Package 'unrepx'

May 8, 2026

Type Package

Title Analysis and Graphics for Unreplicated Experiments

Version 1.0-2

Date 2022-08-11

BugReports https://github.com/rvlenth/unrepx/issues

Description Provides half-normal plots, reference plots, and Pareto plots of effects from an unreplicated experiment, along with various pseudo-standard-error measures, simulated reference distributions, and other tools. Many of these methods are described in Daniel C. (1959) <doi:10.1080/00401706.1959.10489866> and/or Lenth R.V. (1989) <doi:10.1080/00401706.1989.10488595>, but some new approaches are added and integrated in one package.

Suggests knitr, rmarkdown

VignetteBuilder knitr

License GPL (>= 2)

NeedsCompilation no

Author Russell Lenth [aut, cre, cph]

Maintainer Russell Lenth <russell-lenth@uiowa.edu>

Repository CRAN

Date/Publication 2022-08-11 21:50:03 UTC

Contents

`unrepx-package` *Analysis and graphics for unreplicated experiments*

Description

Provides half-normal plots, reference plots, and Pareto plots of effects from an unreplicated experiment, along with various PSE measures, simulated reference distributions, and other tools

Details

See the package vignette: `vignette("overview", package="unrepx")`

Author(s)

Russell V. Lenth

Maintainer: Russell V. Lenth <russell-lenth@uiowa.edu>

`daniel.paper` *Half-normal graph paper*

Description

The `daniel.paper` function draws a graph-paper grid suitable for manually constructing a half-normal plot of effects. The paper is customized to the number of effects to be plotted, thus making the task as simple as possible.

Usage

```
daniel.paper(n.effects = 15, linear = 0:40, a = 0.375,
    rank.axis = c("y", "x"), type = c("half.normal", "normal"),
    theme = "blue", lin.lab, rank.lab)
```

Arguments

`n.effects`	Numeric vector of effects or contrasts to be explored.
`linear`	sequence of numbers to use for the linear scale.
`a`	The adjustment used in scaling and centering ranks in the interval (0, 1). The ith ordered (half-) normal score is computed as the $(i-a)/(n+1-2*a)$ quantile of the reference distribution. The value should always be less than 1, and is recommended to be in [0,.5]. The scale becomes somewhat more nonlinear as a increases.
`rank.axis`	Character value of `"y"` or `"x"` selecting whether the ranks of effects should be plotted as the vertical or the horizontal coordinate.

`type` Character value matching `"half.normal"` or `"normal"` specifying whether the scaling is for a half-normal or a normal plot. The latter is not recommended.

`theme` Character giving the name of the theme to use for the grid lines. The default of `"blue"` plots the major grid lines in blue and the minor ones in cyan. Other built-in themes are `"bw"` (black majors, and dotted black minors), `"gray"` (black majors and gray minors), and `"bold"` (all lines black, but major lines are heavier). See details below for information on how to create one's own theme.

`lin.lab`, `rank.lab` Labels for the linear and rank scales.

Details

Use of `type = "normal"` is discouraged, as is discussed in the Details section in `hnplot`.

A custom theme, say `"foo"`, may be created by creating a `list` named `foo_thm`, which should contain two elements named `major` and `minor`, and (optionally) `axis`. `major` must be a named `list` with graphical parameters `col`, `lty`, and `lwd` as described in `par`. `minor` only requires `col` and `lty`, as minor lines are always drawn using `lwd = 1`. `axis` must contain `col` and `lwd`, which are used for the tick marks and labels on the rank scale. If `axis` is not included, `major` is used.

Value

Nothing is returned.

