Package 'epitools'

May 8, 2026

Version 0.5-10.1

Date 2017-10-26

Title Epidemiology Tools

Depends R (>= 2.10)

Description Tools for training and practicing epidemiologists including methods for two-way and multi-way contingency tables.

License GPL (>= 2)

Author Tomas J. Aragon [aut],
Michael P. Fay [ctb],
Daniel Wollschlaeger [ctb],
Adam Omidpanah [cre, ctb]

Maintainer Adam Omidpanah <adam.omidpanah@wsu.edu>

Repository CRAN

Date/Publication 2020-03-22 09:46:12 UTC

NeedsCompilation no

Contents

`ageadjust.direct` *Age standardization by direct method, with exact confidence intervals*

Description

Calculates age standardized (adjusted) rates and "exact" confidence intervals using the direct method

Usage

```
ageadjust.direct(count, pop, rate = NULL, stdpop, conf.level = 0.95)
```

Arguments

`count`	vector of age-specific count of events
`pop`	vector of age-specific person-years or population estimates
`rate`	vector of age-specific rates
`stdpop`	vector of age-specific standarad population
`conf.level`	confidence level (default = 0.95)

Details

To make valid comparisons between rates from different groups (e.g., geographic area, ethnicity), one must often adjust for differences in age distribution to remove the confounding affect of age. When the number of events or rates are very small (as is often the case for local area studies), the normal approximation method of calculating confidence intervals may give a negative number for the lower confidence limit. To avoid this common pitfall, one can approximate exact confidence intervals. This function implements this method (Fay 1997).

Original function written by TJ Aragon, based on Anderson, 1998. Function re-written and improved by MP Fay, based on Fay 1998.

Value

crude.rate	crude (unadjusted) rate
adj.rate	age-adjusted rate
lci	lower confidence interval limit
uci	upper confidence interval limit

Author(s)

Michael P. Fay, <mfay@niaid.nih.gov>; Tomas Aragon, <aragon@berkeley.edu>, http://www.phdata.science

References

Fay MP, Feuer EJ. Confidence intervals for directly standardized rates: a method based on the gamma distribution. Stat Med. 1997 Apr 15;16(7):791-801. PMID: 9131766

Steve Selvin. Statistical Analysis of Epidemiologic Data (Monographs in Epidemiology and Biostatistics, V. 35), Oxford University Press; 3rd edition (May 1, 2004)

Anderson RN, Rosenberg HM. Age Standardization of Death Rates: Implementation of the Year 200 Standard. National Vital Statistics Reports; Vol 47 No. 3. Hyattsville, Maryland: National Center for Health Statistics. 1998, pp. 13-19. Available at http://www.cdc.gov/nchs/data/nvsr/nvsr47/nvs47_03.pdf.

See Also

See also ageadjust.indirect

Examples

```
## Data from Fleiss, 1981, p. 249
population <- c(230061, 329449, 114920, 39487, 14208, 3052,
72202, 326701, 208667, 83228, 28466, 5375, 15050, 175702,
207081, 117300, 45026, 8660, 2293, 68800, 132424, 98301,
46075, 9834, 327, 30666, 123419, 149919, 104088, 34392,
319933, 931318, 786511, 488235, 237863, 61313)
population <- matrix(population, 6, 6,
dimnames = list(c("Under 20", "20-24", "25-29", "30-34", "35-39",
"40 and over"), c("1", "2", "3", "4", "5+", "Total")))
population
count <- c(107, 141, 60, 40, 39, 25, 25, 150, 110, 84, 82, 39,
3, 71, 114, 103, 108, 75, 1, 26, 64, 89, 137, 96, 0, 8, 63, 112,
262, 295, 136, 396, 411, 428, 628, 530)
count <- matrix(count, 6, 6,
dimnames = list(c("Under 20", "20-24", "25-29", "30-34", "35-39",
"40 and over"), c("1", "2", "3", "4", "5+", "Total")))
count

### Use average population as standard
standard<-apply(population[,-6], 1, mean)
standard
```

```
### This recreates Table 1 of Fay and Feuer, 1997
birth.order1<-ageadjust.direct(count[,1],population[,1],stdpop=standard)
round(10^5*birth.order1,1)

birth.order2<-ageadjust.direct(count[,2],population[,2],stdpop=standard)
round(10^5*birth.order2,1)

birth.order3<-ageadjust.direct(count[,3],population[,3],stdpop=standard)
round(10^5*birth.order3,1)

birth.order4<-ageadjust.direct(count[,4],population[,4],stdpop=standard)
round(10^5*birth.order4,1)

birth.order5p<-ageadjust.direct(count[,5],population[,5],stdpop=standard)
round(10^5*birth.order5p,1)
```

`ageadjust.indirect` *Age standardization by indirect method, with exact confidence intervals*

Description

Calculates age standardized (adjusted) rates and "exact" confidence intervals using the indirect method

Usage

```
ageadjust.indirect(count, pop, stdcount, stdpop, stdrate = NULL,
conf.level = 0.95)
```

Arguments

`count`	vector of age-specific count of events
`pop`	vector of age-specific person-years or population estimates
`stdcount`	vector of age-specific standard counts
`stdpop`	vector of age-specific standarad population
`stdrate`	vector of age-specific standard rates
`conf.level`	confidence level (default = 0.95)

Details

To make valid comparisons between rates from different groups (e.g., geographic area, ethnicity), one must often adjust for differences in age distribution to remove the confounding affect of age. When the number of events or rates are very small (as is often the case for local area studies), the normal approximation method of calculating confidence intervals may give a negative number for the lower confidence limit. To avoid this common pitfall, one can approximate exact confidence intervals. This function implements this method (Anderson 1998).

Value

$sir	observed, expected, standardized incidence ratio, and confidence interval
$rate	crude.rate, adjusted rate, and confidence interval

Note

Visit https://repitools.wordpress.com/ for the latest

Author(s)

Tomas Aragon, <aragon@berkeley.edu>, http://www.phdata.science. Thanks to Giles Crane (<giles.crane@doh.state.nj.us>) for reporting error in 'ageadjust.indirect' function.

References

Anderson RN, Rosenberg HM. Age Standardization of Death Rates: Implementation of the Year 200 Standard. National Vital Statistics Reports; Vol 47 No. 3. Hyattsville, Maryland: National Center for Health Statistics. 1998, pp. 13-19. Available at http://www.cdc.gov/nchs/data/nvsr/nvsr47/nvs47_03.pdf.

Steve Selvin. Statistical Analysis of Epidemiologic Data (Monographs in Epidemiology and Biostatistics, V. 35), Oxford University Press; 3rd edition (May 1, 2004)

See Also

See also ageadjust.direct

Examples

```
##From Selvin (2004)
##enter data
dth60 <- c(141, 926, 1253, 1080, 1869, 4891, 14956, 30888,
41725, 26501, 5928)

pop60 <- c(1784033, 7065148, 15658730, 10482916, 9939972,
10563872, 9114202, 6850263, 4702482, 1874619, 330915)

dth40 <- c(45, 201, 320, 670, 1126, 3160, 9723, 17935,
22179, 13461, 2238)

pop40 <- c(906897, 3794573, 10003544, 10629526, 9465330,
8249558, 7294330, 5022499, 2920220, 1019504, 142532)

##calculate age-specific rates
rate60 <- dth60/pop60
rate40 <- dth40/pop40

#create array for display
tab <- array(c(dth60, pop60, round(rate60*100000,1), dth40, pop40,
round(rate40*100000,1)),c(11,3,2))
agelabs <- c("<1", "1-4", "5-14", "15-24", "25-34", "35-44", "45-54",
```

```
"55-64", "65-74", "75-84", "85+")
dimnames(tab) <- list(agelabs,c("Deaths", "Population", "Rate"),
c("1960", "1940"))
tab

##implement direct age standardization using 'ageadjust.direct'
dsr <- ageadjust.direct(count = dth40, pop = pop40, stdpop = pop60)
round(100000*dsr, 2) ##rate per 100,000 per year

##implement indirect age standardization using 'ageadjust.indirect'
isr <- ageadjust.indirect(count = dth40, pop = pop40,
                          stdcount = dth60, stdpop = pop60)
round(isr$sir, 2)         ##standarized incidence ratio
round(100000*isr$rate, 1) ##rate per 100,000 per year
```

`as.hour` *Convert date-time object into hour units*

Description

Convert date-time object into hour or half-hour units

Usage

```
as.hour(x, mindt, maxdt, half.hour = FALSE)
```

Arguments

`x`	Date-time object in standard format: for example, "2004-12-23 08:27:00", "2004-12-23 08:27", "2004-12-23"
`mindt`	[required] Date-time object in standard format that will form the lower boundary of the hour or half-hour time categories. `mindt` must less than or equal to the minimum value in `x`, and must be rounded off to the nearst hour for hour categories (e.g., HH:00:00) or rounded off to the nearest half-hour for half-hour categories (e.g., HH:30:00).
`maxdt`	[required] Date-time object in standard format that will form the upper boundary of the hour or half-hour time categories. `maxdt` must greater than or equal to the minimum value in `x`, and must be rounded off to the nearst hour for hour categories (e.g., HH:00:00) or rounded off to the nearest half-hour for half-hour categories (e.g., HH:30:00).
`half.hour`	Set to TRUE for half-hour categories.

Details

This function (1) converts standard date-time objects into 1-hour or 1/2-hour categories, and (2) generates levels for range of values that that the new 1-hour or 1/2-hour categories can take. These levels are use for converting x into a factor and for providing names for labeling the x-axis in plot. This function is used by `epicurves.hours`.

