

Package ‘rpanel’

April 19, 2009

Type Package

Title Simple interactive controls for R using the tcltk library.

Version 1.0-5

Date 2008-12-15

Author Bowman, Bowman, Gibson and Crawford

Maintainer Ewan Crawford <ewan@stats.gla.ac.uk>

Depends R (>= 2.5), tcltk

Suggests tkrplot, rgl, sp, geoR, RandomFields, sm

Description rpanel provides a set of functions to build simple GUI controls for R functions. These are built on the tcltk package. Uses could include changing a parameter on a graph by animating it with a slider or a “doublebutton”, up to more sophisticated control panels.

License GPL-2

Repository CRAN

Date/Publication 2008-12-13 10:30:13

R topics documented:

rpanel-package	2
aircond	4
CofE	5
gullweight	5
luthor	6
river	7
rodent	7
rp.ancova	8
rp.block	9
rp.button	10
rp.cartoons	12

rp.checkbox	13
rp.ci	14
rp.clearlines	15
rp.control	16
rp.deleteline	18
rp.do	19
rp.doublebutton	20
rp.firth	22
rp.geosim	23
rp.grid	25
rp.gulls	26
rp.image	27
rp.likelihood	28
rp.line	29
rp.listbox	31
rp.logistic	32
rp.menu	34
rp.messagebox	35
rp.mururoa	36
rp.normal	38
rp.panel	39
rp.panelname	40
rp.plot3d	41
rp.pos	42
rp.power	44
rp.radiogroup	45
rp.regression	47
rp.rmpplot	49
rp.slider	50
rp.tables	52
rp.textentry	53
rp.tkrplot	54

Index	57
--------------	-----------

rpanel-package	<i>simple interactive controls for R functions using the tcltk package</i>
----------------	--

Description

rpanel provides a set of wrapper functions to build simple GUI controls for R functions. Uses could include changing a parameter on a graph (and animating it) with a slider, or a "doublebutton", right up to more sophisticated mini-applications.

Details

Package: rpanel
Type: Package
Version: 1.0-5
Date: 2008-12-15
License: GNU

This package contains a number of functions (with help and examples) and several example scripts.

Package functions

`rp.panelname`: automatic generation of a panel name
`rp.control`: create an rpanel
`rp.slider`: add a slider to a panel, to graphically control a numeric variable
`rp.textentry`: adds a box allows text to be entered
`rp.button`: adds a button to the panel with a nominated function called on pressing
`rp.checkbox`: adds a checkbox to the panel, to control a logical variable
`rp.radiogroup`: adds a set of radiobuttons to the panel
`rp.listbox`: adds a listbox to the panel
`rp.doublebutton`: adds a widget with '+' and '-' buttons, to increment and decrement a variable
`rp.image`: adds an image to the panel, the action function is called with coordinates on clicking
`rp.line`: draws a line connecting the pixel locations x1, y1 to x2, y2 on the specified `rp.image`
`rp.deleteline`: removes a line from an `rp.image`
`rp.clearlines`: removes all lines from an `rp.image`
`rp.messagebox`: displays a message in a pop-up window
`rp.tkrplot`: calls Luke Tierney's `tkrplot` function to allow R graphics to be displayed in a panel
`rp.tkrreplot`: calls Luke Tierney's `tkrreplot` functions to allow R graphics to be displayed in a panel.
`rp.block`: blocks use of the R console until a panel is closed
`rp.panel`: returns a named panel or the most recently created panel
`rp.pos`: a demonstration function for layout control
`rp.grid`: a grid system for layout control
`rp.do`: executes a nominated user defined callback function

Applications functions

`rp.gulls`: STEPS module - the Birds and the Bees
`rp.ci`: Confidence intervals
`rp.ancova`: Analysis of covariance
`rp.power`: Power calculations for a two-sample t-test
`rp.normal`: Fitting a normal distribution to a single sample
`rp.rmplot`: Plotting of repeated measurement data
`rp.tables`: Interactive statistical tables
`rp.regression`: Regression with one or two covariates
`rp.plot3d`: Interactive display of a plot of three variables

[rp.likelihood](#): Exploration of one and two parameter likelihood functions
[rp.logistic](#): Interactive display of logistic regression with a single covariate
[rp.cartoons](#): A menu-driven set of rpanel illustrations
[rp.geosim](#): Simulation of spatial processes
[rp.mururoa](#): Sampling in Mururoa Atoll
[rp.firth](#): Sampling in a firth

Author(s)

A.W.Bowman & E.Crawford

Maintainer: Ewan Crawford <ewan@stats.gla.ac.uk>

References

rpanel: Simple interactive controls for R functions using the tcltk package. Journal of Statistical Software, 17, issue 9.

See Also

[rp.control](#),[rp.button](#),[rp.slider](#),[rp.doublebutton](#),[rp.textentry](#),[rp.checkbox](#),[rp.radiogroup](#)

Examples

```
if (interactive()) {
  rp.gulls()
}
```

aircond

Intervals between the failure of air-conditioning equipment in aircraft

Description

These data, reported by Proschan (1963, Technometrics 5, 375-383), refer to the intervals, in service-hours, between failures of the air-conditioning equipment in a Boeing 720 aircraft. (Proschan reports data on 10 different aircraft. The data from only one of the aircraft is used here. Cox and Snell (1981, Applied Statistics: principles and examples, Chapman and Hall, London) discuss the analysis of the data on all 10 aircraft.)

The dataset consists of a single vector of data.

The data are used in the rp.likelihood example script.

References

rpanel: Simple interactive controls for R functions using the tcltk package. Journal of Statistical Software, 17, issue 9.

Examples

```

if (interactive()) {
  data(aircond)
  hist(aircond)
  rp.likelihood("dexp(data, theta)", aircond, 0.005, 0.03)
  rp.likelihood("dgamma(data, theta[1], theta[2])",
               aircond, c(0.3, 0.005), c(3, 0.06))
}

```

CofE

*Data for the rp.regression2 example script***Description**

These data record the average annual giving in pounds per church member in the dioceses of the Church of England in the early 1980's. Three potentially relevant covariates are also recorded for each diocese, namely the percentage of the population who are employed, the percentage of the population on the electoral roll of the church and the percentage of the population who usually attend church. Background details are available in Pickering (1985; Applied Economics 17, 619-32).

The data are used in the `rp.regression` and `rp.regression2` example scripts.

References

rpanel: Simple interactive controls for R functions using the tcltk package. Journal of Statistical Software, 17, issue 9.