Author(s)

Russell V. Lenth

References

Daniel, C (1959) Use of Half-Normal Plots in Interpreting Factorial Two-Level Experiments. *Technometrics*, 1(4), 311-341

See Also

`hnplot`

Examples

```
require("unrepx")

old.par <- par(mar = c(5, 5, 1, 1) + .1)
daniel.paper(15)

neon_thm <- list(
    major = list(col = "green", lty = 1, lwd = 2),
    minor = list(col = "orange", lty = 1),
    axis = list(col = "magenta", lwd = 2)
)
daniel.paper(31, rank.axis = "x", theme = "neon")

par(old.par)
```

dot.plot *Resizeable dot plots*

Description

Flexible stacked-dot plots of relatively small samples, and provisions for identifying points. The dot plot is responsive to resizing of the plot window: a vertical resizing does not affect the vertical spacing of the dots, and a horizontal resizing may lead to re-binning to fit well within the range.

Usage

```
dot.plot(x, pch = 16, cex.dot = 1, spacing = 1, xlab, xlim = range(x), ...)

dot.id(env, height.id = 2, cex.id = 1, col.id = "black")
dot.mod(env, ...)
```

Arguments

x	Numeric vector of values to be plotted.
pch	Plotting character for the dots. Default is a filled circle.
cex.dot	Size of dots relative to par("cex").
spacing	Factor for adjusting the vertical spacing of stacked dots.
xlab, xlim	The usual graphical parameters (see par), but used by dot.plot or dot.id.
env	Environment returned by dot.plot.
height.id	Height (in character heights) above the horizontal axis for displaying identification labels.
cex.id, col.id	cex and col settings for labeling identified points.
...	In dot.plot, additional graphical parameters (see par) used in constructing the plot. In dot.mod, parameters to modify, from among cex.dot, cex.id, col.id, and height.id.

Details

dot.id and dot.mod work only with interactive graphics devices.

The dot.id works similarly to identify, but all dots in a particular stack are identified at once. The user should click along the bottom row of dots. No labels are displayed until the user exits identify mode (and then it may be necessary to refresh the plot by resizing it slightly). Also, the points that are labeled may change if the plot is resized horizontally, because only the x values are actually identified.

The user may call dot.id(..., modify = FALSE) more than once, if it is desired to change which values are identified.

After a call to dot.id or (especially) to dot.mod, the graph may need to be refreshed. There appears to be no standard way to do this, so the user may need to resize its window slightly.

Value

dot.plot returns an environment that is used to hold information on points that are identified. dot.id modifies this environment; it (invisibly) returns the currently identified x values. (These are values from the original call to dot.plot, not their rounded values used in the plot.)

Author(s)

Russell V. Lenth

Examples

```
require("unrepx")

educ = swiss$Education
names(educ) = abbreviate(row.names(swiss))
dp <- dot.plot(educ, xlab = "Percent Post-Primary Education (Switzerland, 1888)")

## Not run:
    dot.id(dp)

    dot.mod(dp, height.id = 3, cex.id = .5)

## End(Not run)
```

---

hnplot *Half-normal plots (Daniel plots) of effects*

Description

The hnplot function constructs a (half-) normal plot of effects (see Daniel 1959) that is traditionally used to identify active effects in a screening experiment. Reference lines and various other options and extensions are supported.

Usage

```
hnplot(effects, ref = TRUE, half = TRUE, horiz = TRUE, method = "Zahn",
       a = 0.375, col = half, pch = 16, ID = FALSE, alpha, ...)
```

Arguments

effects	Numeric vector of effects or contrasts to be explored.
ref	Logical value. If TRUE, a reference line is added to the plot determined by method. If FALSE, no reference is shown.
half	Logical value. If TRUE, a plot based on the absolute effects is constructed. If FALSE (not recommended: see Details), the original signed effects are used.
horiz	Logical value. If TRUE, the (absolute) effects are plotted on the horizontal scale and the (half-) normal scores or labels are plotted on the vertical scale. If FALSE, these axes are reversed.