Value

$ct	Date-time object that contains the number of seconds since the beginning of 1970 as a numeric vector and produced by as.POSIXct. You can use as.POSIXlt to convert this output in human legible (already done by this function).
$sec	seconds
$min	minutes
$hour	hours (0-23)
$hour12	hours (1-12)
$stratum	number of hours or 1/2 hours since beginning of 1970
$stratum2	factor (categorical variable) with number of hours of 1/2 hours since beginning of 1970 using $cstratum as the levels
$stratum3	factor (categorical variable) in standard date-time format indicating number of hours or 1/2 hours since beginning of 1970 using $cstratum2 as the levels
$cstratum	levels for creating $stratum2 factor
$cstratum2	levels for creating $stratum3 factor
$csec	seconds from $cstratum2
$cmin	minutes from $cstratum2
$chour	hours from $cstratum2 in 24-hour format
$chour12	hours from $cstratum2 in 12-hour format
$campm	corresponding 'AM' or 'PM' for $chour12
$campm2	corresponding 'am' or 'pm' for $chour12
$cweekday	day of the week for $cstratum2
$cwkday	abbreviated day of the week for $cstratum2
$cmday	day of the month for $cstratum2
$cmonth	month for $cstratum2
$cmon	abbreviated month for $cstratum2
$cyear	year for $cstratum2
$half.hour	FALSE (default) for 1-hour categories; TRUE for 1/2-hour categories

Author(s)

Tomas Aragon, <aragon@berkeley.edu>, http://www.phdata.science

References

none

See Also

epitools: as.month, epicurve.dates

as.Date, strptime, DateTimeClasses

Examples

```
dates <- c("1/1/04", "1/2/04", "1/3/04", "1/4/04", "1/5/04",
"1/6/04", "1/7/04", "1/8/04", "1/9/04", "1/10/04", NA, "1/12/04",
"1/14/04", "3/5/04", "5/5/04", "7/6/04", "8/18/04", "12/13/05",
"1/5/05", "4/6/05", "7/23/05", "10/3/05")
aw <- as.week(dates, format = "%m/%d/%y")
aw

aw2 <- as.week(dates, format = "%m/%d/%y", sunday= FALSE)
aw2

aw3 <- as.week(dates, format = "%m/%d/%y", min.date="2003-01-01")
aw3
```

---

as.month *Convert dates into months of the year for plotting epidemic curves*

Description

Converts dates into months of the year (1-12); but also creates range of calendar months that can be used to plot an epidemic curve

Usage

```
as.month(x, format = "%Y-%m-%d",
         min.date, max.date, before = 31, after = 31,
         origin = as.Date("1970-01-01"), abbreviate = TRUE)
```

Arguments

x	character vector of dates
format	date format of x; default is of form "2004-08-10"
min.date	[optional] minimum calendar date for plotting x-axis of an epidemic curve; should be of the form of "2004-08-10"; if no date is specified, then several days are subtracted from the minimum date in x as specified by the before option
max.date	[optional] maximum calendar date for plotting x-axis of an epidemic curve plot; should be f the form of "2004-08-10"; if no date is specified, then several days are added to the maximum date in x as specified by the after option
before	if min.date is not specified, then these number of days are subtracted from the minimum date in x for plotting minimum calendar date for epidemic curve
after	if max.date is not specified, then these number of days are added to the maximum date in x for plotting maximum calendar date for epidemic curve
origin	allows user to specify an alternative origin for Julian dates that are generated by this function (default = "1970-01-01")
abbreviate	abbreviate month names to Jan, Feb, Mar, etc.; often used for labeling plots

Details

This function converts dates to months (1-12). In addition, a range of calendar months are generated that can be used to plot the x-axis of an epidemic curve.

Value

Returns a list of the following:

`$dates`	input dates are converted to standard calendar date format
`$mon`	month of the year (1-12)
`$month`	month of the year (Jan, Feb, Mar, ...)
`$stratum`	the Julian date for the mid-month day of the `$mon` value
`$stratum2`	the Julian date for the mid-month day of the `$mon` value converted to a factor with levels determined by the Julian dates (`$cstratum`)used to plot an epidemic curve
`$stratum3`	the mid-month day of the `$mon` value converted to standard calendar dates
`$cmon`	the month of the year (1-12) used for plotting the x-axis of the epidemic curve
`$cmonth`	the months (Jan, Feb, Mar, ...) for the calendar dates used for plotting the x-axis of an epidemic curve
`$cstratum`	the Julian date for the mid-month day of the `$cmonth` value used for plotting the x-axis of an epidemic curve
`$cstratum2`	the standard calendar date for the mid-month day of the `$cmonth` value used for plotting the x-axis of an epidemic curve
`$cmday`	the day of the mon (1-31) for the calendar dates used for plotting the x-axis of an epidemic curve
`$cyear`	the years (e.g., 1996, 2001, ...) for the calendar dates used for plotting the x-axis of the epidemic curve

Author(s)

Tomas Aragon, <aragon@berkeley.edu>, http://www.phdata.science

References

none

See Also

epitools: `as.week`, `epicurve.dates`

`as.Date`, `strptime`, `DateTimeClasses`

Examples

```
dates <- c("1/1/04", "1/2/04", "1/3/04", "1/4/04", "1/5/04", "1/6/04",
"1/7/04", "1/8/04", "1/9/04", "1/10/04", NA, "1/12/04", "1/14/04",
"3/5/04", "5/5/04", "7/6/04", "8/18/04", "12/13/05", "1/5/05",
"4/6/05", "7/23/05", "10/3/05")
aw <- as.month(dates, format = "%m/%d/%y")
aw

aw2 <- as.month(dates, format = "%m/%d/%y", min.date="2003-01-01")
aw2
```

---

as.week *Convert dates object in 'disease week' for plotting epidemic curves*

Description

Convert dates into "disease week" with values of 1 to 53 for plotting epidemic curves

Usage

```
as.week(x, format = "%Y-%m-%d",
        min.date, max.date, before = 7, after = 7,
        origin = as.Date("1970-01-01"), sunday = TRUE)
```

Arguments

x	character vector of dates
format	date format of x; default is of form "2004-08-10"
min.date	[optional] minimum calendar date for plotting x-axis of an epidemic curve; should be of the form of "2004-08-10"; if no date is specified, then several days are subtracted from the minimum date in x as specified by the before option
max.date	[optional] maximum calendar date for plotting x-axis of an epidemic curve plot; should be f the form of "2004-08-10"; if no date is specified, then several days are added to the maximum date in x as specified by the after option
before	if min.date is not specified, then these number of days are subtracted from the minimum date in x for plotting minimum calendar date for epidemic curve
after	if max.date is not specified, then these number of days are added to the maximum date in x for plotting maximum calendar date for epidemic curve
origin	allows user to specify an alternative origin for Julian dates that are generated by this function (default = "1970-01-01")
sunday	First day of the week is Sunday (default = TRUE); setting to FALSE makes Monday the first day of the week

Details

In public health, reportable diseases are often reported by 'disease week' (either week of reporting or week of symptom onset). In R, weeks are numbered from 0 to 53 in the same year. The first day of week 1 starts with either the first Sunday or Monday of the year. Days before week 1 are numbered as 0s.

In contrast to R, the `as.week` function generates weeks numbered from 1 to 53. The week before week 1 takes on the value (52 or 53) from the last week of the previous year. The `as.week` functions facilitates working with multiple years and generating epidemic curves.

Value

Returns a list of the following:

`$dates`	input dates are converted to standard calendar date format
`$firstday`	first day of the week is reported
`$week`	week of the year (1-53); note that week 52 or 53 can represent both last week of a year but also the first few days at the beginning of the year
`$stratum`	the Julian date for the mid-week day of the `$week` value
`$stratum2`	the Julian date for the mid-week day of the `$week` value converted to a factor with levels determined by the Julian dates (`$cstratum`) used to plot the epidemic curve
`$stratum3`	the mid-week day of the `$week` value converted to standard calendar dates
`$cweek`	the week of the year used for plotting the x-axis of an epidemic curve
`$cstratum`	the Julian date for the mid-week day of the `$cweek` value used for plotting the x-axis of an epidemic curve
`$cstratum2`	the standard calendar date for the mid-week day of the `$cweek` value used for plotting the x-axis of an epidemic curve
`$cmday`	the day of the mon (1-31) for the calendar dates used for plotting the x-axis of an epidemic curve
`$cmonth`	the months (Jan, Feb, Mar, ...) for the calendar dates used for plotting the x-axis of an epidemic curve
`$cyear`	the years (e.g., 1996, 2001, ...) for the calendar dates used for plotting the x-axis of an epidemic curve

Author(s)

Tomas Aragon, <aragon@berkeley.edu>, http://www.phdata.science

References

none

See Also

epitools: `as.month`, `epicurve.dates`

`as.Date`, `strptime`, `DateTimeClasses`

Examples

```
dates <- c("1/1/04", "1/2/04", "1/3/04", "1/4/04", "1/5/04",
"1/6/04", "1/7/04", "1/8/04", "1/9/04", "1/10/04", NA, "1/12/04",
"1/14/04", "3/5/04", "5/5/04", "7/6/04", "8/18/04", "12/13/05",
"1/5/05", "4/6/05", "7/23/05", "10/3/05")
aw <- as.week(dates, format = "%m/%d/%y")
aw

aw2 <- as.week(dates, format = "%m/%d/%y", sunday= FALSE)
aw2

aw3 <- as.week(dates, format = "%m/%d/%y", min.date="2003-01-01")
aw3
```

---

binom.conf.int *Confidence intervals for binomial counts or proportions*

Description

Calculates confidence intervals for binomial counts or proportions

Usage

```
binom.exact(x, n, conf.level = 0.95)
binom.wilson(x, n, conf.level = 0.95)
binom.approx(x, n, conf.level = 0.95)
```

Arguments

x	number of successes in n trials, can be a vector
n	number of Bernoulli trials, can be a vector
conf.level	confidence level (default = 0.95), can be a vector

Details

The function, binom.exact, calculates exact confidence intervals for binomial counts or proportions. This function uses R's binom.test function; however, the arguments to this function can be numeric vectors of any length.

The function, binom.wilson, calculates confidence intervals for binomial counts or proportions using Wilson's formula which approximate the exact method. The arguments to this function can be numeric vectors of any length (Rothman).

The function, binom.approx, calculates confidence intervals for binomial counts or proportions using a normal approximation to the binomial distribution. The arguments to this function can be numeric vectors of any length.

Value

This function returns a n x 6 matrix with the following colnames:

x	number of successes in n trials
n	number of Bernoulli trials
prop	proportion = x/n
lower	lower confidence interval limit
upper	upper confidence interval limit
conf.level	confidence level

Author(s)

Tomas Aragon, <aragon@berkeley.edu>, http://www.phdata.science

References

Tomas Aragon, et al. Applied Epidemiology Using R. Available at http://www.phdata.science

Kenneth Rothman (2002), Epidemiology: An Introduction, Oxford University Press, 1st Edition.

See Also

pois.exact, binom.test

Examples

```
binom.exact(1:10, seq(10, 100, 10))
binom.wilson(1:10, seq(10, 100, 10))
binom.approx(1:10, seq(10, 100, 10))
```

---

colorbrewer *Display and create ColorBrewer palettes*

Description

Display and create ColorBrewer palettes based on Cindy Brewer's website at www.colorbrewer.org.

Usage

```
colorbrewer.display(nclass = 5,
                    type = c("qualitative", "sequential", "diverging"),
                    col.bg = "white")
colorbrewer.palette(nclass = 5,
                    type = c("qualitative", "sequential", "diverging"),
                    palette = letters[1:18])
colorbrewer.data()
```

Arguments

nclass	number of classes or categories to be compared graphically
type	select either 'qualitative' (default), 'sequential', or 'diverging'
col.bg	set background color (default is white)
palette	select palette (letter) from displayed plot

Details

These R functions includes color specifications and designs developed by Cynthia Brewer (http://www.colorbrewer.org). For more details on color selection please visit this excellent site.

First, select the number of classes or categories to be compared (nclass). Second, select the type of comparison (qualitative vs. sequential vs. diverging). Third, use colorbrewer.display to display the available ColorBrewer palette for a given type and number of classes. Fourth, using the colorbrewer.palette function, create a color palette for use in R graphics functions (e.g, col = mypal, where mypal was created from colorbrewer.palette).

Note that you can change the background color.

These functions for epitools were created to make the ColorBrewer palettes readily available to epitools users, and to have the same 3-step selection order as the www.colorbrewer.org site. A more visually appealing display of the ColorBrewer schemes is available in the RColorBrewer package.