Examples

```

if (interactive()) {
  data(CofE)
  attach(CofE)
  rp.regression(cbind(Employ, Attend), Giving)
}

```

gullweight

*The weights of herring gulls captured at different times of year***Description**

These data are part of a large sample collected by Prof. P. Monaghan of the University of Glasgow in a study of the weight changes in herring gulls throughout the year. Some birds were caught in June (coded as month 1) and others in December (month 2). Since weight is dependent on the size of the bird this information is recorded in the form of the head and bill length, `hab` (in mm), the distance from the back of the head to the tip of the bill.

The data are used in the `rp.ancova` example script.

References

rpanel: Simple interactive controls for R functions using the tcltk package. Journal of Statistical Software, 17, issue 9.

Examples

```
if (interactive()) {
  data(gullweight)
  attach(gullweight)
  rp.ancova(hab, weight, month)
}
```

luthor

Data for the rp.rmpplot example script

Description

These data, reported by Raz(1989, Biometrics 54, 851-71) refer to an experiment which compared the concentrations of leutinizing hormone (LH) in 16 suckled and 16 nonsuckled cows. Measurements were made daily from day 1 through to day 4 postpartum, and twice daily from day 5 through to day 10 postpartum. The cows were ovariectomised on day 5 postpartum.

The first column of the dataset defines the group (1 - non-suckled, 2 - suckled) while the remaining columns give the LH values at the successive recording times.

The data are used in the [rp.rmpplot](#) example script.

References

rpanel: Simple interactive controls for R functions using the tcltk package. Journal of Statistical Software, 17, issue 9.

Examples

```
if (interactive()) {
  data(luthor)
  LH <- luthor[,2:16]
  gp <- factor(luthor[,1])
  times <- c(1:5, (5+(1:10)/2))
  rp.rmpplot(log(LH), fac = gp, timept = times)
}
```

river

Temperature and DO threshold in the River Clyde

Description

These data record the water temperature at a sampling station on the River Clyde, together with an indicator of whether (1) or not (0) the concentration of dissolved oxygen fell below the threshold of 5 percent.

The data are used in the [rp.logistic](#) example script.

The data were kindly provided by the Scottish Environment Protection Agency, with the assistance of Dr. Brian Miller.

References

rpanel: Simple interactive controls for R functions using the tcltk package. *Journal of Statistical Software*, 17, issue 9.

Examples

```
if (interactive()) {  
  data(river)  
  attach(river)  
  rp.logistic(Temperature, Low)  
}
```

rodent

The mass and speed of quadrupedal rodents

Description

In an investigation of the relationship between mass and speed in animals, Garland (1983) collected information from published articles on these two variables for a large number of different species. These measurements are given below for a variety of four-footed rodents. (The common names of the species are taken from Corbet & Hill (1986).) Notice that the measurements are not all recorded to the same level of accuracy since the results have been collated from the work of a number of different scientists.

The data are used in [rp.cartoons](#).

References

Bowman, A.W. & Robinson, D.R. (1990). *Introduction to Regression and Analysis of Variance: a computer illustrated text*. Bristol: Adam Hilger.

Garland, T. (1983). The relation between maximal running speed and body mass in terrestrial animals. *Journal of the Zoological Society of London*, 199, 155-170.

Corbet, G.B. & Hill, J.E. (1986). A World List of Mammalian Species. 2nd edition. London: British Museum, Natural History.

rpanel: Simple interactive controls for R functions using the tcltk package. Journal of Statistical Software, 17, issue 9.

Examples

```
if (interactive()) {
  data(rodent)
  attach(rodent)
  rp.regression(log(Mass), log(Speed))
}
```

rp.ancova

Interactive analysis of covariance

Description

This function plots a response variable against a covariate, with different groups of data identified by colour and symbol. It also creates a panel which controls the model which is fitted to the data and displayed on the plot.

Usage

```
rp.ancova(x, y, group, panel = TRUE, panel.plot = TRUE, model = "None",
          xlab = deparse(substitute(x)), ylab = deparse(substitute(y)), hscale = NA, v
```

Arguments

x	a vector of covariate values.
y	a vector of response values.
group	a vector of group indicators.
panel	a logical variable which determines whether a panel is created to allow interactive control of the fitted models.
panel.plot	a logical parameter which determines whether the plot is placed inside the panel (TRUE) or the standard graphics window (FALSE). If the plot is to be placed inside the panel then the <code>tkrplot</code> library is required.
model	a character variable defining the model to be fitted, if panel is set to FALSE. The valid values are "None", "Single mean", "Single line", "Parallel lines", and "Different lines".
xlab	a character variable used for the covariate axis label.
ylab	a character variable used for the response axis label.
hscale, vscale	scaling parameters for the size of the plot when panel.plot is set to TRUE. The default values are 1 on Unix platforms and 1.4 on Windows platforms.

Details

Static plots, for printing or other purposes can be created by setting the panel argument to FALSE and specifying the model of interest.

Value

If panel is TRUE, the name of the R panel object is returned. If panel is FALSE, nothing is returned.

References

rp.panel: Simple interactive controls for R functions using the tcltk package. Journal of Statistical Software, 17, issue 9.

Examples

```
if (interactive()) {  
  data(gullweight)  
  attach(gullweight)  
  rp.ancova(hab, weight, month)  
}
```

rp.block

Blocks use of the R console until a panel is closed

Description

This function prevents the R console from accepting further input until a panel is closed. The function has two uses: The first is to keep R active when an R script is run in batch mode. This prevents the R session from terminating until the panel has been closed. The second use is to block the user from further use of the command prompt. There may be circumstances in which it is helpful to do this.

Usage

```
rp.block(panel)
```

Arguments

panel the panel whose closure will lead to termination of rp.block. This may be passed as a panelname string or the panel object itself.

Details

rp.block should usually be the very last function executed in a script, to prevent termination until the panel has been closed. It should not normally be used in interactive mode, except where one wishes to prevent use of the R command line whilst the panel is running.

Value

If the parameter panel is the panelname string the same string is returned. If the panel object is used the altered panel is assigned to both the calling level and panel's environment level.

Warning

Note: This may stop the panel responding to button and other events in MacOS command line.

References

rpanel: Simple interactive controls for R functions using the tcltk package. Journal of Statistical Software, 17, issue 9.

See Also

[rp.control](#)

Examples

```
if (interactive()) {  
  # This function will be called on pressing the button "Simulate".  
  boxp.sim <- function(panel) {  
    boxplot(rnorm(50))  
    panel  
  }  
  # Create an rpanel and add the button "Simulate" to it.  
  panel <- rp.control()  
  rp.button(panel, action = boxp.sim, title = "Simulate")  
  rp.block(panel)  
}
```

rp.button

Button widget for rpanel

Description

This function adds a button to the panel. A nominated function is called when the button is pressed.