method	Character value. When ref is TRUE, the method to use in determining the reference line, curve, and/or critical values. This must be the name of a provided pseudo-standard-error method (see PSE), or a compatible user-supplied one.
a	The adjustment used in scaling and centering ranks in the interval (0, 1). The ith ordered (half-) normal score is computed as the $(i - a)/(n + 1 - 2 * a)$ quantile of the reference distribution.
col	Scalar or vector of colors; or a logical value. If logical, a value of TRUE colors the positive effects blue, the negative effects red, and any zeros as black. A logical value of FALSE colors them all black.
pch	Plotting character(s) to use.
alpha	Numeric value. If specified, a null reference distribution for method is used (see ref.dist) to determine a margin of error (labeled 'ME') and simultaneous margin of error (labeled 'SME') corresponding to a significance level of alpha, and reference lines are added to the plot at those positions as an aid to assessing the statistical significance of the effects. This is based on a suggestion in Mee (2015). These reference lines are omitted when alpha is left unspecified.
ID	Logical or numeric value. If logical and TRUE, then after the plot is contructed, the plot is put in identify mode, where the user may click on points to be labeled on the plot. If a numeric value is supplied, it is used as a threshold by which all effects greater than ID[1] in absolute value are labeled.
...	Additional graphical parameters (see par) used in constructing the plot.

Details

Use of half = FALSE is not recommended because it can be misleading to the user. Inactive effects are those that are close to zero, and a regular normal plot displays deviations from normality rather than deviations from zero.

Author(s)

Russell V. Lenth

References

Daniel, C (1959) Use of Half-Normal Plots in Interpreting Factorial Two-Level Experiments. *Technometrics*, 1(4), 311-341

Mee, R (2015) Discussion: Better, not Fewer, Plots. *Journal of Quality Technology*, 47(2), 107-109

See Also

Other ways of assessing active effects include a dot plot with a reference curve (refplot), a pareto plot of effects (see parplot), and a tabular style of presenting effects and P values (see eff.test). For more information on methods, see PSE and ref.dist.

Examples

```
require("unrepx")

hnplot(pdEff, ID = ME(pdEff))
```

---

parplot *Pareto plot of effects*

Description

Constructs a bar plot of ordered effects, along with cutoff values for the margin of error (ME) and simultaneous margin of error (SME). Such a plot is suggested in Lenth (1989), but other methods may be used for obtaining the ME and SME.

Usage

```
parplot(effects, pareto = TRUE, absolute = TRUE, horiz = FALSE, col = absolute,
    critvals, method = "Zahn", alpha = .05, refdist, sim.opts,
    ylab = "Estimated effects", top = n.effects, cex.annot = 0.75, ...)
```

Arguments

`effects`	Numeric vector of effects or contrasts to be explored.
`pareto`	Logical value. If `TRUE`, the effects are plotted in decreasing order of their absolute values.
`absolute`	Logical value. If `TRUE`, the absolute effects are plotted. If `FALSE`, the original signed effects are used, so that there are potentially positive- and negative-going bars in the plot.
`horiz`	Logical value. If `TRUE`, the bars are horizontal, and if `FALSE`, they are vertical.
`col`	A logical value, or valid color code(s) or names(s). If logical, a value of `TRUE` shades the positive effects in light blue, and the negative effects in pink. A logical value of `FALSE` colors them all light gray.
`critvals`	Numeric value(s). If these are provided, the first two elements of `critvals` are used as the ME and SME respectively (on the absolute scale of the effects). When `critvals` is specified, `method`, `alpha`, and `refdist` are ignored.
`method`	Character value designating the method to use in determining the margins of error displayed in the plot when `critvals` is not given. This must be the name of a provided pseudo-standard-error method (see `PSE`), or a compatible user-supplied one.
`alpha`	Numeric value. A null reference distribution for `method` is used (see `ref.dist`) to determine a margin of error (labeled 'ME') and simultaneous margin of error (labeled 'SME') corresponding to a significance level of `alpha`, and reference lines are added to the plot at those positions as an aid to assessing the statistical significance of the effects.

refdist	A result of ref.dist. If given, it is used to obtain critical values, rather than running a new simulation of the null distribution. The user should be careful that refdist indeed matches method and the number of effects.
sim.opts	A list containing arguments nsets and/or save to pass to ref.dist in case a new reference distribution needs to be simulated. See also details below.
ylab	Character axis label (overrides the default).
top	Numeric value giving the number of effects to display (this may help make all the important effect names visible). When top is less than the number of effects (n.effects), this forces pareto = TRUE and only the largest top effects are displayed. When this happens, an annotation is added to the plot to help clarify that not all effects are displayed.
cex.annot	Character magnification for annotations
...	Additional graphical parameters (see par) used in constructing the plot.