Value

colorbrewer.display displays ColorBrewer selection and invisibly returns data that corresponds to graphical display

colorbrewer.palette returns rgb vector palette

Author(s)

Tomas Aragon, <aragon@berkeley.edu>, http://www.phdata.science

References

ColorBrewer, by Cynthia Brewer, Pennsylvanis State University, <cbrewer@psu.edu>, http://www.colorbrewer.org accessed on 2004-11-26

See Also

epitools package: colors.plot

Examples

```
##display available palettes for given nclass and type
colorbrewer.display(9, "sequential")

##change background to blue
colorbrewer.display(9, "sequential", "blue")

##display available palettes for given nclass and type,
##but also display RGB numbers to create your own palette
cbrewer.9s <- colorbrewer.display(9, "sequential")
cbrewer.9s


##Display and use ColorBrewer palette
##first, display and choose palette (letter)
colorbrewer.palette(10, "q")

##second, extract and use ColorBrewer palette
mycolors <- colorbrewer.palette(nclass = 10, type = "q", palette = "b")
xx <- 1:10
yy <- outer(1:10, 1:10, "*")
matplot(xx,yy, type="l", col = mycolors, lty = 1, lwd = 4)
```

---

colors.plot *Plots R's 657 named colors for selection*

Description

Plots R's 657 named colors for selection

Usage

```
colors.plot(locator = FALSE, cex.axis = 0.7)
colors.matrix()
```

Arguments

`colors.plot`:

`locator`	activates 'locator' for interactive selection of color names (default is FALSE)
`cex.axis`	change size of axes labels

`colors.matrix` has no arguments.

Details

The `colors.plot` function plots R's 657 named colors. If `locator=TRUE` then you can interactively point and click to select the colors for which you want names. To end selection, right click on the mouse and select 'Stop', then R returns the selected color names.

The `colors.matrix` function is used by `colors.plot` to create the matrix of color names that corresponds to the graph created by `colors.plot`. `colors.matrix` can be used alone to create the matrix of name without generating a plot. To see the matrix it must be assigned an object name and then displayed.

Value

`colors.plot` generates plot with R colors and, when `locator=TRUE`, returns matrix with graph coordinates and names of colors selected

`colors.matrix` quietly returns matrix of names

Author(s)

Tomas Aragon, <aragon@berkeley.edu>, http://www.phdata.science

References

none

See Also

`colorbrewer.display`, `colorbrewer.palette`, `colorbrewer.data`

`colors`

Examples

```
##creates matrix with color names
cm <- colors.matrix()
cm[1:3, 1:3]

##generates plot
colors.plot()

##generates plot and activates 'locator'
##don't run
##colors.plot(TRUE)
```

epicurve *Construct an epidemic curve*

Description

Construct an epidemic curve

Usage

```
epicurve.dates(x, format = "%Y-%m-%d", strata = NULL,
               min.date, max.date, before = 7, after = 7,
               width = 1, space = 0, tick = TRUE,
               tick.offset = 0.5, segments = FALSE, ...)

epicurve.weeks(x, format = "%Y-%m-%d", strata = NULL,
               min.date, max.date, before = 7, after = 7,
               width = 1, space = 0, tick = TRUE,
               tick.offset = 0.5, segments = FALSE,
               origin = as.Date("1970-01-01"), sunday = TRUE, ...)

epicurve.months(x, format = "%Y-%m-%d", strata = NULL,
                min.date, max.date, before = 31, after = 31,
                width = 1, space = 0, tick = TRUE,
                tick.offset = 0.5, segments = FALSE,
                origin = as.Date("1970-01-01"), ...)

epicurve.hours(x, mindt, maxdt, strata = NULL, half.hour = FALSE,
               width = 1, space = 0, tick = TRUE,
               tick.offset = 0.5, segments = FALSE, ...)

epicurve.table(x, width = 1, space = 0, tick = TRUE,
               tick.offset = 0.5, segments = FALSE, ...)
```

Arguments

x	character vector of dates
format	date format of x; default is of form "2004-08-10"
strata	[optional] categorical vector (character or factor) for stratifying x
min.date	[optional] minimum calendar date for plotting x-axis of an epidemic curve; should be of the form of "2004-08-10"; if no date is specified, then several days are subtracted from the minimum date in x as specified by the before option
max.date	[optional] maximum calendar date for plotting x-axis of an epidemic curve; should be of the form of "2004-08-10"; if no date is specified, then several days are added to the maximum date in x as specified by the after option

before	if min.date is not specified, then these number of days are subtracted from the minimum date in x for plotting minimum calendar date for epidemic curve
after	if max.date is not specified, then these number of days are added to the maximum date in x for plotting maximum calendar date for epidemic curve
mindt	[required] Date-time object in standard format that will form the lower boundary of the hour or half-hour time categories. The mindt option must less than or equal to the minimum value in x, and must be rounded off to the nearst hour for hour categories (e.g., HH:00:00) or rounded off to the nearest half-hour for half-hour categories (e.g., HH:30:00).
maxdt	[required] Date-time object in standard format that will form the upper boundary of the hour or half-hour time categories. The maxdt option must greater than or equal to the minimum value in x, and must be rounded off to the nearst hour for hour categories (e.g., HH:00:00) or rounded off to the nearest half-hour for half-hour categories (e.g., HH:30:00).
half.hour	Set to TRUE for half-hour categories in epicurve.hours.
width	width of bars in the epidemic curve; this value is passed to barplot function
space	space between bars in the epidemic curve; this value is passed to barplot function
tick	adds tick marks to the x-axis (default = TRUE)
tick.offset	offsets tick marks so that they plotted between the bars
segments	segments bars so that each box represents one case
origin	allows user to specify an alternative origin for Julian dates that are generated by this function (default = "1970-01-01")
sunday	First day of the week is Sunday (default = TRUE); setting to FALSE makes Monday the first day of the week
...	options are passed to the barplot function

Details

These functions makes plotting epidemic curves much easier in R. Normally, to plot an epidemic curve in R, one must do the following: (1) have disease onset dates in some calendar date format, (2) convert these onset dates into a factor with the levels specified by the range of calendar dates for the x-axis of the epidemic curve, (3) convert this factor into a table (with or without stratification), (4) use this table as an argument in the barplot function to plot the epidemic curve, and (5) make final adjustments (labels, titles, etc.).

Why use the barplot function? Strictly speaking, an epidemic curve is a histogram displaying the distribution of onset times which are categorized into, for example, dates. However, histogram functions seems to work better for measurements that our continuous (e.g., height, weight). In contrast, epidemic curves are constructed from onset time data that has been categorized into days, weeks, or months. For this type of categorical data, the barplot does a better job. The caveat, however, is that we need to specify the range of possible calendar dates, weeks, or months in order to construct an appropriate plot. To do this we convert the data into a factor with the levels specified by the possible calendar date values.

To make this whole process much easier, and to generate additional data that can be use for labeling your epidemic curve, the epicurve functions were created.

Value

`epicurve.dates`	returns list:
`$dates`	input dates are converted to standard calendar date format
`$dates2`	input dates are also converted to a factor with levels determined by the calendar dates (`$cdates`) used to plot the epidemic curve
`$xvals`	x-axis numeric values used for plotting the epidemic curve; this comes from the `barplot` function
`$cdates`	the calendar dates used for plotting the epidemic curve
`$cmday`	the day of the mon (1-31) for the calendar dates used for plotting the x-axis of the epidemic curve
`$cmonth`	the months (Jan, Feb, Mar, ...) for the calendar dates used for plotting the x-axis of the epidemic curve
`$cyear`	the years (e.g., 1996, 2001, ...) for the calendar dates used for plotting the x-axis of the epidemic curve
`epicurve.weeks`	returns list:
`$dates`	input dates are converted to standard calendar date format
`$firstday`	first day of the week is reported
`$week`	week of the year (1-53); note that week 52 or 53 can represent both last week of a year but also the first few days at the beginning of the year
`$stratum`	the Julian date for the mid-week day of the `$week` value
`$stratum2`	the Julian date for the mid-week day of the `$week` value converted to a factor with levels determined by the Julian dates (`$cstratum`) used to plot the epidemic curve
`$stratum3`	the mid-week day of the `$week` value converted to standard calendar dates
`$xvals`	x-axis numeric values used for plotting the epidemic curve; this comes from the `barplot` function
`$cweek`	the week of the year used for plotting the x-axis of the epidemic curve
`$cstratum`	the Julian date for the mid-week day of the `$cweek` value used for plotting the x-axis of the epidemic curve
`$cstratum2`	the standard calendar date for the mid-week day of the `$cweek` value used for plotting the x-axis of the epidemic curve
`$cmday`	the day of the mon (1-31) for the calendar dates used for plotting the x-axis of the epidemic curve
`$cmonth`	the months (Jan, Feb, Mar, ...) for the calendar dates used for plotting the x-axis of the epidemic curve
`$cyear`	the years (e.g., 1996, 2001, ...) for the calendar dates used for plotting the x-axis of the epidemic curve
`epicurve.months`	returns list:
`$dates`	input dates are converted to standard calendar date format
`$mon`	month of the year (1-12)

$month	month of the year (Jan, Feb, Mar, ...)
$stratum	the Julian date for the mid-month day of the $mon value
$stratum2	the Julian date for the mid-month day of the $mon value converted to a factor with levels determined by the Julian dates ($cstratum)used to plot the epidemic curve
$stratum3	the mid-month day of the $mon value converted to standard calendar dates
$xvals	x-axis numeric values used for plotting the epidemic curve; this comes from the barplot function
$cmon	the month of the year (1-12) used for plotting the x-axis of the epidemic curve
$cmonth	the months (Jan, Feb, Mar, ...) for the calendar dates used for plotting the x-axis of the epidemic curve
$cstratum	the Julian date for the mid-month day of the $cmonth value used for plotting the x-axis of the epidemic curve
$cstratum2	the standard calendar date for the mid-month day of the $cmonth value used for plotting the x-axis of the epidemic curve
$cmday	the day of the mon (1-31) for the calendar dates used for plotting the x-axis of the epidemic curve
$cyear	the years (e.g., 1996, 2001, ...) for the calendar dates used for plotting the x-axis of the epidemic curve
epicurve.hours	returns list:
$ct	Date-time object that contains the number of seconds since the beginning of 1970 as a numeric vector and produced by as.POSIXct. You can use as.POSIXlt to convert this output in human legible (already done by this function).
$sec	seconds
$min	minutes
$hour	hours (0-23)
$hour12	hours (1-12)
$stratum	number of hours or 1/2 hours since beginning of 1970
$stratum2	factor (categorical variable) with number of hours of 1/2 hours since beginning of 1970 using $cstratum as the levels
$stratum3	factor (categorical variable) in standard date-time format indicating number of hours or 1/2 hours since beginning of 1970 using
$xvals	
$cstratum	levels for creating $stratum2 factor
$cstratum2	levels for creating $stratum3 factor
$csec	seconds from $cstratum2
$cmin	minutes from $cstratum2
$chour	hours from $cstratum2 in 24-hour format
$chour12	hours from $cstratum2 in 12-hour format
$campm	corresponding 'AM' or 'PM' for $chour12