Usage

```
rp.button(panel, action, title = deparse(substitute(action)), id = "", parent = win
```

Arguments

panel	the panel in which the button should appear. This may be passed as a panelname string or the panel object itself.
action	the function executed when the button is pressed.
title	the text displayed on the button.
id	the name of the button. This is helpful in allowing possible modification and layout changes which have not been implemented and will be considered later.
parent	this specifies the widget inside which the button should appear. In the current version of rpanel, it should not normally be used.
repeatinterval	the interval between auto-repeats (milliseconds) when the button is held down.
repeatdelay	the time after which the button starts to auto-repeat (milliseconds).
quitbutton	this defaults to FALSE. Set to TRUE this creates a button which will close the window and escape from a rp.block call. Before the window is destroyed the action function will be called.
pos	the layout instructions. Please see the rp.pos example and help for full details.
...	information for pos can be passed individually as additional arguments.

Details

The function `action` should take one argument, which should be the panel. See [rp.grid](#) for details of the grid layout system.

Value

If the parameter `panel` is the `panelname` string the same string is returned. If the panel object is used the altered panel is assigned to both the calling level and panel's environment level.

Warning

The `action` function should return the panel. Without this assignment any widgets added or alterations made to panel parameters within the `action` function will be lost.

References

rpanel: Simple interactive controls for R functions using the tcltk package. Journal of Statistical Software, 17, issue 9.

See Also

[rp.doublebutton](#), [rp.control](#)

Examples

```
if (interactive()) {  
  # This function will be called on pressing the button "Simulate".  
  boxp.sim <- function(panel) {  
    boxplot(rnorm(50))  
    panel  
  }  
  # Create an rpanel and add the button "Simulate" to it.  
  panel <- rp.control()  
  rp.button(panel, action = boxp.sim, title = "Simulate")  
}
```

rp.cartoons

Access to a collection of rpanel illustrations

Description

This function creates a panel with a menu which launches a variety of rpanel illustrations. The function provides a template which can be amended by users to create tailored sets of illustrations.

Usage

```
rp.cartoons()
```

Arguments

None.

Value

Nothing.

References

rpanel: Simple interactive controls for R functions using the tcltk package. *Journal of Statistical Software*, 17, issue 9.

Examples

```
if (interactive()) {  
  rp.cartoons()  
}
```

rp.checkbox *Check Box widget for rpanel*

Description

Adds one or more checkboxes to the panel, to control logical variables.

Usage

```
rp.checkbox(panel, var, action = I, labels = NA, names = labels, title = NA,
           initval = NA, parent = window, pos = NULL, doaction = FALSE, ...)
```

Arguments

panel	the panel in which the checkbox(es) should appear. This may be passed as a panelname string or the panel object itself.
var	the name of the variable within the panel that the checkbox(es) should control.
action	the function to call whenever a checkbox is clicked.
labels	the labels of the checkboxes. This defaults to the name of the variable <code>var</code> plus an identifying integer.
names	the names attached to the elements of <code>var</code> . These provide a helpful means of referring to particular items in multiple checkboxes.
title	the title of the checkbox group. This defaults to the name of the variable <code>var</code> . If there is only one box, and <code>labels</code> is not specified, <code>title</code> is taken as the label.
initval	the initial value for <code>var</code> (optional). The initial value can also be specified in the call to <code>rp.control</code> .
parent	this specifies the widget inside which the checkbox should appear. In the current version, it should not normally be used.
pos	the layout instructions. Please see the <code>rp.pos</code> example and help for full details.
doaction	a logical variable which determines whether the action function is called when the widget is created. The default is <code>FALSE</code> , so that the <code>rp.do</code> function should be called after all widgets have been created, to initialise the state of the panel display.
...	information for <code>pos</code> can be passed individually as additional arguments.

Details

The function `action` should take one argument, which should be the panel to which the checkbox is attached. See `rp.grid` for details of the grid layout system.

Value

If the parameter `panel` is the panelname string the same string is returned. If the panel object is used the altered panel is assigned to both the calling level and panel's environment level.

Warning

The `action` function should return the panel. Without this assignment any widgets added or alterations made to panel parameters within the `action` function will be lost.

References

`rp.panel`: Simple interactive controls for R functions using the `tcltk` package. Journal of Statistical Software, 17, issue 9.

See Also

[rp.radiogroup](#), [rp.control](#)

Examples

```
if (interactive()) {
  plot.hist <- function(panel) {
    with(panel, {
      xlim <- range(c(x, mean(x) + c(-3, 3) * sd(x)))
      if (panel$scbox[3])
        clr <- "lightblue" else clr <- NULL
      hist(x, freq = FALSE, col = clr, xlim = xlim)
      if (panel$scbox[1]) {
        xgrid <- seq(xlim[1], xlim[2], length = 50)
        dgrid <- dnorm(xgrid, mean(x), sd(x))
        lines(xgrid, dgrid, col = "red", lwd = 3)
      }
      if (panel$scbox[2])
        box()
    })
    panel
  }
  x <- rnorm(50)
  panel <- rp.control(x = x)
  rp.checkbox(panel, cbox, plot.hist,
    labels = c("normal density", "box", "shading"), title = "Options")
  rp.do(panel, plot.hist)
}
```

Description

This function shows simulated confidence intervals for the mean of a normal distribution. It also creates a panel which controls the mean and standard deviation of the population, the size of the simulated sample.

Usage

```
rp.ci(mu = 0, sigma = 1, sample.sizes = c(30, 50, 100, 200, 500),
      panel.plot = TRUE, hscale = NA, vscale = hscale)
```

Arguments

`mu`, `sigma` the population mean and standard deviation.

`sample.sizes` the available sample sizes for simulated data.

`panel.plot` a logical parameter which determines whether the plot is placed inside the panel (TRUE) or the standard graphics window (FALSE). If the plot is to be placed inside the panel then the `tkrplot` library is required.

`hscale`, `vscale` scaling parameters for the size of the plot when `panel.plot` is set to TRUE. The default values are 1 on Unix platforms and 1.4 on Windows platforms.

Details

A button is provided to sample repeatedly from the current settings. Confidence intervals which cover the population mean are coloured blue while those which miss are coloured red. Repeated simulations illustrate the property of confidence intervals to capture the true value with probability determined by the confidence level (which here is set to 0.95).

Value

Nothing is returned.

References

rpanel: Simple interactive controls for R functions using the tcltk package. *Journal of Statistical Software*, 17, issue 9.

Examples

```
if (interactive()) {
  rp.ci()
}
```

`rp.clearlines` *Remove lines from an rpanel image*

Description

This function removes line(s) from an rpanel image widget: `rp.clearlines` removes all the lines from an image while `rp.deleteLine` deletes only a given line.

Usage

```
rp.clearlines(panel, image)
```

Arguments

`panel` the panel which contains the image. This may be passed as a panelname string or the panel object itself.

`image` the name of the image within the panel.