Details

The cutoff values displayed in the plot are labeled "ME", the margin of error, and "SME", the simultaneous margin of error. If not specified using crtivals, they are obtained using the 1-alpha quantiles of the reference distribution of absolute pseudo-t ratios. ME is based on the distribution of $|t|$. SME is based on the distribution of the maximum $|t|$ for a whole set of null effects.

In determining cutoff values, parplot tries to avoid re-simulating the reference distribution. Specifically, if the global variable .Last.ref.dist exists, and its contents match the given method and number of effects, it is used as the reference distribution. Similarly, if refdist is supplied, it is used (without checking). If a suitable reference distribution is not found, then it is simulated via ref.dist, with any arguments from sim.opts added.

If critvals is supplied, the specified values are used as the ME and SME: no reference distribution is needed, and hence method, alpha, and refdist are ignored.

The plot is scaled so that the ME cutoff always shows. The SME cutoff will only be visible if an observed effect is near or exceeds that boundary. The numeric values of the ME and SME are also shown in an annotation in the plot.

Value

Invisibly, the vector of the ME and SME values.

Author(s)

Russell V. Lenth

References

Lenth, R (1989) Quick and Easy Analysis of Unrelicated Factorials. *Technometrics* 31(4), 469-473

See Also

For more details on PSEs and reference distributions, see PSE and ref.dist. Note that parplot produces in essence a graphical version of the information from eff.test, but the latter provides more resolution in terms of P values.

Other graphical ways of assessing active effects include a dot plot with a reference curve (refplot) and a half-normal plot (see hnplot).

Examples

```
require("unrepx")

parplot(pdEff, top = 10)
```

---

pdEff *Effect examples*

Description

Published or simulated examples of effects.

Usage

```
pdEff
bikeEff
viseEff
shnkEff
shnkDisp
```

Format

Each is a named numeric vector of effect estimates from unreplicated experiments. Also, each has an additional mean attribute containing the response mean.

Details

pdEff A vector of 15 effects from a four-factor experiment on process development. The experiment is described in Box, Hunter, and Hunter (2005), Section 5.13, and the effects are tabulated in Table 5.11, page 200. The response variable is conversion percent, and the experiment involves one replication each of each combination of four two-level factors: catalyst charge (C), temperature (T), pressure (P), and concentration (c). (The text labels these factors as 1, 2, 3, and 4 but we elected to use more suggestive alphabetic labels.) The effects are in standard (Yates) order.

bikeEff A vector of 7 effects from a saturated experiment in 8 runs with 7 two-level factors. The experiment is described in Box, Hunter, and Hunter (2005), Section 6.5 and the effects are reported in Table 6.5, page 245. The response variable is time required to climb a particular hill, and the factors are seat height, dynamo, gear, handlebars, raincoat, breakfast, and tires. (Effect labels are abbreviations of these.) The effects are in standard order of the first, second, and fourth factors.

viseEff A vector of 15 effect estimates from a fictitious experiment, simulated by the package developer. The effects are labeled alphabetically, A-O, in stanradr order of A, B, D, and H.

shnkEff, shnkDisp Vectors of location and dispersion effects, respectively, from the speedometer-cable shrinkage example discussed in Box, Hunter, and Hunter (2005), Section 6.14. Each vector is of length 15. The effect names are 4-letter abbreviations of the factor names. Effects are computed anew from the data in Table 6.18: `shnkEff` from the "Average" response and `shnkDisp` from the "Log Variance" response. The table in the book actually gives variances, not log variances, and logs were taken before dispersion effects were calculated. A few effect values differ somewhat from those in the book.