$campm2	corresponding 'am' or 'pm' for $chour12
$cweekday	day of the week for $cstratum2
$cwkday	abbreviated day of the week for $cstratum2
$cmday	day of the month for $cstratum2
$cmonth	month for $cstratum2
$cmon	abbreviated month for $cstratum2
$cyear	year for $cstratum2
$half.hour	FALSE (default) for 1-hour categories; TRUE for 1/2-hour categories
epicurve.table	returns numeric vector:
xvals	x-axis numeric values used for plotting the epidemic curve; this comes from the barplot function

Author(s)

Tomas Aragon, <aragon@berkeley.edu>, http://www.phdata.science

References

none

See Also

barplot, strptime

Examples

```
##epicurve.dates
sampdates <- seq(as.Date("2004-07-15"), as.Date("2004-09-15"), 1)
x <- sample(sampdates, 100, rep=TRUE)
xs <- sample(c("Male","Female"), 100, rep=TRUE)
epicurve.dates(x)
epicurve.dates(x, strata = xs)
rr <- epicurve.dates(x, strata = xs, segments = TRUE,
                     axisnames = FALSE)
axis(1, at = rr$xvals, labels = rr$cmday, tick = FALSE, line = 0)
axis(1, at = rr$xvals, labels = rr$cmonth, tick = FALSE, line = 1)

##epicurve.weeks
sampdates <- seq(as.Date("2004-07-15"), as.Date("2004-09-15"), 1)
x <- sample(sampdates, 100, rep=TRUE)
xs <- sample(c("Male","Female"), 100, rep=TRUE)
epicurve.weeks(x)

epicurve.weeks(x, strata = xs)

rr <- epicurve.weeks(x, strata = xs, segments = TRUE)
rr
```

```
##epicurve.months
dates <- c("1/1/04", "1/2/04", "1/3/04", "1/4/04", "1/5/04",
"1/6/04", "1/7/04", "1/8/04", "1/9/04", "1/10/04", NA, "1/12/04",
"1/14/04", "3/5/04", "5/5/04", "7/6/04", "8/18/04", "12/13/05",
"1/5/05", "4/6/05", "7/23/05", "10/3/05")
aw <- as.month(dates, format = "%m/%d/%y")
aw
aw2 <- as.month(dates, format = "%m/%d/%y", min.date="2003-01-01")
aw2

##epicurve.hours
data(oswego)
## create vector with meal date and time
mdt <- paste("4/18/1940", oswego$meal.time)
mdt[1:10]
## convert into standard date and time
meal.dt <- strptime(mdt, "%m/%d/%Y %I:%M %p")
meal.dt[1:10]
## create vector with onset date and time
odt <- paste(paste(oswego$onset.date,"/1940",sep=""), oswego$onset.time)
odt[1:10]
## convert into standard date and time
onset.dt <- strptime(odt, "%m/%d/%Y %I:%M %p")
onset.dt[1:10]

##set colors
col3seq.d <- c("#43A2CA", "#A8DDB5", "#E0F3DB")

par.fin <- par()$fin
par(fin=c(5,3.4))

##1-hour categories
xv <- epicurve.hours(onset.dt, "1940-04-18 12:00:00", "1940-04-19 12:00:00",
                     axisnames = FALSE, axes = FALSE, ylim = c(0,11),
                     col = col3seq.d[1], segments =  TRUE,
                     strata = oswego$sex)

hh <- xv$chour12==3 | xv$chour12== 6 | xv$chour12== 9
hh2 <- xv$chour12==12
hh3 <- xv$chour12==1
hlab <- paste(xv$chour12,xv$campm2,sep="")
hlab2 <- paste(xv$cmonth,xv$cmday)
axis(1, at = xv$xval[hh], labels = xv$chour12[hh], tick = FALSE, line = -.2)
axis(1, at = xv$xval[hh2], labels = hlab[hh2], tick = FALSE, line = -.2)
axis(1, at = xv$xval[hh3], labels = hlab2[hh3], tick = FALSE, line = 1.0)
axis(2, las = 1)
title(main = "Figure 1. Cases of Gastrointestinal Illness
by Time of Onset of Symptoms (Hour Category)
Oswego County, New York, April 18-19, 2004",
      xlab = "Time of Onset",
      ylab = "Cases")
```

```
##1/2-hour categories
xv <- epicurve.hours(onset.dt, "1940-04-18 12:00:00", "1940-04-19 12:00:00",
                     axisnames = FALSE, axes = FALSE, ylim = c(0,11),
                     col = col3seq.d[1], segments =  TRUE,
                     half.hour = TRUE, strata = oswego$sex)
hh <- xv$chour12==3 | xv$chour12== 6 | xv$chour12== 9
hh2 <- xv$chour12==12
hh3 <- xv$chour12==1
hlab <- paste(xv$chour12,xv$campm2,sep="")
hlab2 <- paste(xv$cmonth,xv$cmday)
axis(1, at = xv$xval[hh], labels = xv$chour12[hh], tick = FALSE, line = -.2)
axis(1, at = xv$xval[hh2], labels = hlab[hh2], tick = FALSE, line = -.2)
axis(1, at = xv$xval[hh3], labels = hlab2[hh3], tick = FALSE, line = 1.0)
axis(2, las = 1)
title(main = "Figure 2. Cases of Gastrointestinal Illness
by Time of Onset of Symptoms (1/2 Hour Category)
Oswego County, New York, April 18-19, 2004",
      xlab = "Time of Onset",
      ylab = "Cases")

par(fin=par.fin)


##epicurve.table
xvec <- c(1,2,3,4,5,4,3,2,1)
epicurve.table(xvec)

names(xvec) <- 1991:1999
epicurve.table(xvec)

xmtx <- rbind(xvec, xvec)
rownames(xmtx) <- c("Male", "Female")
epicurve.table(xmtx)

epicurve.table(xmtx, seg = TRUE)
```

`epidate` *Convert dates into multiple legible formats*

Description

Convert character vector of dates into multiple legible formats.

Usage

```
epidate(x, format = "%m/%d/%Y", cal.dates = FALSE,
        before = 7, after = 7, sunday = TRUE)
```

Arguments

x	character vector of dates to be converted
format	format of character vector of dates
cal.dates	Calendar dates that contains x, starting 7 days 'before' (default) until 7 days 'after' x
before	defines lower limit of cal.dates: default is 7 days before earliest date in x
after	defines upper limit of cal.dates: default is 7 days after latest date in x
sunday	first day of the week is either Sunday (default) or Monday

Details

Dates can come in many formats (e.g., November 12, 2001, 12Nov01, 11/12/2001, 11/12/01, 2001-11-12) and need to be converted into other formats for data analysis, graphical displays, generating reports, etc.

There is tremendous flexibility in converting any character vector with sufficient information to be converted into a unique date. For complete options for the format option see strptime.

Value

dates	dates wtih date-time class
julian	number of days since 1970-01-01
mday	day of the month: 1-31
mon	month of the year: 0-11
month	month: January, February, March, ...
month2	month: Jan, Feb, Mar, ...
firstday	first day of the week: Sunday or Monday
week	week of the year: 0-53
year	year: YYYY
yr	year: YY
wday	day of the week: 0-6
weekday	weekday: Monday, Tuesday, Wednesday, ...
wkday	weekday: Mon, Tue, Wed, ...
yday	day of the year: 0-365
quarter	quarter of the year: Q1, Q2, Q3, Q4
cdates	Calendar dates that contains dates
cjulian	Julian calendar dates

Author(s)

Tomas Aragon, <aragon@berkeley.edu>, http://www.phdata.science

References

none

See Also

epitools: as.week

DateTimeClasses to learn about date-time classes

format.Date to convert character vector of dates into calendar dates with date-time class (done by epidate)

strptime to convert date-time character strings into a date-time class

Examples

```
#x <- c("12/1/03", "11/2/03", NA, "1/7/04", "1/14/04", "8/18/04")
#epidate(x, format = "%m/%d/%y")
#epidate(x, format = "%m/%d/%y", TRUE)
#
###convert vector of disease weeks into vector of mid-week dates
#dwk <- sample(0:53, 100, replace = TRUE)
#wk2date <- paste(dwk, "/", "Wed", sep="")
#wk2date[1:10]
#wk2date2 <- epidate(wk2date, format = "%U/%a")
#wk2date2$dates[1:20]
```

---

epitab *Epidemiologic tabulation for a cohort or case-control study*

Description

Calculates risks, risk ratio, odds ratio, and confidence intervals for epidemiologic data

Usage

```
epitab(x, y = NULL,
       method = c("oddsratio", "riskratio", "rateratio"),
       conf.level = 0.95,
       rev = c("neither", "rows", "columns", "both"),
       oddsratio = c("wald", "fisher", "midp", "small"),
       riskratio = c("wald", "boot", "small"),
       rateratio = c("wald", "midp"),
       pvalue = c("fisher.exact", "midp.exact", "chi2"),
       correction = FALSE,
       verbose = FALSE)
```

Arguments

x	For odds ratio or risk ratio, input data can be one of the following: r x 2 table, vector of numbers from a contigency table (will be transformed into r x 2 table in row-wise order), or single factor or character vector that will be combined with y into a table. For rate ratio, input data can be one of the following: r x 2 table where first column contains disease counts and second column contains person time at risk; a single numeric vector of counts followed by person time at risk; a single numeric vector of counts combined with y which would be a numeric vector of corresponding person time at risk
y	For odds ratio or risk ratio, a single factor or character vector that will be combined with x into a table (default is NULL) For rate ratio, a numeric vector of person-time at risk; if provided, x must be a numeric vector of disease counts
method	select measure of association: "oddsratio" (default), "riskratio", or "rateratio"
conf.level	confidence level (default is 0.95)
rev	reverse order of "rows", "colums", "both", or "neither" (default)
oddsratio	selection estimation method: "wald" (default), "fisher", "midp", "small"
riskratio	selection estimation method: "wald" (default), "boot", "small"
rateratio	"wald" (default), "midp"
pvalue	"fisher.exact" (default), "midp.exact", "chi2" (normal approximation); for rate ratio, "fisher.exact" not calculated
correction	set to TRUE for Yate's continuity correction (default is FALSE)
verbose	set to TRUE to return more detailed results (default is FALSE)

Details

The epitab calculates odds ratios, risk ratios, or rate ratios for rx2 tables. The odds ratios are estimated using unconditional maximum likelihood (Wald), conditional maximum likelihood (Fisher), median-unbiased method (mid-p), or small-sample adjusted. The confidence intervals are estimated using a normal approximation (Wald), hypergeometric exact (Fisher), mid-p exact, or small sample adjusted method.

The risk ratios are estimated using unconditional maximum likelihood (Wald), or small-sample adjusted. The confidence intervals are estimated using a normal approximation (Wald), or bootstrap estimation.

The rate ratios are estimated using unconditional maximum likelihood estimation (Wald), or median unbiased method (mid-p). The confidence intervals are estimated using normal approximation, or mid-p exact method.

Notice the expected structure of the data to be given to 'epitab':

```
               Disease
 Exposure      No (ref)  Yes
  Level 1 (ref)  a         b
  Level 2        c         d
```

```
Level 3        e         f
```

This function expects the following table struture for rate ratios:

```
                  counts   person-time
exposed=0 (ref)   n00        t01
exposed=1         n10        t11
exposed=2         n20        t21
exposed=3         n30        t31
```

If the table you want to provide to this function is not in the preferred form, just use the `rev` option to "reverse" the rows, columns, or both. If you are providing categorical variables (factors or character vectors), the first level of the "exposure" variable is treated as the reference. However, you can set the reference of a factor using the `relevel` function.