Value

If the parameter `panel` is the panelname string the same string is returned. If the panel object is used the altered panel is assigned to both the calling level and panel's environment level.

References

rpanel: Simple interactive controls for R functions using the tcltk package. Journal of Statistical Software, 17, issue 9.

See Also

[rp.image](#), [rp.line](#)

Examples

```
if (interactive()) {
  panel <- rp.control()
  image.file <- file.path(system.file(package = "rpanel"), "images", "gulllmks.gif")
  rp.image(panel, image.file, id = "gulls.image")
  rp.line(panel, gulls.image, 10, 10, 100, 100, color = "green")
  rp.line(panel, gulls.image, 100, 100, 100, 10, color = "blue")
  rp.clearlines(panel, gulls.image)
}
```

rp.control

Create an rpanel

Description

This function creates a panel window into which rpanel widgets can be placed. It also returns, and can set up variables within, the rpanel object.

Usage

```
rp.control(title = "", size = c(100, 100), panelname, realname, aschar = TRUE, ...)
```

Arguments

title	the title of the panel displayed in the banner.
size	a two-element numeric vector specifying width and height of the panel in pixels. If this argument is omitted the size of the panel will adapt to the subsequent addition of widgets.
panelname	the name of the panel at .rpenv. If this is not assigned, then realname is taken.
realname	retained for backward compatibility. If this is not assigned, the realname is assigned automatically using rp.panelname.
aschar	if TRUE (default) the panel's name as a string is returned. If FALSE the panel list itself is returned.
...	additional arguments which are treated as variable initialisations and are stored within the returned rpanel object. For example inserting <code>x=3</code> creates a variable <code>x</code> in the rpanel object with the value 3. Note that the names of these additional arguments should not conflict with those of the main arguments of <code>rp.control</code> .

Details

On some machines the R and Tcl/Tk code can become out of step if a large number of objects, or objects of large size, are passed in `...`. This can lead to the panel failing to size itself appropriately. A solution is to insert a `Sys.sleep(0.5)` instruction immediately after the call to `rp.control`, to ensure that sufficient time is allowed for the initialisation to complete. Clearly, the duration of the sleep time may need to be adjusted on different machines.

Value

Dependent on parameter `ischar`. If `ischar` is TRUE this is the string name of the panel if FALSE this is the list object which defines the panel.

References

rpanel: Simple interactive controls for R functions using the tcltk package. Journal of Statistical Software, 17, issue 9.

See Also

[rp.button](#), [rp.slider](#), [rp.doublebutton](#), [rp.textentry](#), [rp.checkbox](#), [rp.radiogroup](#)

Examples

```
if (interactive()) {
  hist.or.boxp <- function(panel) {
    if (panel$plot.type == "histogram")
      hist(panel$x)
    else
      boxplot(panel$x)
    panel
  }
  panel <- rp.control(x=rnorm(50))
}
```

```
rp.radiogroup(panel, plot.type, c("histogram", "boxplot"), title="Plot type", action = hi

# Try also
# panel <- rp.control(ischar = TRUE) # returns a string ".rpanel1" in panel
# panel <- rp.control(ischar = FALSE) # returns the panel list object itself
}
```

rp.deleteline *Removes a line from an rpanel image*

Description

This removes a previously drawn line which was given an id in rp.line.

Usage

```
rp.deleteline(panel, image, id)
```

Arguments

panel	the panel containing the image. This may be passed as a panelname string or the panel object itself.
image	the image on which the line was drawn.
id	the identifier of the line to be deleted.

Value

If the parameter panel is the panelname string the same string is returned. If the panel object is used the altered panel is assigned to both the calling level and panel's environment level.

References

rpanel: Simple interactive controls for R functions using the tcltk package. *Journal of Statistical Software*, 17, issue 9.

Examples

```
if (interactive()) {
  panel <- rp.control()
  image.file <- file.path(system.file(package = "rpanel"), "images", "gulllmks.gif")
  rp.image(panel, image.file, id = "gulls.image")
  rp.line(panel, gulls.image, 10, 10, 100, 100, color = "green", id="first")
  rp.line(panel, gulls.image, 100, 100, 100, 10, color = "blue", id="second")
  rp.deleteline(panel, gulls.image,"first") # deletes only the first line leaving the verti
}
```

`rp.do`*Runs a user written callback function*

Description

Runs a user written callback function, passing a panel to it as a parameter. This should be used to get the rpanel into its initial state. For instance it is useful when using radiobuttons as these do not automatically call the action function when the controls are first created.

Usage

```
rp.do(panel, action = I)
```

Arguments

<code>panel</code>	the panel to be passed as a parameter to the function. This may be passed as a panelname string or the panel object itself.
<code>action</code>	the function to be executed.

Value

If the parameter `panel` is the panelname string the same string is returned. If the panel object is used the altered panel is assigned to both the calling level and panel's environment level.

References

rpanel: Simple interactive controls for R functions using the tcltk package. Journal of Statistical Software, 17, issue 9.

See Also

[rp.radiogroup](#)

Examples

```
if (interactive()) {
  data.plotfn <- function(panel) {
    if (panel$plot.type == "histogram")
      hist(panel$x)
    else
      if (panel$plot.type == "boxplot")
        boxplot(panel$x)
      else
        plot(density(panel$x))
    panel
  }
  panel <- rp.control(x = rnorm(50))
  rp.radiogroup(panel, plot.type,
    c("histogram", "boxplot", "density estimate"),
```

```

        action = data.plotfn, title = "Plot type")
rp.do(panel, data.plotfn)
}

```

rp.doublebutton *Double-button widget for rpanel*

Description

Adds a widget with '+' and '-' buttons, to increment and decrement a variable.

Usage

```

rp.doublebutton(panel, var, step, title = deparse(substitute(var)), action = I,
  initval = NULL, range = c(NA, NA), log = FALSE, showvalue = FALSE, showvaluewidth = 10,
  repeatinterval = 100, repeatdelay = 100, parent = window, pos = NULL, ...)

```

Arguments

panel	the panel in which the doublebutton should appear. This may be passed as a panelname string or the panel object itself.
var	the name of the variable within the panel that the doublebutton should control.
step	the value by which the variable "var" is incremented or decremented on pressing a button. When log is TRUE this is a factor instead.
action	the function which is called when a button is pressed.
title	the label for the doublebutton. This defaults to the name of var.
parent	this specifies the widget inside which this widget should appear. In the current version, it should not normally be used.
initval	the initial value for var (optional). The initial value can also be specified in the call to <code>rp.control</code> .
range	a 2-element numeric vector containing lower and upper limits for var. Use NA for no limit (upper and/or lower).
log	a logical variable which determines whether the increment (step) is multiplicative or additive.
showvalue	a logical variable which determines whether the present value of "var" is shown between the + and - buttons. This is forced to FALSE when log is TRUE.
showvaluewidth	defines the width of the shown value in characters.
repeatinterval	the interval between auto-repeats (milliseconds) when the button is held down.
repeatdelay	the time after which the button starts to auto-repeat (milliseconds).
pos	the layout instructions. Please see the <code>rp.pos</code> example and help for full details.
...	information for <code>pos</code> can be passed individually as additional arguments.