References

Box, GEP, Hunter, JS, and Hunter, WG (2005) *Statistics for Experimenters* (2nd ed) New York: John Wiley & Sons

Examples

```
require("unrepx")

parplot(bikeEff, method = "Zahn")

opar <- par(mfcol = c(1,2))
  hnplot(shnkEff, half = FALSE, main = "Normal plot")
  hnplot(shnkEff, half = TRUE, main = "Half-Normal plot")
#   Why ordinary normal plots are a bad idea
#   Both plots have the same reference line
par(opar)

# Note - Examples in help pages for hnplot, parplot, refplot, and eff.test
#        use pdEff for illustration

## Not run:
    # Do try this at home:
    hnplot(viseEff, ID = TRUE)
    refplot(viseEff, ID = TRUE)

## End(Not run)
```

PSE *Pseudo standard error of effects*

Description

Computes a pseudo standard error using any of a variety of built-in methods, or a user-supplied one.

Usage

```
PSE(effects, method = "Zahn", verbose = FALSE)

ME(effects, method = "Zahn", alpha = .05, ...)
```

Arguments

effects	Numeric vector of effects or contrasts to be explored.
method	Character value. The name of the method to be used. See Details.
verbose	Logical value. If TRUE, the parameters (if any) generated by the method's setup code are printed.
alpha	Numeric significance level, between 0 and 1.
...	Additional arguments passed to ref.dist. If a matching reference distribution is already available in .Last.ref.dist, these arguments have no effect.

Details

The PSE function implements methods of estimating the standard error of effects estimates from unreplicated designs. The underlying assumption is that the effects all have the same variance, and that "effect sparsity" assumption applies, whereby the majority of the effects are inactive and only a handful are active. The method may be any "directed" method (as described in Hamada and Balakrishnan (1998). A number of built-in methods are available; see the list below.

Users may easily write their own method. The method "foo" would be implemented by writing a function foo_pse <- function(effects) { ... } and saving it where it can be found in the search path. An example can be found by listing unrepx:::Lenth_pse.

If the user-supplied function needs to use weights, coefficients, or other parameters that depend on length(effects) that would be cumbersome in simulations (e.g., in ref.dist), the user may instead provide a function foo_pse <- function(effects, parm) { ... }, along with attr(foo_pse, "setup") <- function(n.effects) { ... } which returns the parm argument (say, a list) to be used when length(effects) = n.effects. The setup function is called automatically if the "setup" attribute exists, and if so, the function is expected to have the second argument. See a listing of unrepx:::Zahn_pse for an example.

Value

PSE returns a single numeric value, named in the style method_PSE. ME returns a named numeric vector of length 2, containing the margin of error ME and the simultaneous margin of error SME.

Built-in methods

Daniel The 68.3rd quantile of the absolute effects. See Daniel (1959).

Dong The RMS method, applied after excluding all effects that exceed 2.5 * PSE(effects, "SMedian") in absolute value. See Dong (19??).

JuanPena An iterated median method whereby we repeatedly calculate the median of the absolute effects that don't exceed 3.5 times the previous median, until it stabilizes. The estimate is the final median, divided by .6578. See Juan and Pena (1992).

Lenth The SMedian method, applied after excluding all effects that exceed 2.5 * PSE(effects, "SMedian") in absolute value. See Lenth (1989).

RMS Square root of the mean of the squared effects. This is not a good PSE in the presence of active effects, but it is provided for sake of comparisons.

SMedian 1.5 times the median of the absolute effects.

Zahn, WZahn The Zahn method is the slope of the least-squares line fitted to the first m points of `hnplot(effects, horiz = FALSE)`, where `m = floor(.683 * length(effects))`. (This line is fitted through the origin.) The WZahn method is an experimental version of Zahn's method, based on weighted least-squares with weights decreasing linearly from $m - .5$ to .5, but bounded above by $.65m$.