Likewise, each row of the rx2 table is compared to the exposure reference level and test of independence two-sided p values are calculated using fisher exact, mid-p exact, or normal approximation method.

Value

`tab`	primary table
`measure`	odds ratio, risk ratio, or rate ratio
`conf.level`	confidence level
`pvalue`	p value method
`x`	data input
`data`	data with margin totals
`p.exposed`	proportion exposed
`p.outcome`	proportion outcome
`p.value`	p value
`correction`	TRUE if Yate's continuity correction was used

Author(s)

Tomas Aragon, <aragon@berkeley.edu>, http://www.phdata.science

References

Nicolas P Jewell, Statistics for Epidemiology, 1st Edition, 2004, Chapman & Hall

Kenneth J. Rothman and Sander Greenland (1998), Modern Epidemiology, Lippincott-Raven Publishers

Kenneth J. Rothman (2002), Epidemiology: An Introduction, Oxford University Press

See Also

`riskratio`, `oddsratio`, `rateratio`

Examples

```
r243 <- matrix(c(12,2,7,9), 2, 2)
dimnames(r243) <- list(Diarrhea = c("Yes", "No"),
                       "Antibody level" = c("Low", "High")
                       )
r243
r243b <- t(r243)
r243b
epitab(r243, rev = "b", verbose = TRUE)
epitab(r243, method="riskratio",rev = "b", verbose = TRUE)
epitab(matrix(c(41, 15, 28010, 19017),2,2)[2:1,],
       method="rateratio", verbose = TRUE)
```

`epitable` *Create r x c contigency table (exposure levels vs. binary outcome)*

Description

Create r x c contigency table for r exposure levels and c outcome levels

Usage

```
epitable(..., ncol =2, byrow = TRUE,
         rev = c("neither", "rows", "columns", "both"))
```

Arguments

`...`	see details
`ncol`	number of columns = 2 (default) when a table is constructed from a vector or sequence of numbers
`byrow`	Default is TRUE and single vector or collection of numbers is read in row-wise. Set to FALSE to read in column-wise.
`rev`	reverse order of "rows", "colums", "both", or "neither" (default)

Details

Creates r x 2 table with r exposure levels and 2 outcome levels (No vs. Yes). Arguments can be one of the following:

(1) four or more integers that be converted into r x 2 table (the number of integers must be even),

(2) two categorical vectors (1st vector is exposure with r levels, 2nd vector is outcome with 2 levels),

(3) r x 2 contingency table, or

(4) single vector that be converted into r x 2 table (the number of integers must be even).

The contingency table created by this function is usually used for additional analyses, for example, the `epitab` function.

Value

Returns r x 2 contingency table, usually for additional analyses.

Author(s)

Tomas Aragon, <aragon@berkeley.edu>, http://www.phdata.science

References

none

See Also

epitable

Examples

```
## single vector
dat <- c(88, 20, 555, 347)
epitable(dat)

## 4 or more integers
epitable(1,2,3,4,5,6)

## single matrix
epitable(matrix(1:6, 3, 2))

## two categorical vectors
exposure <- factor(sample(c("Low", "Med", "High"), 100, rep=TRUE),
                   levels=c("Low", "Med", "High"))
outcome <- factor(sample(c("No", "Yes"), 100, rep=TRUE))
epitable(exposure, outcome)
epitable("Exposure"=exposure, "Disease"=outcome)

## reversing row and/or column order
zz <- epitable("Exposure Level"=exposure, "Disease"=outcome)
zz
epitable(zz, rev = "r")
epitable(zz, rev = "c")
epitable(zz, rev = "b")
```

---

expand.table *Expand contingency table into individual-level data set*

Description

Expands contingency table or array into individual-level data set.

Usage

```
expand.table(x)
```

Arguments

x	table or array with `dimnames(x)` and `names(dimnames(x))`

Details

For educational purposes, one may want to convert a multi-dimensional contingency table into an individual-level data frame. In R, multi-dimensional contigency tables are represented by arrays. An array can be created using the `array` command, or the `table` command with 3 or more vectors (usually fields from a data frame).

It is this array, `x`, that is processed by `expand.table`. In order to generate a data frame, `expand.table` needs to process the field names and the possible values for each field. The array x must have dimension names [i.e., `dimnames(x)`] and field names [i.e., `names(dimnames(x))`]. The `expand.table` function converts `names(dimnames(x))` to field names and the `dimnames(x)` to factor levels for each field. Study the examples.

An `ftable` object, say `ftab`, can be expanded using `expand.table(as.table(ftab))`.

Study the Titanic example to compare how a data frame can contain either individual-level data or group-level data.

Value

Returns an individual-level data frame

Author(s)

Tomas Aragon, <aragon@berkeley.edu>, http://www.phdata.science; Daniel Wollschlaeger, <dwoll@psychologie.uni-kiel.de>, http://www.uni-kiel.de/psychologie/dwoll/

References

none

See Also

expand.grid

Examples

```
##Creating array using 'array' function and expanding it
tab <- array(1:8, c(2, 2, 2))
dimnames(tab) <- list(c("No","Yes"), c("No","Yes"), c("No","Yes"))
names(dimnames(tab)) <- c("Exposure", "Disease", "Confounder")
tab
df <- expand.table(tab)
df
```

```
##Creating array using 'table' function and expanding it
tab2 <- table(Exposure = df$Exp, Disease = df$Dis, Confounder = df$Conf)
expand.table(tab2)

##Expanding ftable object
ftab2 <- ftable(tab2)
ftab2
expand.table(as.table(ftab2))

##Convert Titanic data into individual-level data frame
data(Titanic)
expand.table(Titanic)[1:20,]

##Convert Titanic data into group-level data frame
as.data.frame(Titanic)
```

`expected` *Expected values in a table*

Description

Assuming independence, calculates expected values in a matrix or table.

Usage

```
expected(x)
```

Arguments

x	is a matrix or table

Details

Assuming independence, calculates expected values in a matrix or table.

Value

expected values

Author(s)

Tomas Aragon, <aragon@berkeley.edu>, http://www.phdata.science

References

Steve Selvin (2001), Epidemiologic Analysis: A Case-Oriented Approach, Oxford University Press

See Also

See also margin.table

Examples

```
##From Selvin, 2001, p.2
##year = year of birth
##one+ = one or more congenital defects
##one = one congenital defect
dat <- c(369, 460, 434, 434, 506, 487, 521, 518, 526, 488,
605, 481, 649, 477, 733, 395, 688, 348)

##observed
oi <- matrix(dat, nrow =2)
colnames(oi) <- 1983:1991
rownames(oi) <- c("one+", "one")

##expected
ei <- expected(oi)

##Pearson chi-square test
chi2.T <- sum((oi - ei)^2/ei)
pchisq(q = chi2.T, df = 8, lower.tail = FALSE)
```

---

julian2date *Convert a julian date into standard a date format*

Description

Convert a julian date into a standard calendar date format

Usage

```
julian2date(x)
```

Arguments

x	julian date; that is, the number of days since day 0 (default is 1970-01-01)

Details

In R, the julian function converts a date-time object into a Julian date: the number of day since day 0 (default is 1970-01-01). However, there is no function, without loading another package, that converts a Julian date back into a date object. The julian2date function does this conversion.

Value

Return standard calendar date format.

Author(s)

Tomas Aragon, <aragon@berkeley.edu>, http://www.phdata.science

References

none

See Also

format.Date, weekdays

Examples

```
mydates <- c("1/1/04", "1/2/04", "1/7/04", "1/14/04", "8/18/04");
mydates <- as.Date(mydates, format = "%m/%d/%y")
mydates
myjulian <- julian(mydates)
myjulian
julian2date(myjulian)
```

---

kapmeier *Implements product-limit (Kaplan-Meier) method*

Description

Implements product-limit (Kaplan-Meier) method for time-to-event data with censoring.

Usage

```
kapmeier(time, status)
```

Arguments

time	numeric vector with individual observation times
status	integer vector indicating status at the end of the observation time: 1 = event, 0 = censored

Details

This function implements the product-limit method for estimating survival probability for time-to-event data with censoring:

```
S(t) = product[(nj - dj) / nj] for all tj <= t,
```

where tj are event times (i.e., times at which one or more events occur), nj are the number at risk at time tj (by convention, subjects censored at time tj are considered at-risk and included in nj), and dj are the number of events at time tj.

A primary purpose of this function was to demonstrate the use of available R functions to implement a simple statistical method. For example, kapmeier uses sort, order, duplicated, tapply, unique, cumprod, cbind, and dimnames. Studying this function carefully helps one understand and appreciate the utility of R functions to implement simple methods.

For serious survival analysis load the survival package. The survfit function in this package implements the product-limit method and much more. See examples.

Value

Returns an individual-level data frame

Author(s)

Tomas Aragon, <aragon@berkeley.edu>, http://www.phdata.science

References

Selvin S. Statistical Analysis of Epidemiologic Data (Monographs in Epidemiology and Biostatistics, V. 35). Oxford University Press; 3rd edition (May 1, 2004)

See Also

See also survfit

Examples

```
##Product-limit method using 'kapmeier' function
tt <- c(1,17,20,9,24,16,2,13,10,3)
ss <- c(1,1,1,1,0,0,0,1,0,1)
round(kapmeier(tt, ss), 3)
```

---

oddsratio *Odds ratio estimation and confidence intervals*

Description

Calculates odds ratio by median-unbiased estimation (mid-p), conditional maximum likelihood estimation (Fisher), unconditional maximum likelihood estimation (Wald), and small sample adjustment (small). Confidence intervals are calculated using exact methods (mid-p and Fisher), normal approximation (Wald), and normal approximation with small sample adjustment (small).

Usage

```
oddsratio(x, y = NULL,
          method = c("midp", "fisher", "wald", "small"),
          conf.level = 0.95,
          rev = c("neither", "rows", "columns", "both"),
          correction = FALSE,
          verbose = FALSE)
oddsratio.midp(x, y = NULL,
               conf.level = 0.95,
               rev = c("neither", "rows", "columns", "both"),
               correction = FALSE,
               verbose = FALSE,
               interval =  c(0, 1000))
oddsratio.fisher(x, y = NULL,
                 conf.level = 0.95,
                 rev = c("neither", "rows", "columns", "both"),
                 correction = FALSE,
                 verbose = FALSE)
oddsratio.wald(x, y = NULL,
               conf.level = 0.95,
               rev = c("neither", "rows", "columns", "both"),
               correction = FALSE,
               verbose = FALSE)
oddsratio.small(x, y = NULL,
                conf.level = 0.95,
                rev = c("neither", "rows", "columns", "both"),
                correction = FALSE,
                verbose = FALSE)
```

Arguments

x	input data can be one of the following: r x 2 table, vector of numbers from a contigency table (will be transformed into r x 2 table in row-wise order), or single factor or character vector that will be combined with y into a table.
y	single factor or character vector that will be combined with x into a table (default is NULL)
method	method for calculating odds ratio and confidence interval
conf.level	confidence level (default is 0.95)
rev	reverse order of "rows", "colums", "both", or "neither" (default)
correction	set to TRUE for Yate's continuity correction (default is FALSE)
verbose	set to TRUE to return more detailed results (default is FALSE)
interval	interval for the uniroot that finds the odds ratio median-unbiased estimate and mid-p exact confidence interval for oddsratio.midp

Details

Calculates odds ratio by median-unbiased estimation (mid-p), conditional maximum likelihood estimation (Fisher), unconditional maximum likelihood estimation (Wald), and small sample adjustment (small). Confidence intervals are calculated using exact methods (mid-p and Fisher), normal approximation (Wald), and normal approximation with small sample adjustment (small).