Details

`action` should be a function of one argument, which should be the panel. The panel can then be manipulated, and data stored in the panel may be used/modified, then the (optionally modified) panel must be returned.

See [rp.grid](#) for details of the grid layout system.

Value

If the parameter `panel` is the `panelname` string the same string is returned. If the panel object is used the altered panel is assigned to both the calling level and panel's environment level.

Warning

The `action` function should return the panel. Without this assignment any widgets added or alterations made to panel parameters within the `action` function will be lost.

Note that setting `log=TRUE` and `showvalue=TRUE` is not allowed. The slider value shown would be incorrect (it wouldn't be the log value) and so `showvalue` is over-ridden and set to `FALSE`. A new widget `rp.label` is under development which would be used in these circumstances.

References

`rppanel`: Simple interactive controls for R functions using the `tcltk` package. *Journal of Statistical Software*, 17, issue 9.

See Also

[rp.radiogroup](#), [rp.control](#)

Examples

```
if (interactive()) {
  density.draw <- function(panel) {
    plot(density(panel$x, bw = panel$h))
    panel
  }
  panel <- rp.control(x = rnorm(50))
  rp.doublebutton(panel, var = h, step = 0.05,
    title = "Density estimate", action = density.draw,
    range = c(0.1, 5), initval=1)
}
```

Description

This function gives access to a sampling scenario which is based on the mapping radioactivity and the calculation of a radionuclide inventory within a water body. (A ‘firth’ is a Scottish term for a long, narrow indentation of the sea coast at the mouth of a river.) Interest lies in nuclides which, on release into a water body, attach (absorb) to sediment in a manner which depends on the sediment particle size. Cobalt-60 and caesium-137 are examples of nuclides which exhibit this behaviour. In this sampling scenario, the map of sediment type is used to define regions of different particle size from which the sediment samples will be collected by grabs from a boat. The presence of strata therefore has to be considered, as the different types of material on the sea bed may affect the mean values of the measurements taken.

The function displays a map and gives graphical control over a variety of sampling strategies. Once the user has drawn a sample, some simple predictions over the whole firth can be produced. The `geoR` package is used to construct these predictions.

Usage

```
rp.firth(hscale = NA, col.palette = rev(heat.colors(40)), col.se = "blue", file = N
        parameters = NA, sleep = 0.5)
```

Arguments

<code>hscale</code>	a scaling parameter which expands (>1) or contracts (<1) the size of the plot within the panel. This can be useful for projection onto a screen, for example. The vertical scale is set to the same value as the horizontal scale, to ensure that the plot is square. The default values are 1 on Unix platforms and 1.4 on Windows platforms.
<code>col.palette</code>	the colour palette used to display the predicted and true spatial surfaces.
<code>col.se</code>	the colour used to draw the standard error contours on the predicted surface.
<code>file</code>	the name of a file to which the sampled data will be written.
<code>parameters</code>	a list which can be used to change the parameters which control the simulated measurement data.
<code>sleep</code>	the duration in seconds of a pause while the necessary internal information is loaded into the panel. See Details.

Details

The use of the function is discussed in the paper referenced below.

Once the data have been sampled, a data file may be saved for further analysis external to the `rp.firth` function, using the `file` argument. A convenient way of saving to the current working directory, for example to a file named `firth.dmp`, is to set the `file` argument to `file.path(getwd(), "firth.dmp")`. The `load` function can then be applied to the saved file to create an object called

mururoa.data, which is a three-column matrix with the x and y locations in columns 1 and 2 and the observed values in column 3.

On some machines the R and Tcl/Tk code can become out of step because of the time taken to initialise panel with the large amount of internal information required for plotting. The `sleep` argument allows a pause for this to be completed before further `rpanel` instructions are executed. If the `rpanel` window displays with very small size, try increasing the value of `sleep`.

Value

Nothing.

References

Bowman, A.W., Crawford, E., Alexander, G. Gibson and Bowman, R.W. (2007). `rpanel`: Simple interactive controls for R functions using the `tcltk` package. *Journal of Statistical Software*, 17, issue 9.

Bowman, A.W., Gibson, I., Scott, E.M. and Crawford, E. (2008). *Interactive Teaching Tools for Spatial Sampling*. Technical report, available from the `rpanel` web page at www.stats.gla.ac.uk/~rpanel.

See Also

[rp.mururoa](#), [rp.mururoa](#)

Examples

```
if (interactive()) {  
  rp.firth()  
}
```

rp.geosim

Interactive visualisation of spatially correlated random fields

Description

This function allows Gaussian random fields to be simulated and visualised, using graphical controls for a variety of parameter settings.

Usage

```
rp.geosim(max.Range = 0.2, max.pSill = 1, max.Nugget = 1, max.Kappa = 10,  
          min.ngrid = 10, max.ngrid = 25, hscale = NA, vscale = hscale,  
          col.palette = terrain.colors(40), sleep = 0.5)
```

Arguments

<code>max.Range</code> , <code>max.pSill</code> , <code>max.Nugget</code>	the maximum values of the range, sill and nugget parameters. These define the end-points of the corresponding slider scales.
<code>max.Kappa</code>	The maximum value of the kappa parameter in the Matern family of spatial covariance functions.
<code>min.ngrid</code> , <code>max.ngrid</code>	the minimum and maximum values of the grid size for sampling points.
<code>hscale</code> , <code>vscale</code>	horizontal and vertical scaling factors for the size of the plots. It can be useful to adjust these for projection on a screen, for example. The default values are 1 on Unix platforms and 1.4 on Windows platforms.
<code>col.palette</code>	the colour palette used to display the random fields.
<code>sleep</code>	the duration in seconds of a pause while the necessary internal information is loaded into the panel. See Details.

Details

The aim of the tool is to allow the generation of repeated simulated fields without the distraction of re-executing code explicitly. This can help to gain an intuitive understanding of the nature of spatial data. In particular, interactive control of parameters can help greatly in understanding the meaning and effects of parameter values. Nugget effects can be added and sampled points displayed. Two-dimensional contour plots are produced. Three-dimensional plots are also produced if the `rgl` package is available.

The use of the function is discussed in the paper referenced below.

On some machines the R and Tcl/Tk code can become out of step because of the time taken to initialise panel with the large amount of internal information required for plotting. The `sleep` argument allows a pause for this to be completed before further `rpanel` instructions are executed. If the `rpanel` window displays with very small size, try increasing the value of `sleep`.