Author(s)

Russell V. Lenth

References

Daniel, C (1959) Use of Half-Normal Plots in Interpreting Factorial Two-Level Experiments. *Technometrics*, 1(4), 311-341

Dong, F (1993) On the Identification of Active Contrasts in Unreplicated Fractional Factorials. *Statistica Sinica* 3, 209-217

Hamada and Balakrishnan (1998) Analyzing Unreplicated Factorial Experiments: A Review With Some New Proposals. *Statistica Sinica* 8, 1-41

Juan, J and Pena, D (1992) A Simple Method to Identify Significant Effects in Unreplicated Two-Level Factorial Designs. *Communications in Statistics: Theory and Methods* 21, 1383-1403

Lenth, R (1989) Quick and Easy Analysis of Unrelicated Factorials *Technometrics* 31(4), 469-473

Zahn, D (1975) Modifications of and Revised Critical Values for the Half-Normal Plot. *Technometrics* 17(2), 189-200

Examples

```
require("unrepx")

PSE(shnkEff, method = "Lenth")
```

`ref.dist` *Analysis of effects from screening experiments*

Description

These functions facilitate making inferences based on effect estimates in an unreplicated experiment, withn an underlying effect-sparsity model.

Usage

```
ref.dist(method, n.effects, nsets, save = TRUE)

eff.test(effects, method = "Zahn", pareto = TRUE, refdist, save = TRUE)
```

Arguments

method	The method to use in determining the reference line, curve, and/or critical values. This must be the name of a provided pseudo-standard-error method (see PSE), or a compatible user-supplied one.
n.effects	Integer number of effects estimated.
nsets	The number of complete-null samples of size n.effects to be simulated. If omitted, nsets is determined so that the total number of simulated effects is about 40,000.
save	Logical value. If TRUE, the simulated reference distribution is saved in the workspace under .Last.ref.dist. Other routines in this package try to avoid re-simulating a reference distribution if .Last.refdist exists and matches the current method and n.effects.
effects	Vector of observed effects to be tested against the reference distribution.
pareto	Logical value. If TRUE, the effects are presented in decreasing order of their absolute size.
refdist	A result of a previous call to ref.dist, in case the user wishes to manually supply a previously simulated reference distribution. Note however that eff.test will automatically reuse .Last.ref.dist if it is available and matches.

Details

ref.dist simulates samples of effects from the standard normal dstribution. For each sample, the pseudo standaerd error (PSE) of the effects (typically some kind of outlier-resistant estimate of the SD) is obtained via a call to PSE with specified method. The absolute t values are obtained as ratios of the simulated effects and the PSE, as well as and the maxima of these absolute t values. Quantiles and tail areas of these simulated distributions then form a reference for obtaining critical values and P values in testing an observed sample of effects.

eff.test performs a traditional-style analysis for an observed sample of effects. It outputs the effects, PSE, t ratios; and uses tail areas of the associated reference distribution to compute individual and simultaneous P values. The simultaneous P values implement a multiplicity correction for *any* type-I errors occurring among the tests.

Value

ref.dist returns an object of class "eff_refdist" – structurally, a list with elements abst (the absolute values of the simulated t statistics), max.abst (the sample maxima of abst), and sig (a signature of the form method_n.effects). There is a print method for this class that displays a summary.

eff.test returns a data.frame containing the estimates, $tratios$, and estimated P values as tail areas of abst and max.abst from the reference distribution.

Author(s)

Russell V. Lenth

Examples

```
require("unrepx")

zahn15 <- ref.dist("Zahn", 15)
eff.test(pdEff, refdist = zahn15)
```

`refplot` *Dot plot of effects with a reference distribution*

Description

The `refplot` function constructs a dot plot of effects along with a rweference distribution (either normal or simulated) to help in visually identifying active effects in a screening experiment.