This function expects the following table struture:

```
                disease=0   disease=1
exposed=0 (ref)    n00         n01
exposed=1          n10         n11
exposed=2          n20         n21
exposed=3          n30         n31
```

The reason for this is because each level of exposure is compared to the reference level.

If you are providing a 2x2 table the following table is preferred:

```
                disease=0   disease=1
exposed=0 (ref)    n00         n01
exposed=1          n10         n11
```

however, for odds ratios from 2x2 tables, the following table is equivalent:

```
                disease=1   disease=0
exposed=1          n11         n10
exposed=0          n01         n00
```

If the table you want to provide to this function is not in the preferred form, just use the `rev` option to "reverse" the rows, columns, or both. If you are providing categorical variables (factors or character vectors), the first level of the "exposure" variable is treated as the reference. However, you can set the reference of a factor using the `relevel` function.

Likewise, each row of the rx2 table is compared to the exposure reference level and test of independence two-sided p values are calculated using mid-p exact, Fisher's Exact, Monte Carlo simulation, and the chi-square test.

Value

`x`	table that was used in analysis (verbose = TRUE)
`data`	same table as `x` but with marginal totals
`p.exposed`	proportions exposed (verbose = TRUE)
`p.outcome`	proportions experienced outcome (verbose = TRUE)
`measure`	risk ratio and confidence interval
`conf.level`	confidence level used (verbose = TRUE)
`p.value`	p value for test of independence
`replicates`	number of replicates used in Monte Carlo simulation p value (verbose = TRUE)
`correction`	logical specifying if continuity correction was used

Author(s)

Tomas Aragon, <aragon@berkeley.edu>, http://www.phdata.science

References

Kenneth J. Rothman and Sander Greenland (1998), Modern Epidemiology, Lippincott-Raven Publishers

Kenneth J. Rothman (2002), Epidemiology: An Introduction, Oxford University Press

Nicolas P. Jewell (2004), Statistics for Epidemiology, 1st Edition, 2004, Chapman & Hall, pp. 73-81

See Also

tab2by2.test, riskratio, rateratio, ormidp.test, epitab

Examples

```
##Case-control study assessing whether exposure to tap water
##is associated with cryptosporidiosis among AIDS patients

tapw <- c("Lowest", "Intermediate", "Highest")
outc <- c("Case", "Control")
dat <- matrix(c(2, 29, 35, 64, 12, 6),3,2,byrow=TRUE)
dimnames(dat) <- list("Tap water exposure" = tapw, "Outcome" = outc)
oddsratio(dat, rev="c")
oddsratio.midp(dat, rev="c")
oddsratio.fisher(dat, rev="c")
oddsratio.wald(dat, rev="c")
oddsratio.small(dat, rev="c")
```

---

or.midp *Odds ratio estimation and confidence intervals using mid-p method*

Description

Calculates odds ratio by median-unbiased estimation and exact confidence interval using the mid-p method (Rothman 1998).

Usage

```
or.midp(x, conf.level = 0.95, byrow = TRUE, interval = c(0, 1000))
```

Arguments

x	input data can be 2x2 matrix or vector of length 4
conf.level	confidence level (default is 0.95)
byrow	integer vectors are read in row-wise (default)
interval	interval for the uniroot that finds the odds ratio median-unbiased estimate and mid-p exact confidence interval for oddsratio.midp

Details

Calculates odds ratio by median-unbiased estimation and exact confidence interval using the mid-p method (Rothman 1998, p. 251).

This function expects the following 2x2 table struture:

```
            exposed   not exposed
disease      a1        a0
no disease   b1        b0
```

or a numeric vector of the form c(a1, a0, b1, b0).

This function is used by `oddsratio.midp`.

Value

`x`	table that was used in analysis
`data`	same table as `x` but with marginal totals
`estimate`	median unbiased odds ratio
`conf.level`	confidence level used

Author(s)

Tomas Aragon, <aragon@berkeley.edu>, http://www.phdata.science

References

Kenneth J. Rothman and Sander Greenland (1998), Modern Epidemiology, Lippincott-Raven Publishers

See Also

`oddsratio`

Examples

```
##rothman p. 243
z1 <- matrix(c(12,2,7,9),2,2,byrow=TRUE)
z2 <- z1[2:1,2:1]
##jewell p. 79
z3 <- matrix(c(347,555,20,88),2,2,byrow=TRUE)
z4 <- z3[2:1,2:1]
or.midp(z1)
or.midp(z2)
or.midp(z3)
or.midp(z4)
```

`ormidp.test` *odds ratio test for independence (p value) for a 2x2 table*

Description

Test for independence using the mid-p method (Rothman 1998)

Usage

```
ormidp.test(a1, a0, b1, b0, or = 1)
```

Arguments

`a1`	number of exposed cases
`a0`	number of unexposed cases
`b1`	number of exposed noncases (controls)
`b0`	number of unexposed noncases (controls)
`or`	odds ratio reference value (default is no association)

Details

Test for independence using the mid-p method (Rothman 1998)

Value

`$one.sided`	one-sided p value
`$two.sided`	two-sided p value

Author(s)

Tomas Aragon, <aragon@berkeley.edu>, http://www.phdata.science

References

Kenneth J. Rothman and Sander Greenland (1998), Modern Epidemiology, Lippincott-Raven Publishers

Kenneth J. Rothman (2002), Epidemiology: An Introduction, Oxford University Press

Nicolas P. Jewell (2004), Statistics for Epidemiology, 1st Edition, 2004, Chapman & Hall, pp. 73-81

See Also

`tab2by2.test`, `oddsratio`, `riskratio`

Examples

```
##rothman p. 243
ormidp.test(12,2,7,9)

##jewell p. 79
ormidp.test(347,555,20,88)
```

---

oswego *Outbreak of Gastrointestinal Illness in Oswego County, 1940*

Description

On April 19, 1940, the local health officer in the village of Lycoming, Oswego County, New York, reported the occurrence of an outbreak of acute gastrointestinal illness to the District Health Officer in Syracuse. Dr. A. M. Rubin, epidemiologist-in-training, was assigned to conduct an investigation.

When Dr. Rubin arrived in the field, he learned from the health officer that all persons known to be ill had attended a church supper held on the previous evening, April 18. Family members who did not attend the church supper did not become ill. Accordingly, Dr. Rubin focused the investigation on the supper. He completed Interviews with 75 of the 80 persons known to have attended, collecting information about the occurrence and time of onset of symptoms, and foods consumed. Of the 75 persons interviewed, 46 persons reported gastrointestinal illness.

The onset of illness in all cases was acute, characterized chiefly by nausea, vomiting, diarrhea, and abdominal pain. None of the ill persons reported having an elevated temperature; all recovered within 24 to 30 hours. Approximately 20 physicians. No fecal specimens were obtained for bacteriologic examination.

The supper was held in the basement of the village church. Foods were contributed by numerous members of the congregation. The supper began at 6:00 p.m. and continued until 11:00 p.m. Food was spread out on a table and consumed over a period of several hours. Data regarding onset of illness and food eaten or water drunk by each of the 75 persons interviewed are provided in the attached line listing (Oswego dataset). The approximate time of eating supper was collected for only about half the persons who had gastrointestinal illness.

Usage

```
##data(oswego)
```

Format

- `id` subject identificaton number
- `age` age
- `sex` sex: F = Female, M = Male
- `meal.time` meal time on April 18th
- `ill` developed illness: Y = Yes N = No
- `onset.date` onset date: "4/18" = April 18th, "4/19" = April 19th

- onset.time onset time: HH:MM AM/PM
- baked.ham consumed item: Y = Yes N = No
- spinach consumed item: Y = Yes N = No
- mashed.potato consumed item: Y = Yes N = No
- cabbage.salad consumed item: Y = Yes N = No
- jello rolls consumed item: Y = Yes N = No
- brown.bread consumed item: Y = Yes N = No
- milk consumed item: Y = Yes N = No
- coffee consumed item: Y = Yes N = No
- water consumed item: Y = Yes N = No
- cakes consumed item: Y = Yes N = No
- vanilla.ice.cream consumed item: Y = Yes N = No
- chocolate.ice.cream consumed item: Y = Yes N = No
- fruit.salad consumed item: Y = Yes N = No

Source

Center for Disease Control and Prevention, Epidemic Intelligence Service

References

Oswego: An Outbreak of Gastrointestinal Illness Following a Church Supper (updated 2003): S. aureus outbreak among church picnic attendees, 1940; the classic, straightforward outbreak investigation in a defined population. Training modules available at https://www.cdc.gov/eis/casestudies/xoswego.401-303.student.pdf.

---

pois.conf.int *Confidence intervals for Poisson counts or rates*

Description

Calculates confidence intervals for Poisson counts or rates

Usage

```
pois.exact(x, pt = 1, conf.level = 0.95)
pois.daly(x, pt = 1, conf.level = 0.95)
pois.byar(x, pt = 1, conf.level = 0.95)
pois.approx(x, pt = 1, conf.level = 0.95)
```

Arguments

x	count or vector of counts
pt	person-time at risk (default = 1) or vector of person-times
conf.level	confidence level (default = 0.95)

Details

These functions calculate confidence intervals for a Poisson count or rate using an exact method (pois.exact), gamma distribution (pois.daly), Byar's formula (pois.byar), or normal approximation to the Poisson distribution (pois.approx).

To calculate an exact confidence interval for a crude rate (count divided by person-time at risk), set pt equal to the person-time at risk. Both x and pt can be either a number or a vector of numbers.

The pois.daly function gives essentially identical answers to the pois.exact function except when x = 0. When x = 0, for the upper confidence limit pois.exact returns 3.689 and pois.daly returns 2.996.

Value

This function returns a n x 6 matrix with the following colnames:

x	Poisson count
pt	person-time at risk
rate	crude rate = x/pt
lower	lower confidence interval limit
upper	upper confidence interval limit
conf.level	confidence level

Author(s)

Tomas Aragon, <aragon@berkeley.edu>, https://repitools.wordpress.com/; with contributions by Francis Dimzon, <fdimzon@yahoo.com>; with contributions by Scott Nabity, <scott.nabity@sfdph.org>

References

Tomas Aragon, et al. Applied Epidemiology Using R. Available at http://www.phdata.science

Leslie Day (1992), "Simple SAS macros for the calculation of exact binomial and Poisson confidence limits." Comput Biol Med, 22(5):351-361

Kenneth Rothman (2002), Epidemiology: An Introduction, Oxford University Press, 1st Edition.