Value

None.

References

- Adler, D. (2005). `rgl`: 3D Visualization Device System (OpenGL). <http://CRAN.R-project.org>.
- Diggle, P.J. and Ribeiro, P.J. (2008). *Model-based Geostatistics*. Springer, New York.
- Bowman, A.W., Crawford, E., Alexander, G. Gibson and Bowman, R.W. (2007). `rpanel`: Simple interactive controls for R functions using the `tcltk` package. *Journal of Statistical Software*, 17, issue 9.
- Bowman, A.W., Gibson, I., Scott, E.M. and Crawford, E. (2008). *Interactive Teaching Tools for Spatial Sampling*. Technical report, available from the `rpanel` web page at www.stats.gla.ac.uk/~rpanel.

See Also

[rp.firth](#), [rp.mururoa](#)

Examples

```
if (interactive()) {
  rp.geosim()
}
```

rp.grid

Define a subsidiary grid within an rpanel

Description

A subsidiary grid is defined at a specified row and column location within an rpanel.

Usage

```
rp.grid(panel, name, pos, bg = NULL, parent = window, ...)
```

Arguments

panel	the panel in which the slider appears. This may be passed as a panelname string or the panel object itself.
name	a string defining the name of the grid.
pos	a list containing components which define various features of the grid; see the help information on "grid" mode in rp.pos , for more information. These components can also be specified as additional arguments in ...
bg	a character variable defining a background colour. (This is not the same as colours in R, but simple colours are available.)
parent	this specifies the widget inside which the grid should appear. In the current version of rpanel, it should not normally be used.
...	any other arguments will be assumed to refer to components of pos; see the help information on "grid" mode in rp.pos for details.

Details

The role of this function is to specify a subsidiary grid at a particular row and column position of the parent grid. Nesting of grids within grids is permitted. See the help information on "grid" mode in [rp.pos](#) for a description of the settings of the pos argument.

Value

If the parameter panel is the panelname string the same string is returned. If the panel object is used the altered panel is assigned to both the calling level and panel's environment level.

References

rpanel: Simple interactive controls for R functions using the tcltk package. *Journal of Statistical Software*, 17, issue 9.

Examples

```
if (interactive()) {
  rp.pos("grid")
}
```

rp.gulls

STEPS module: the Birds and the Bees

Description

The function launches a panel which contains an image of a herring gull. With this bird, sex cannot easily be identified by visual inspection. The user is invited to identify length measurements, defined by pairs of landmarks, which will enable males and females to be identified.

Usage

```
rp.gulls(df.name = "gulls", panel.plot = TRUE, sleep = 0.5)
```

Arguments

df.name	a string defining the name of the dataframe to which collected measurements will be added.
panel.plot	whether to plot or not.
sleep	the duration in seconds of a pause while the necessary internal information is loaded into the panel. See Details.

Details

The panel contains an image with landmarks indicated by yellow dots. When the user clicks two landmarks, a length measurement is indicated by a coloured line. The ‘Collect data’ button can be clicked to request that this measurement is collected, on a database of birds whose sex is known. If the measurement is a valid and useful one, it is added to the named dataframe, which is available for inspection and analysis. If the measurement is invalid or not useful, an appropriate message is given in a pop-up window.

On some machines the R and Tcl/Tk code can become out of step because of the time taken to initialise panel with the large amount of internal information required for plotting. The `sleep` argument allows a pause for this to be completed before further `rpanel` instructions are executed. If the `rpanel` window displays with very small size, try increasing the value of `sleep`.

Value

the name of the panel created.

References

`rpanel`: Simple interactive controls for R functions using the `tcltk` package. *Journal of Statistical Software*, 17, issue 9.

Examples

```
if (interactive()) {
  rp.gulls()
}
```

rp.image

Placement of an image within a rpanel

Description

An image is placed inside a panel. When the image is clicked the action function is called with the x and y coordinates of the clicked position.

Usage

```
rp.image(panel, filename, action = NA, mousedrag = NA, mouseup = NA,
         id = "img", parent = window, pos = NULL, ...)
```

Arguments

panel	the panel in which the image should appear. This may be passed as a panelname string or the panel object itself.
filename	the name of the file where the image is located.
action	the function which is called when the image is clicked.
mousedrag	the function which is called when the mouse is dragged.
mouseup	the function which is called when the mouse is released.
id	the name of the image.
parent	this specifies the widget inside which the image should appear. In the current version of rpanel, it should not normally be used.
pos	the layout instructions. Please see the rp.pos example and help for full details.
...	information for pos can be passed individually as additional arguments.

Details

The function `action` should take three arguments, the panel and the coordinates x and y where the image was clicked. At present only GIF images are supported.

See [rp.grid](#) for details of the grid layout system.

Value

If the parameter `panel` is the panelname string the same string is returned. If the panel object is used the altered panel is assigned to both the calling level and panel's environment level.

Warning

The `action` function should return the panel. Without this assignment any widgets added or alterations made to panel parameters within the `action` function will be lost.

References

rpanel: Simple interactive controls for R functions using the tcltk package. Journal of Statistical Software, 17, issue 9.

Examples

```
if (interactive()) {
  gulls.click <- function(panel, x, y) {
    print(c(x, y))
    panel
  }
  panel <- rp.control()
  image.file <- file.path(system.file(package = "rpanel"), "images", "gulllmks.gif")
  rp.image(panel, image.file, id = "gulls.image", action = gulls.click)
}
```

 rp.likelihood

Interactive inspection of one- or two-parameter likelihood surfaces

Description

This function plots a likelihood surface for a model with one or two parameters. It also creates a panel which allows the maximum likelihood estimate, a confidence region and other objects of interest to be added to the plot. For one parameter models, the `tkrplot` package is required. For two-parameter models the `rgl` package is required.

Usage

```
rp.likelihood(loglik.fn, data, theta.low, theta.high, form = "log-likelihood",
             hscale = NA, vscale = hscale)
```

Arguments

<code>loglik.fn</code>	This should be either the name of a function, with arguments <code>theta</code> and <code>data</code> , or R code, in text form, which evaluates the log-likelihood function. The latter form allows simple R expressions such as <code>sum(log(dexp(data, theta)))</code> or <code>sum(log(dgamma(data, theta[1], theta[2])))</code> to be used to define the log-likelihood.
<code>data</code>	an object which contains the data. This will be referred to in likelihood contributions.
<code>theta.low</code>	a vector of length one or two which defines the lower limit(s) of the parameter values for initial plotting.

theta.high	a vector of length one or two which defines the upper limit(s) of the parameter values for initial plotting.
form	a text variable which determines whether the likelihood or log-likelihood function is to be plotted. This applies only to one-parameter models. With two-parameter models, only the log-likelihood is plotted.
hscale, vscale	scaling parameters for the size of the plot when there is one covariate. The default values are 1 on Unix platforms and 1.4 on Windows platforms.

