot the results
plot(m.EM)
plot(m.D)
# see '?plot.zcurve' for more plotting options

# to specify more options, set the control arguments
# ei. increase the maximum number of iterations and change alpha level
ctr1 <- list(
  "max_iter" = 9999,
  "alpha"    = .10
  )
m1.EM <- zcurve(OSC.z, method = "EM", bootstrap = FALSE, control = ctr1)
# see '?control_EM' and '?control_density' for more information about different
# z-curves specifications
```

---

zcurve.estimates *z-curve estimates*

Description

The following functions extract estimates from the z-curve object.

Usage

```
ERR(object, round.coef = 3)

EDR(object, round.coef = 3)

ODR(object, round.coef = 3)

Soric(object, round.coef = 3)

file_drawer_ration(object, round.coef = 3)

expected_n(object, round.coef = 0)

missing_n(object, round.coef = 0)

significant_n(object)

included_n(object)
```

Arguments

`object`	the z-curve object
`round.coef`	rounding for the printed values

Details

Technically, ODR, significant n, and included n are not z-curve estimates but they are grouped in this category for convenience.

See Also

`zcurve()`

`zcurve_data` *Prepare data for z-curve*

Description

`zcurve_data` is used to prepare data for the `zcurve()` function. The function transform strings containing reported test statistics `"z"`, `"t"`, `"f"`, `"chi"`, `"p"` into two-sided p-values. Test statistics reported as inequalities are as considered to be censored as well as test statistics reported with low accuracy (i.e., rounded to too few decimals). See details for more information.

Usage

```
zcurve_data(data, rounded = TRUE, stat_precise = 2, p_precise = 3)
```

Arguments

`data`	a vector strings containing the test statistics.
`rounded`	an optional argument specifying whether de-rounding should be applied. Defaults to `FALSE` to treat all input as exact values or a numeric vector with values specifying precision of the input. The other option, `FALSE`, automatically extracts the number of decimals from input and treats the input as censored if it does not surpass the `stat_precise` and the `p_precise` thresholds.
`stat_precise`	an integer specifying the numerical precision of `"z"`, `"t"`, `"f"` statistics treated as exact values.
`p_precise`	an integer specifying the numerical precision of p-values treated as exact values.

Details

By default, the function extract the type of test statistic:

- `"F(df1, df2)=x"`F-statistic with df1 and df2 degrees of freedom,
- `"chi(df)=x"`Chi-square statistic with df degrees of freedom,
- `"t(df)=x"`for t-statistic with df degrees of freedom,

- "z=x"for z-statistic,
- "p=x"for p-value.

The input is not case sensitive and automatically removes empty spaces. Furthermore, inequalities ("<" and ">") can be used to denote censoring. I.e., that the p-value is lower than "x" or that the test statistic is larger than "x" respectively. The automatic de-rounding procedure (if `rounded = TRUE`) treats p-values with less decimal places than specified in `p_precise` or test statistics with less decimal places than specified in `stat_precise` as censored on an interval that could result in a given rounded value. I.e., a "`p = 0.03`" input would be de-rounded as a p-value lower than 0.035 but larger than 0.025.

Value

An object of type "`zcurve_data`".

See Also

`zcurve()`, `print.zcurve_data()`, `head.zcurve_data()`

Examples

```
# Specify a character vector containing the test statistics
data <- c("z = 2.1", "t(34) = 2.21", "p < 0.03", "F(2,23) > 10", "p = 0.003")

# Obtain the z-curve data object
data <- zcurve_data(data)

# inspect the resulting object
data
```

`z_to_power` *Compute power corresponding to z-scores*

Description

A function for computing power of two-sided tests corresponding to z-scores for a given significance level. `alpha` (or corresponding cut-off z-score `a`)

Usage

```
z_to_power(
  z,
  alpha = 0.05,
  a = stats::qnorm(alpha/2, lower.tail = FALSE),
  two.sided = TRUE
)
```

Arguments

`z`	A vector of z-scores
`alpha`	Level of significance alpha
`a`	Or, alternatively a z-score corresponding to `alpha`
`two.sided`	Whether directionality of the effect size should be taken into account.

Examples

```
# mean powers corresponding to the mean components of KD2
z_to_power(0:6, alpha = .05)
```

Index