See Also

binom.exact

Examples

```
pois.exact(1:10)
pois.exact(1:10, 101:110)
pois.daly(1:10)
pois.daly(1:10, 101:110)
pois.byar(1:10)
pois.byar(1:10, 101:110)
pois.approx(1:10)
pois.approx(1:10, 101:110)
```

probratio *Obtain unbiased probability ratios from logistic regression models*

Description

Estimates probability (prevalence or risk) ratios from logistic regression models using either maximum likelihood or marginal standardization. When using the latter, standard errors are calculated using the delta method or bootstrap.

Usage

```
probratio(object, parm, subset, method=c('ML', 'delta', 'bootstrap'),
  scale=c('linear', 'log'), level=0.95, seed, NREPS=100, ...)
```

Arguments

object	a glm object with the family attribute equal to "binomial"
parm	a specification of which parameters are to be sequentially assigned predicted responses, either a vector of numbers or a vector of names. If missing, all parameters are considered except the intercept which should not be used except when the method argument is "model".
subset	a logical vector referring to which observations are included in the numerators and denominators of risk calculation. The default is TRUE, corresponding to a total population prediction ratios. User can supply subsets to calculate exposed population prediction ratios.
method	One of three ways that standard errors of prediction ratios are calculate. Maximum likelihood uses relative risk regression directly. Delta-method uses asymptotically correct normal approximations to prediction ratios.
scale	The scale on which marginal standardization calculates normal approximations to variability. When using ML, the log scale is the efficient parameterization.
level	The confidence level for confidence intervals.
seed	The random number generation seed
NREPS	The number of bootstrap samples to be drawn
...	Further arguments to glm when using maximum likelihood

Details

Estimates prevalence and risk ratios from logistic regression models using either maximum likelihood or marginal standardization. Maximum likelihood is relative risk regression: a GLM with binomial variance structure and a log link. Marginal standardization averages predicted probabilities from logistic regression models in the total sample or exposed sample to obtain prevalence or risk ratios. Standard errors for marginal standardization estimates are calculated with the delta method or the normal bootstrap, which is not bias corrected. Ratios can be estimated on the linear or log scale, which may lead to different inference due to the invariance of Wald statistics.

Value

An array of ratios or log ratios, their standard errors, a z-score for a hypothesis test for the log ratio being different from 0 or the ratio being different from 1, the corresponding p-value, and the confidence interval for the estimate.

Note

Maximum likelihood estimation via Newton Raphson may result in predicted probabilities greater than 1. This dominates estimating functions and leads to either false convergence or failure. Users should attempt to refit such models themselves using glms with the family argument binomial(link=log). By modifying inputs to glm.control, domination may be averted. An ideal first step is supplying starting coefficients. Input start=c(-log(p), 0,0,...,0) where p is the prevalence of the outcome. The current implementation of bootstrap standard errors, inference, and confidence intervals are not bias corrected. This will be updated in a later version.

Author(s)

Adam Omidpanah, <adam.omidpanah@wsu.edu>

References

Muller, Clemma J., and Richard F. MacLehose. "Estimating predicted probabilities from logistic regression: different methods correspond to different target populations." International journal of epidemiology 43.3 (2014): 962-970.

Lumley, Thomas, Richard Kronmal, and Shuangge Ma. "Relative risk regression in medical research: models, contrasts, estimators, and algorithms." (2006).

See Also

glm, deriv,w predict.glm, family

Examples

```
set.seed(123)
x <- rnorm(500)
y <- rbinom(500, 1, exp(-1 + .3*x))
logreg <- glm(y ~ x, family=binomial)
confint.default(logreg) ## 95% CI over-estimates the 0.3 log-RR
pr1 <- probratio(logreg, method='ML', scale='log', start=c(log(mean(y)), 0))

## generally more efficient to calculate log-RR then exponentiate for non-symmetric 95% CI
pr1 <- probratio(logreg, scale='log', method='delta')
pr2 <- probratio(logreg, scale='linear', method='delta')
exp(pr1[, 5:6])
pr2[, 5:6]
```

`rate2by2.test` *Comparative tests of independence in rx2 rate tables*

Description

Tests for independence where each row of the rx2 table is compared to the exposure reference level and test of independence two-sided p values are calculated using mid-p xxact, and normal approximation.

Usage

```
rate2by2.test(x, y = NULL, rr = 1,
              rev = c("neither", "rows", "columns", "both"))
```

Arguments

`x`	input data can be one of the following: r x 2 table where first column contains disease counts and second column contains person time at risk; or a single numeric vector for counts followed by person time at risk
`y`	vector of person-time at risk; if provided, x must be a vector of disease counts
`rr`	rate ratio reference value (default is no association)
`rev`	reverse order of "rows", "colums", "both", or "neither" (default)

Details

Tests for independence where each row of the rx2 table is compared to the exposure reference level and test of independence two-sided p values are calculated using mid-p xxact, and normal approximation.

This function expects the following table struture:

```
                  counts   person-time
exposed=0 (ref)     n00        t01
exposed=1           n10        t11
exposed=2           n20        t21
exposed=3           n30        t31
```

The reason for this is because each level of exposure is compared to the reference level.

If the table you want to provide to this function is not in the preferred form, just use the `rev` option to "reverse" the rows, columns, or both. If you are providing categorical variables (factors or character vectors), the first level of the "exposure" variable is treated as the reference. However, you can set the reference of a factor using the `relevel` function.

Likewise, each row of the rx2 table is compared to the exposure reference level and test of independence two-sided p values are calculated using mid-p exact method and normal approximation.

This function can be used to construct a p value function by testing the MUE to the null hypothesis (rr=1) and alternative hypotheses (rr not equal to 1) to calculate two-side mid-p exact p values. For more detail, see Rothman.

Value

x	table that was used in analysis
p.value	p value for test of independence

Author(s)

Tomas Aragon, <aragon@berkeley.edu>, http://www.phdata.science

References

Kenneth J. Rothman and Sander Greenland (2008), Modern Epidemiology, Lippincott Williams and Wilkins Publishers

Kenneth J. Rothman (2002), Epidemiology: An Introduction, Oxford University Press

See Also

rateratio,

Examples

```
##Examples from Rothman 1998, p. 238
bc <- c(Unexposed = 15, Exposed = 41)
pyears <- c(Unexposed = 19017, Exposed = 28010)
dd <- matrix(c(41,15,28010,19017),2,2)
dimnames(dd) <- list(Exposure=c("Yes","No"), Outcome=c("BC","PYears"))
##midp
rate2by2.test(bc,pyears)
rate2by2.test(dd, rev = "r")
rate2by2.test(matrix(c(15, 41, 19017, 28010),2,2))
rate2by2.test(c(15, 41, 19017, 28010))
```

---

rateratio *Rate ratio estimation and confidence intervals*

Description

Calculates rate ratio by median-unbiased estimation (mid-p), and unconditional maximum likelihood estimation (Wald). Confidence intervals are calculated using exact methods (mid-p), and normal approximation (Wald).

Usage

```
rateratio(x, y = NULL,
          method = c("midp", "wald"),
          conf.level = 0.95,
          rev = c("neither", "rows", "columns", "both"),
          verbose = FALSE)
```

```
rateratio.midp(x, y = NULL,
               conf.level = 0.95,
               rev = c("neither", "rows", "columns", "both"),
               verbose = FALSE)
rateratio.wald(x, y = NULL,
               conf.level = 0.95,
               rev = c("neither", "rows", "columns", "both"),
               verbose = FALSE)
```

Arguments

x	input data can be one of the following: r x 2 table where first column contains disease counts and second column contains person time at risk; a single numeric vector of counts followed by person time at risk; a single numeric vector of counts combined with y which would be a numeric vector of corresponding person time at risk
y	numeric vector of person-time at risk; if provided, x must be a numeric vector of disease counts
method	method for calculating rate ratio and confidence interval
conf.level	confidence level (default is 0.95)
rev	reverse order of "rows", "colums", "both", or "neither" (default)
verbose	set to TRUE to return more detailed results (default is FALSE)

Details

Calculates rate ratio by median-unbiased estimation (mid-p), and unconditional maximum likelihood estimation (Wald). Confidence intervals are calculated using exact methods (mid-p), and normal approximation (Wald).

This function expects the following table struture:

```
                  counts   person-time
exposed=0 (ref)     n00         t01
exposed=1           n10         t11
exposed=2           n20         t21
exposed=3           n30         t31
```

The reason for this is because each level of exposure is compared to the reference level.

If the table you want to provide to this function is not in the preferred form, just use the rev option to "reverse" the rows, columns, or both. If you are providing categorical variables (factors or character vectors), the first level of the "exposure" variable is treated as the reference. However, you can set the reference of a factor using the relevel function.

Likewise, each row of the rx2 table is compared to the exposure reference level and test of independence two-sided p values are calculated using mid-p exact method and normal approximation (Wald).

Value

x	table that was used in analysis (verbose = TRUE)
data	same table as x but with marginal totals
measure	rate ratio and confidence interval
conf.level	confidence level used (verbose = TRUE)
p.value	p value for test of independence

Author(s)

Rita Shiau (original author), <rita.shiau@sfdph.org>; Tomas Aragon, <aragon@berkeley.edu>, http://www.phdata.science; Adam Omidpanah, <adam.omidpanah@wsu.edu> https://repitools.wordpress.com/

References

Kenneth J. Rothman, Sander Greenland, and Timothy Lash (2008), Modern Epidemiology, Lippincott-Raven Publishers

Kenneth J. Rothman (2012), Epidemiology: An Introduction, Oxford University Press

See Also

rate2by2.test, oddsratio, riskratio, epitab

Examples

```
##Examples from Rothman 1998, p. 238
bc <- c(Unexposed = 15, Exposed = 41)
pyears <- c(Unexposed = 19017, Exposed = 28010)
dd <- matrix(c(41,15,28010,19017),2,2)
dimnames(dd) <- list(Exposure=c("Yes","No"), Outcome=c("BC","PYears"))
##midp
rateratio(bc,pyears)
rateratio(dd, rev = "r")
rateratio(matrix(c(15, 41, 19017, 28010),2,2))
rateratio(c(15, 41, 19017, 28010))

##midp
rateratio.midp(bc,pyears)
rateratio.midp(dd, rev = "r")
rateratio.midp(matrix(c(15, 41, 19017, 28010),2,2))
rateratio.midp(c(15, 41, 19017, 28010))

##wald
rateratio.wald(bc,pyears)
rateratio.wald(dd, rev = "r")
rateratio.wald(matrix(c(15, 41, 19017, 28010),2,2))
rateratio.wald(c(15, 41, 19017, 28010))
```

`ratetable` *Create r x 2 count and person-time table for calculating rates*

Description

Create r x 2 count and person-time table for calculating rates

Usage

```
ratetable(..., byrow = FALSE,
          rev = c("neither", "rows", "columns", "both"))
```

Arguments

`...`	see details
`byrow`	Default is TRUE and single vector or collection of numbers is read in row-wise. Set to FALSE to read in column-wise.
`rev`	reverse order of "rows", "colums", "both", or "neither" (default)

Details

Creates r x 2 table with r exposure levels and 2 columns (counts and person-time exposed). Arguments can be one of the following:

(1) r x 2 table of the following form:

```
                      Outcome
Exposure    cases pyears
E = 0 (ref)   a     PT0
E = 1         b     PT1
```

(2) Two numeric vectors: 1st should be vector of counts, and the 2nd vector should be vector of person-times at risk. For example,

```
cases  <- c(a, b)
pyears <- c(PT0, PT1)
```

(3) >= 4 numbers in the following order: a, PT0, b, PT1

(4) One numeric vector of the following form: c(a, PT0, b, PT1)

Value

Returns r x 2 rate table, usually for additional analyses.