Usage

```
refplot(effects, ref = TRUE, half = TRUE, method = "Zahn",
    col = half, guides = FALSE, ID = FALSE, pch = 16, xlab, xlim, ...)
```

Arguments

`effects`	Numeric vector of effects or contrasts to be explored.
`ref`	Logical or character value. If `TRUE`, a reference line or curve is added to the plot determined by `method`. If `FALSE`, no reference is added to the plot. A character value matching `"normal"` or `"simulated"` is also permitted. `"normal"` is equivalent to `ref = TRUE`. With `"simulated"`, , a kernel density estimate is displayed; it is obtained by scaling the simulated reference distribution for `"method"` by its observed PSE.
`half`	Logical value. If `TRUE`, a dot plot of the absolute effects is constructed. If `FALSE`, the original signed effects are plotted.
`method`	Character value. When `ref` is not false, the method to use in determining the PSE for scaling the reference curve (and also the method used for simulating the referemnce distribution when `ref = "simulated"`). This must be the name of a provided pseudo-standard-error method (see `PSE`), or a compatible user-supplied one.
`col`	Scalar or vector of colors; or a logical value. If logical, a value of `TRUE` colors the positive effects blue, the negative effects red, and any zeros as black. A logical value of `FALSE` colors them all black.
`guides`	Logical value. If `TRUE`, dotted lines are added that illustrate guide lines that could be used to draw the normal curve by hand.
`ID`	Logical value. If logical and `TRUE`, then after the plot is contructed, the `dot.id` invoked so that the user may click on points to be labeled on the plot. If a numeric value is given, it is used as a threshold by which all effects greater than `ID[1]` in absolute value are labeled.
`pch, xlab, xlim, ...`	Additional graphical parameters (see `par`) passed to `dot.plot`.

Details

If the returned environment is saved, then dot.id or dot.mod may be used later as for dot.plot results.

Value

An environment that can be modified using dot.id or dot.mod.

Author(s)

Russell V. Lenth

See Also

Other ways of assessing active effects include a half-normal plot (hnplot), a Pareto plot of effects (see parplot), and a tabular style of presenting effects and P values (see eff.test). For more information on PSEs and methods, see PSE and ref.dist.

Examples

```
require(unrepx)

refplot(pdEff, ID = ME(pdEff))

## Not run:
    # Batman lives!
    refplot(pdEff, ref = "sim", method = "Lenth", half = FALSE)

## End(Not run)
```

---

yates *Yates's algorithm*

Description

Implementations of Yates's method for obtaining factor effects; and reverse Yates's method for recovering response values; and a generalization for balanced unreplicated experiments having other than two-level factors.

Usage

```
yates(y, labels = LETTERS, sep = "")
gyates(y, nlevels, basis = "poly")
```

Arguments

y	Numeric vector of response values or effects, in standard order. See Details.
labels	Character labels for 2-level factors to use when y contains resoponse values. The first $log_2(n)$ are used, where n is `length(y)`.
sep	Character to use between labels for interaction effects.
nlevels	Numeric vector of numbers of levels. The number of observations is assumed to be `prod(nlevels)`.
basis	Character value or vector specifying what method to use to generate orthonormal contrasts. Built-in ones include `"poly"` and `"helmert"`, but the user may create others. The elements of `basis` are recycled cyclically as needed.

Details

These functions implement a method for computing factor effects for balanced, unreplicated designs without need for a matrix of predictor levels. Instead, the responses must be arranged in standard order. The method is described in Yates (1937) for the case where all factors have two levels (as implemented in `yates`), and its generalization (`gyates`) can be found in Good (1958). Both are described in Drum (2005). In this implementation, `gyates()` uses orthonormal (unitary) matrices, rather than keeping track of needed divisors.

Standard order (also called Yates order) is that in which the first factor varies the fastest and the last varies the slowest. The `expand.grid` function creates factor combinations in standard order. The `yates` function is for experiments with all factors having two levels. The returned effects are also in standard order: with the default labels in `yates`, the order of the returned effects is (intercept), A, B, AB, C, AC, BC, ABC, D, etc.: note that absence or presence of each factor proceeds in standard order.

