Package ‘adepro’

April 25, 2024

Type Package

Title A 'shiny' Application for the (Audio-)Visualization of Adverse Event Profiles

Version 4.1.0

Author Nicole Rethemeier, Christoph Tasto, Steffen Jeske

Maintainer Bodo Kirsch <bodo.kirsch@bayer.com>

Description Contains a 'shiny' application called AdEPro (Animation of Adverse Event Profiles) which (audio-)visualizes adverse events occurring in clinical trials. As this data is usually considered sensitive, this tool is provided as a stand-alone application that can be launched from any local machine on which the data is stored.

Depends R (>= 3.5.0), shinyBS, seriation (>= 1.2.9)

License GPL-3

URL https://github.com/Bayer-Group/BIC-AdEPro

Encoding UTF-8

RoxygenNote 7.2.3

Imports graphics, MASS, V8, forcats, here, utils, shinyjs, shiny, audio, shape, Cairo, dplyr, readr, rlang, tidyr, haven, stats, shinyWidgets

Suggests knitr, rmarkdown, testthat

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2024-04-25 14:50:02 UTC

R topics documented:

`adae_data` *Adverse event data set included in AdEPro*

Description

Adverse event data set included in AdEPro

`adsl` *Subject level data set included in AdEPro*

Description

Subject level data set included in AdEPro

`launch_adepro` *launch_adepro - Launches the AdEPro application*

Description

Starts the AdEPro application in the client's browser.

Usage

```
launch_adepro(host = "127.0.0.1", port = NULL, browser = NULL)
```

Arguments

`host`	host link (defaults to the local machine "127.0.0.1")
`port`	port number (randomly chosen unless specified as a certain number)
`browser`	path to browser exe (defaults to standard browser)

Details

Further information on how to use this application can be found in the vignette of this package.

Value

A shiny app

Examples

```
## Not run:
## Launch application on localhost (127.0.0.1)
## -------------------------------------------
## By default launch_adepro starts the application on localhost
## and a randomly selected port (e.g. 9876), in which case you can connect
## to the running application by navigating your browser to
## http://localhost:9876.
launch_adepro()

## Launch application on a different host
## --------------------------------------
## You can also run the application on a different host
## by specifying a hostname and port. Just make sure to
## use an open port on your machine. Here "open" means
## that the port should not be used by another service
## and the port is opened by your firewall.
launch_adepro(host="your-hostname", port=8888)


## Make the application available to your coworkers
## ------------------------------------------------
## within your local area network even without a
## dedicated Shiny server. The value set through the
## host argument says to accept any connection (not just from localhost).
## Then take note of your local IP (if you are under linux,
## you can see it through ifconfig). Say your IP is 192.168.1.70.
## Your colleagues can use your app by inserting in the address
## bar of their browser 192.168.1.70:8888, i.e. your IP followed
## by : and the port number you selected.
launch_adepro(host="0.0.0.0", port=8888)

## Launch application on a different browser
## ----------------------------------------
## To run the shiny app on a different browser than your standard browser
## use the "browser" argument to set the path to the respective .exe file.
launch_adepro(browser = "C:/Program Files/Mozilla Firefox/firefox.exe")


## launching the application.

## End(Not run)
```

`my.symbols` *Draw Symbols (User Defined) on a Plot*

Description

This function draws symbols on a plot. It is similar to the builtin symbols function with the difference that it plots symbols defined by the user rather than a prespecified set of symbols.

Usage

```
my.symbols(x, y=NULL, symb, inches=1, xsize, ysize,
add=TRUE,
vadj=0.5, hadj=0.5,
symb.plots=FALSE,
xlab=deparse(substitute(x)),
ylab=deparse(substitute(y)), main=NULL,
xlim=NULL, ylim=NULL, linesfun=lines,
..., MoreArgs)
```

Arguments

`x, y`	The x and y coordinates for the position of the symbols to be plotted. These can be specified in any way which is accepted by xy.coords.
`symb`	Either a matrix, list, or function defining the symbol to be plotted. If it is a matrix or list it needs to be formatted that it can be passed directly to the lines function. It then defines the shape of the symbol on on a range/domain of -1 to 1. If this is a function it can either return a matrix or list as above (points on the range/domain of -1 to 1), or it can do the plotting itself.
`inches`	The size of the square containing the symbol in inches (note: unlike symbols this cannot be FALSE). This is ignored if xsize or ysize is specified.
`xsize`	The width of the bounding box(s) of the symbols in the same units as the x variable. Computed from ysize or inches if not specified. Can be a single value or a vector.
`ysize`	The height of the bounding box(s) of the symbols in the same units as the y variable. Computed from xsize or inches if not specified. Can be a single value or a vector.
`add`	if 'add' is 'TRUE' then the symbols are added to the existing plot, otherwise a new plot is created.
`vadj, hadj`	Numbers between 0 and 1 indicating how 'x' and 'y' specify the location of the symbol. The defaults center the symbol at x,y; 0 means put the bottom/left at x,y; and 1 means put the top/right of the symbol at x,y.
`symb.plots`	If symb is a function that does its own plotting, set this to TRUE, otherwise it should be FALSE.
`xlab, ylab, main, xlim, ylim`	If 'add' is 'FALSE' these are passed to the plot function when setting up the plot.
`linesfun`	The function to draw the lines if the function does not do its own drawing. The default is lines but could be replaced with polygon to draw filled polygons
`...`	Additional arguments will be replicated to the same length as x then passed to symb (if symb is a function) and/or the lines function (one value per symbol drawn).
`MoreArgs`	A list with any additional arguments to be passed to the symb function (as is, without being replicated/split).

Details

The symb argument can be a 2 column matrix or a list with components 'x' and 'y' that defines points on the interval [-1,1] that will be connected with lines to draw the symbol. If you want a closed polygon then be sure to replicate the 1st point as the last point. If any point contains an NA then the line will not be drawn to or from that point. This can be used to create a symbol with disjoint parts that should not be connected. If symb is a function then it should include a '...' argument along with any arguments to define the symbol. Any unmatched arguments that end up in the '...' argument will be replicated to the same length as 'x' (using the rep function) then the values will be passed one at a time to the symb function. If MoreArgs is specified, the elements of it will also be passed to symb without modification. The symb function can either return a matrix or list with the points that will then be passed to the lines function (see above). Or the function can call the plotting functions itself (set symb.plots to TRUE). High level plotting can be done (plot, hist, and other functions), or low level plotting functions (lines, points, etc) can be used; in this case they should add things to a plot with 'x' and 'y' limits of -1 to 1. The size of the symbols can be specified by using inches in which case the symbol will be set inside of squares whose sizes are inches size based on the plotting device. The size can also be set using xsize and/or ysize which use the same units as the x and/or y variables. If only one is specified then the box will be square. If both are specified and they do not match the aspect ratio of the plot then the bounding box will not be square and the symbol will be distorted.

Value

This function is run for its side effect of plotting, it returns an invisible NULL.

Note

Since the '...' argument is passed to both lines and symb, the symb function should have a '...' argument so that it will ignore any additional arguments. Arguments such as 'type' can be passed through the '...' argument if you want the symbol made of something other than lines. Plotting coordinates and sizes are based on the size of the device at the time the function is called. If you resize the device after plotting, all bets are off. Currently missing values in x or y are not handled well. It is best if remove all missing values first.

Author(s)

Greg Snow

Index