Author(s)

Tomas Aragon, <aragon@berkeley.edu>, http://www.phdata.science

References

none

See Also

epitable

Examples

```
##Breast cancer cases from radiation treatment for tuberculosis
##Rothman 1998, p. 238
bc0 <- 15
bc1 <- 41
py0 <- 19017
py1 <- 28010

##4 numbers
ratetable(bc0, py0, bc1, py1)

##1 vector
dat <- c(bc0, py0, bc1, py1)
ratetable(dat)

##2 vectors
cases <- c(bc0, bc1)
pyears <- c(py0, py1)
ratetable(bc.cases = cases, person.years = pyears)

##1 matrix
r238 <- matrix(c(41, 28010, 15, 19017), 2, 2)
dimnames(r238) <- list(c("BC cases", "Person-years"),
                       "Radiation" = c("Yes", "No"))
r238
r238b <- t(r238)
r238b
ratetable(r238b, rev = "r")
```

---

riskratio *Risk ratio estimation and confidence intervals*

Description

Calculates risk ratio by unconditional maximum likelihood estimation (Wald), and small sample adjustment (small). Confidence intervals are calculated using normal approximation (Wald), and normal approximation with small sample adjustment (small), and bootstrap method (boot).

Usage

```
riskratio(x, y = NULL,
          method = c("wald", "small", "boot"),
          conf.level = 0.95,
          rev = c("neither", "rows", "columns", "both"),
          correction = FALSE,
          verbose = FALSE,
          replicates = 5000)
riskratio.wald(x, y = NULL,
               conf.level = 0.95,
               rev = c("neither", "rows", "columns", "both"),
               correction = FALSE,
               verbose = FALSE)
riskratio.small(x, y = NULL,
                conf.level = 0.95,
                rev = c("neither", "rows", "columns", "both"),
                correction = FALSE,
                verbose = FALSE)
riskratio.boot(x, y = NULL,
               conf.level = 0.95,
               rev = c("neither", "rows", "columns", "both"),
               correction = FALSE,
               verbose = FALSE,
               replicates = 5000)
```

Arguments

x	input data can be one of the following: r x 2 table, vector of numbers from a contigency table (will be transformed into r x 2 table in row-wise order), or single factor or character vector that will be combined with y into a table.
y	single factor or character vector that will be combined with x into a table (default is NULL)
method	method for calculating risk ratio and confidence interval
conf.level	confidence level (default is 0.95)
rev	reverse order of "rows", "colums", "both", or "neither" (default)
correction	set to TRUE for Yate's continuity correction (default is FALSE)
verbose	set to TRUE to return more detailed results (default is FALSE)
replicates	Number of bootstrap replicates (default = 5000)

Details

Calculates risk ratio by unconditional maximum likelihood estimation (Wald), and small sample adjustment (small). Confidence intervals are calculated using normal approximation (Wald), and normal approximation with small sample adjustment (small), and bootstrap method (boot).

This function expects the following table struture:

```
                 disease=0   disease=1
exposed=0 (ref)    n00         n01
exposed=1          n10         n11
exposed=2          n20         n21
exposed=3          n30         n31
```

The reason for this is because each level of exposure is compared to the reference level.

If you are providing a 2x2 table the following table is preferred:

```
                 disease=0   disease=1
exposed=0 (ref)    n00         n01
exposed=1          n10         n11
```

If the table you want to provide to this function is not in the preferred form, just use the `rev` option to "reverse" the rows, columns, or both. If you are providing categorical variables (factors or character vectors), the first level of the "exposure" variable is treated as the reference. However, you can set the reference of a factor using the `relevel` function.

Likewise, each row of the rx2 table is compared to the exposure reference level and test of independence two-sided p values are calculated using Fisher's Exact, Monte Carlo simulation, and the chi-square test.

Value

`x`	table that was used in analysis (verbose = TRUE)
`data`	same table as `x` but with marginal totals
`p.exposed`	proportions exposed (verbose = TRUE)
`p.outcome`	proportions experienced outcome (verbose = TRUE)
`measure`	risk ratio and confidence interval
`conf.level`	confidence level used (verbose = TRUE)
`boot.replicates`	number of replicates used in bootstrap estimation of confidence intervals (verbose = TRUE)
`p.value`	p value for test of independence
`mc.replicates`	number of replicates used in Monte Carlo simulation p value (verbose = TRUE)
`correction`	logical specifying if continuity correction was used

Author(s)

Tomas Aragon, <aragon@berkeley.edu>, http://www.phdata.science

References

Kenneth J. Rothman and Sander Greenland (1998), Modern Epidemiology, Lippincott-Raven Publishers

Kenneth J. Rothman (2002), Epidemiology: An Introduction, Oxford University Press

Nicolas P. Jewell (2004), Statistics for Epidemiology, 1st Edition, 2004, Chapman & Hall, pp. 73-81

Steve Selvin (1998), Modern Applied Biostatistical Methods Using S-Plus, 1st Edition, Oxford University Press

See Also

tab2by2.test, oddsratio, rateratio, epitab

Examples

```
##Case-control study assessing whether exposure to tap water
##is associated with cryptosporidiosis among AIDS patients

tapw <- c("Lowest", "Intermediate", "Highest")
outc <- c("Case", "Control")
dat <- matrix(c(2, 29, 35, 64, 12, 6),3,2,byrow=TRUE)
dimnames(dat) <- list("Tap water exposure" = tapw, "Outcome" = outc)
riskratio(dat, rev="c")
riskratio.wald(dat, rev="c")
riskratio.small(dat, rev="c")

##Selvin 1998, p. 289
sel <- matrix(c(178, 79, 1411, 1486), 2, 2)
dimnames(sel) <- list("Behavior type" = c("Type A", "Type B"),
                      "Outcome" = c("CHD", "No CHD")
                     )
riskratio.boot(sel, rev = "b")
riskratio.boot(sel, rev = "b", verbose = TRUE)
riskratio(sel, rev = "b", method = "boot")
```

---

tab2by2.test *Comparative tests of independence in rx2 contigency tables*

Description

Tests for independence where each row of the rx2 table is compared to the exposure reference level and test of independence two-sided p values are calculated using mid-p exact, Fisher's Exact, and the chi-square test.

Usage

```
tab2by2.test(x, y = NULL,
             correction = FALSE,
             rev = c("neither", "rows", "columns", "both"))
```

Arguments

x	input data can be one of the following: r x 2 table, vector of numbers from a contigency table (will be transformed into r x 2 table in row-wise order), or single factor or character vector that will be combined with y into a table.
y	single factor or character vector that will be combined with x into a table (default is NULL)
correction	set to TRUE for Yate's continuity correction (default is FALSE)
rev	reverse order of "rows", "colums", "both", or "neither" (default)

Details

Tests for independence where each row of the rx2 table is compared to the exposure reference level and test of independence two-sided p values are calculated using mid-p exact, Fisher's Exact, and the chi-square test.

This function expects the following table struture:

```
                 disease=0   disease=1
exposed=0 (ref)    n00         n01
exposed=1          n10         n11
exposed=2          n20         n21
exposed=3          n30         n31
```

The reason for this is because each level of exposure is compared to the reference level.

If you are providing a 2x2 table order does not matter:

If the table you want to provide to this function is not in the preferred form, just use the `rev` option to "reverse" the rows, columns, or both. If you are providing categorical variables (factors or character vectors), the first level of the "exposure" variable is treated as the reference. However, you can set the reference of a factor using the `relevel` function.

Likewise, each row of the rx2 table is compared to the exposure reference level and test of independence two-sided p values are calculated using mid-p exact, Fisher's Exact, Monte Carlo simulation, and the chi-square test.

Value

x	table that was used in analysis
p.value	p value for test of independence
correction	logical specifying if continuity correction was used

Author(s)

Tomas Aragon, <aragon@berkeley.edu>, http://www.phdata.science

References

Kenneth J. Rothman and Sander Greenland (1998), Modern Epidemiology, Lippincott-Raven Publishers

Kenneth J. Rothman (2002), Epidemiology: An Introduction, Oxford University Press

Nicolas P. Jewell (2004), Statistics for Epidemiology, 1st Edition, 2004, Chapman & Hall, pp. 73-81

See Also

oddsratio, riskratio

Examples

```
##Case-control study assessing whether exposure to tap water
##is associated with cryptosporidiosis among AIDS patients

tapw <- c("Lowest", "Intermediate", "Highest")
outc <- c("Case", "Control")
dat <- matrix(c(2, 29, 35, 64, 12, 6),3,2,byrow=TRUE)
dimnames(dat) <- list("Tap water exposure" = tapw, "Outcome" = outc)
tab2by2.test(dat, rev="c")
```

---

table.margins *Marginal totals of a table*

Description

Calculates marginal totals of a matrix, table, or array.

Usage

```
table.margins(x)
```

Arguments

x	is a matrix, table, or array

Details

Calculates marginal totals of a matrix, table, or array.

Value

Returns original object with marginal totals

Author(s)

Tomas Aragon, <aragon@berkeley.edu>, http://www.phdata.science

References

none

See Also

See also `margin.table`

Examples

```
x <- matrix(1:4, 2, 2)
table.margins(x)
```

`wcgs` *Western Collaborative Group Study data*

Description

The Western Collaborative Group Study (WCGS), a prospective cohort studye, recruited middle-aged men (ages 39 to 59) who were employees of 10 California companies and collected data on 3154 individuals during the years 1960-1961. These subjects were primarily selected to study the relationship between behavior pattern and the risk of coronary hearth disease (CHD). A number of other risk factors were also measured to provide the best possible assessment of the CHD risk associated with behavior type. Additional variables collected include age, height, weight, systolic blood pressure, diastolic blood pressure, cholesterol, smoking, and corneal arcus.

Usage

```
##data(wcgs)
```

Format

- `id` Subject ID:
- `age0` Age: age in years
- `height0` Height: height in inches
- `weight0` Weight: weight in pounds
- `sbp0` Systolic blood pressure: mm Hg
- `dbp0` Diastolic blood pressure: mm Hg
- `chol0` Cholesterol: mg/100 ml
- `behpat0` Behavior pattern:
- `ncigs0` Smoking: Cigarettes/day
- `dibpat0` Dichotomous behavior pattern: 0 = Type B; 1 = Type A
- `chd69` Coronary heart disease event: 0 = none; 1 = yes
- `typechd` to be done
- `time169` Observation (follow up) time: Days
- `arcus0` Corneal arcus: 0 = none; 1 = yes

Source

UC Berkeley School of Public Health

References

pending

---

wnv — *West Nile Virus human cases reported in California, USA, as of December 14, 2004*

Description

Public Health Surveillance data

Usage

```
##data(wnv)
```

Format

pending

Source

California Department of Health Services

References

pending

Index