In both `yates` and `gyates`, if the length of `y` is one less than expected, `y` is assumed to be a set of effects, without the intercept. In that instance, the algorithm is reversed and the response values are recovered from the effects. Since the intercept is absent, the mean is arbitrary. If `y` has a `"mean"` attribute, the mean is adjusted to that value; otherwise, the recovered responses have a mean of zero.

In `gyates`, the values of `nlevels` and `basis` are saved as attributes. In a subsequent call on the returned effects, these values are used and will override the `nlevels` and `basis` arguments supplied by the user.

Effects are scaled to all have the same variance. In `yates`, we apply the same convention as in most design texts, e.g., Box et al. (2005): effect = (mean at "+" level) - (mean at "-" level), which is twice the regression coefficient one obtains by regressing `y` on predictors of -1s and +1s. In `gyates`, effects are computed using orthonormal contrasts, making the squared effects equal to their ANOVA sums of squares. When effects are reversed, the same scaling is assumed. Also, if the elements of `y` are independent, so are the effects.

The two basis functions supplied are `"poly"` and `"helmert"`, which are based on `contr.poly` and `contr.helmert` respectively. Users may create a custom basis, say `"foo"`, by writing a function `foo_gyb = function(k) {...}`, and supplying `basis = "foo"` in the call. The function should return a `k` by `k` matrix having its first column equal to `1/sqrt(k)` and orthonormal contrasts in the remaining columns.

Value

If `length(y)` is a power of two (in `yates`) or the product of `nlevels` (in `gyates`), a vector of `length(y) - 1` effects is returned (the intercept is omitted). In addition, the returned value has a `"mean"` attribute set equal to `mean(y)`. The effects from `yates` are named in standard order using the labels provided. The effects from `gyates` are labeled using patterns of the characters `.123456789`. Any `.` in a label indicates a factor whose effect is out of play. For illustration, with 3 factors, the effect named `.1.` is the first-order effect of the second factor, and the one named `23.` is the interaction of the second-order effect of the first factor and the third-order effect of the second factor.

If `length(y)` is one less than a power of two or the product of levels, then a vector one longer of response values is returned. The names of the returned responses are symbol combinations from `-+` for `yates`, and `123456789` from `gyates`. For example, in `yates` with `length(y) = 7`, the returned values are labeled `---`, `+--`, `-+-`, `++-`, `--+`, `+-+`, `-++`, `+++`; and `gyates` with 5 effects and `nlevels = c(3, 2)`, the returned effects are labeled `11`, `21`, `31`, `12`, `22`, `32`.

Author(s)

Russell V. Lenth

References

Box, GEP, Hunter, JS, and Hunter, WG (2005) *Statistics for Experimenters* (2nd ed) New York: John Wiley & Sons

Drum, M (2005) Yates's Algorithm. *Encyclopedia of Biostatistics*, 8. Wiley.

Good, IJ (1958) The interaction algorithm and practical Fourier analysis. *Journal of the Royal Statistical Society, Series B* 20, 361-372.

Yates, F (1937) The design and analysis of factorial experiments. Technical Communication of the Commonwealth Bureau of Soils, 35, Commonwealth Agricultural Bureau, Farnham Royal

Examples

```
require("unrepx")

# pilot-plant example, BH^2 p. 177
yates(c(60, 72, 54, 68,  52, 83, 45, 80), labels = c("T", "C", "K"))

# recover shrinkage data
yates(shnkEff)

# A 3 x 2 x 4 example
y <- c( 214, 193, 207,   193, 178, 188,
        225, 206, 213,   221, 214, 216,
        227, 213, 221,   231, 215, 225,
        228, 203, 206,   190, 178, 195 )
yeff <- gyates(y, c(3,2,4), basis = "helmert")
head(eff.test(yeff))  ## Show the largest few effects
```

Index