Details

The interactive controls allow a variety of aspects of the plots to be altered. This is intended to allow students and lecturers to explore likelihood surfaces in a manner which promotes an intuitive understanding of the concepts involved.

In the case of one parameter, the vertical axes of the (log-)likelihood plot can be clicked and grabbed to alter the plotting region interactively. This can be useful, in particular, in identifying the maximum likelihood estimator graphically.

Value

Nothing is returned.

References

rpanel: Statistical cartoons in R. (http://www.mathstore.ac.uk/headocs/doc.php?doc=7403_bowman_a_statsr.pdf)

rpanel: Simple interactive controls for R functions using the tcltk package. Journal of Statistical Software, 17, issue 9.

Examples

```
if (interactive()) {
  data(aircond)
  rp.likelihood("sum(log(dexp(data, theta)))", aircond, 0.005, 0.03)
  rp.likelihood("sum(log(dgamma(data, theta[1], theta[2])))",
    aircond, c(0.3, 0.005), c(3, 0.06))
}
```

rp.line

Draws a line on an rpanel image

Description

This draws a line connecting the pixel locations x_1, y_1 to x_2, y_2 on the specified image. The colour and width of the line can be controlled.

Usage

```
rp.line(panel, image, x1, y1, x2, y2, ..., color = "black", width = 2, id = 'rpline')
```

Arguments

panel	the panel containing the image. This may be passed as a panelname string or the panel object itself.
image	the image on which the line should be drawn.
x1	the horizontal first position of start of the line in pixel co-ordinates.
y1	the vertical first position of start of the line in pixel co-ordinates.
x2	the horizontal final position of end of the line in pixel co-ordinates.
y2	the vertical final position of end of the line in pixel co-ordinates.
...	any further parameters.
color	the colour of the line. The default is "black".
width	the width of the line. The default is 2.
id	the identifier of the line created.

Details

The function `action` should take one argument, which should be the panel to which the line is attached.

Value

If the parameter `panel` is the panelname string the same string is returned. If the panel object is used the altered panel is assigned to both the calling level and panel's environment level.

References

rpanel: Simple interactive controls for R functions using the tcltk package. *Journal of Statistical Software*, 17, issue 9.

See Also

[rp.tkrplot](#), [rp.image](#)

Examples

```
if (interactive()) {
  click.capture <- function(panel,x,y) {
    if (is.null(panel$x)) {
      panel$x <- as.numeric(x)
      panel$y <- as.numeric(y)
    }
    else {
      rp.line(panel, gulls.image, panel$x, panel$y, as.numeric(x), as.numeric(y), width=3,
        panel$x <- as.numeric(x)
```

```

        panel$y <- as.numeric(y)
    }
    panel
}
gulls.panel <- rp.control()
image.file <- file.path(system.file(package = "rpanel"), "images", "gulllmks.gif")
rp.image(gulls.panel, image.file, id = "gulls.image", action = click.capture)
}

```

rp.listbox

*Listbox for a panel***Description**

This function adds a listbox to the panel. When an item is pressed, a variable is set and an action function is called.

Usage

```
rp.listbox(panel, var, vals, labels = vals, rows = length(vals), initval = vals[1],
           parent = window, pos = NULL, title = deparse(substitute(var)), action =
```

Arguments

panel	the panel in which the listbox should appear. This may be passed as a panelname string or the panel object itself.
var	the name of the variable whose value is set by the listbox.
vals	the values of <code>var</code> used by the listbox.
labels	the labels for values of <code>var</code> offered by the listbox.
rows	the number of rows in the list. This defaults to the number of labels. If the number of labels is greater than the number of rows the listbox will be displayed with a scrollbar.
initval	the initial value of <code><var></code> (optional). The initial value can also be specified in the call to <code>rp.control</code> .
parent	this specifies the widget inside which the listbox should appear. In the current version of <code>rpanel</code> , it should not normally be used.
pos	the layout instructions. Please see the <code>rp.pos</code> example and help for full details.
title	the label for the listbox.
action	the function which is called when an item is chosen.
...	information for <code>pos</code> can be passed individually as additional arguments.

Details

The function `action` should take one argument, which should be the panel to which the listbox is attached.

See `rp.grid` for details of the grid layout system.

Value

If the parameter `panel` is the `panelname` string the same string is returned. If the `panel` object is used the altered panel is assigned to both the calling level and `panel`'s environment level.

Warning

The `action` function should return the `panel`. Without this assignment any widgets added or alterations made to `panel` parameters within the `action` function will be lost.

References

`rpanel`: Simple interactive controls for R functions using the `tcltk` package. Journal of Statistical Software, 17, issue 9.

See Also

[rp.checkbox](#), [rp.control](#)

Examples

```
if (interactive()) {
  data.plotfn <- function(panel) {
    if (panel$plot.type == "histogram")
      hist(panel$x)
    else
      if (panel$plot.type == "boxplot")
        boxplot(panel$x)
      else
        plot(density(panel$x))
    panel
  }
  panel <- rp.control(x = rnorm(50))
  rp.listbox(panel, plot.type,
    c("histogram", "boxplot", "density estimate"),
    action = data.plotfn, title = "Plot type")
}
```

rp.logistic

Interactive display of logistic regression with a single covariate

Description

The function `rp.logistic` plots a binary or binomial response variable against a single covariates and creates a panel which controls the position of a logistic curve and allows a logistic regression to be fitted to the data and displayed on the plot.

Usage

```
rp.logistic(x, y, xlab = NA, ylab = NA, panel.plot = TRUE, line.showing = TRUE,  
           hscale = NA, vscale = hscale)
```

Arguments

x	a vector of covariate values.
y	a vector of response values with two levels, or a two-column matrix whose first column is the number of ‘successes’ and the second column is the number of ‘failures’ at each covariate value.
xlab	a character variable used for the covariate axis label.
ylab	a character variable used for the response axis label.
panel.plot	a logical variable which determines whether the plot is placed inside the control panel.
line.showing	a logical value determining whether a regression line is shown on the plot when the function starts.
hscale, vscale	horizontal and vertical scaling factors for the size of the plots. It can be useful to adjust these for projection on a screen, for example. The default values are 1 on Unix platforms and 1.4 on Windows platforms.

Details

The control panel allows a logistic regression line to be drawn on the plot and the intercept and slope of the linear predictor altered interactively. The fitted logistic regression can also be displayed.

If `y` is a vector of responses with two values, these are treated as a factor which is then converted to the (0,1) scale by `as.numeric`.

The values of the response variable can be ‘jittered’.

Value

Nothing is returned.

References

rpanel: Simple interactive controls for R functions using the tcltk package. *Journal of Statistical Software*, 17, issue 9.

See Also

[rp.regression](#)

Examples

```
if (interactive()) {
  data(river)
  attach(river)
  rp.logistic(Temperature, Low)
}
```

rp.menu

Top level menu for a panel

Description

This function adds a menu to the top of the panel window. When a menu item is selected, a variable is set and an action function is called.

Usage

```
rp.menu(panel, var, labels, initval = NULL, parent = window, action = I, ...)
```

Arguments

panel	the panel to which the menu should be attached should appear. This may be passed as a panelname string or the panel object itself.
var	the name of the variable whose value is set by the menu.
labels	the labels for the menu options. These values are returned through var. The menu is defined by a list of lists of character strings. Each major menu heading should be the first item in the sub-lists with the submenu items listed afterwards in the same list. Please see the example.
initval	the initial value of <var> (optional). The initial value can also be specified in the call to <code>rp.control</code> .
parent	this specifies the widget inside which this widget should appear. In the current version, it should not normally be used.
action	the function which is called when a menu item is chosen.
...	any further parameters.

Details

The function `action` should take one argument, which should be the panel to which the listbox is attached.

The list for a menu consisting of "File" and "Edit" only would be defined thus;
`list(list("File"), list("Edit"))`

The list for a menu consisting of "File" with subitem "Quit" and "Edit" with subitems "Copy", "Cut" and "Paste" would be defined thus;
`list(list("File", "Quit"), list("Edit", "Copy", "Cut", "Paste"))`

Value

If the parameter `panel` is the `panelname` string the same string is returned. If the `panel` object is used the altered panel is assigned to both the calling level and `panel`'s environment level.

Warning

The `action` function should return the panel. Without this assignment any widgets added or alterations made to panel parameters within the `action` function will be lost.

The action function must be defined before the `rp.menu` statement as it relies on the function already existing.

References

`rp.panel`: Simple interactive controls for R functions using the `tcltk` package. Journal of Statistical Software, 17, issue 9.

See Also

[rp.checkbox](#), [rp.control](#)

Examples

```
if (interactive()) {
  a <- rp.control()
  # The action function has to come first so that it already exists for rp.menu,
  # as it creates the callback functions on the fly it requires action to already
  # be defined.
  domenu <- function(panel) {
    rp.messagebox(panel$menuchoice, title = "You chose")
    panel
  }
  rp.menu(a, menuchoice, labels=list(list("File", "Quit"),
    list("Edit", "Copy", "Cut", "Paste")), action=domenu)
}
```

rp.messagebox	<i>Displays a message</i>
---------------	---------------------------

Description

This function displays a message in a pop-up window.

Usage

```
rp.messagebox(..., title="rpanel Message")
```

Arguments

... parameters containing the message to be displayed.
 title the title for the message window.

Details

The pop-up window remains displayed and no other action can be taken, until the 'ok' button is pressed.

Value

None.

References

rpanel: Simple interactive controls for R functions using the tcltk package. *Journal of Statistical Software*, 17, issue 9.

See Also

[rp.control](#)

Examples

```
if (interactive()) {
  # This is not run as RCMD CHECK will be suspended until OK is clicked ...
  rp.messagebox("Click OK to continue.", title = "Test message")
}
```

 rp.mururoa

Sampling in Mururoa Atoll

Description

This function is based on a real sampling study on the effects of nuclear experiments conducted between 1966 and 1996 in the South Pacific, at the atolls of Mururoa and Fangataufa, (Report by International Advisory Committee, IAEA, 1998). As part of the assessment of subsequent radiological conditions, both terrestrial and aquatic samples were collected and assayed for activities due to strontium-90, caesium-137, plutonium and tritium. The sampling scenario in the function is based on water sampling by boat in the Mururoa atoll. A graphical control panel allows users to select sampling points. Once the user has drawn a sample, some simple predictions over the whole atoll can be produced.

Usage

```
rp.mururoa(hscale = NA, col.palette = rev(heat.colors(40)), col.se = "blue", file =
  parameters = NA, sleep = 0.5)
```

Arguments

<code>hscale</code>	a scaling parameter which expands (>1) or contracts (<1) the size of the plot within the panel. This can be useful for projection onto a screen, for example. The vertical scale is set to the same value as the horizontal scale, to ensure that the plot is square. The default values are 1 on Unix platforms and 1.4 on Windows platforms.
<code>col.palette</code>	the colour palette used to display the predicted and true spatial surfaces.
<code>col.se</code>	the colour used to draw the standard error contours on the predicted surface.
<code>file</code>	the name of a file to which the sampled data will be written.
<code>parameters</code>	a list which can be used to change the parameters which control the simulated measurement data.
<code>sleep</code>	the duration in seconds of a pause while the necessary internal information is loaded into the panel. See Details.

Details

The panel controls allow the user to experiment with random and systematic sampling, with further control of the alignment and patterns of points in the systematic case. The number of points can also be selected. When a sample is taken, simulated data are generated. Some further controls allow predicted surfaces and standard errors to be displayed, using different types of trend functions. The `geoR` package is used to construct these predictions. The true simulated surface can also be displayed, to indicate the success of the predictions.

Once the data have been sampled, a data file may be saved for further analysis external to the `rp.mururoa` function, using the `file` argument. A convenient way of saving to the current working directory, for example to a file named `mururoa.dmp`, is to set the `file` argument to `file.path(getwd(), "mururoa.dmp")`. The `load` function can then be applied to the saved file to create an object called `mururoa.data`, which is a three-column matrix with the `x` and `y` locations in columns 1 and 2 and the observed values in column 3.

On some machines the R and Tcl/Tk code can become out of step because of the time taken to initialise panel with the large amount of internal information required for plotting. The `sleep` argument allows a pause for this to be completed before further `rpanel` instructions are executed. If the `rpanel` window displays with very small size, try increasing the value of `sleep`.

Value

None.

References

`rpanel`: Simple interactive controls for R functions using the `tcltk` package. *Journal of Statistical Software*, 17, issue 9.

See Also

[rp.firth](#)

Examples

```
if (interactive()) {
  rp.mururoa()
}
```

 rp.normal

Interactive fitting of a normal distribution

Description

This function plots a histogram of a sample of data and creates a panel which controls the mean and standard deviation of the normal distribution which is fitted to the data and displayed on the plot.

Usage

```
rp.normal(y, ylab = deparse(substitute(y)), panel.plot = TRUE, hscale = NA, vscale = NA)
```

Arguments

y	a vector of data.
ylab	a character variable used for the histogram axis label.
panel.plot	a logical parameter which determines whether the plot is placed inside the panel (TRUE) or the standard graphics window (FALSE). If the plot is to be placed inside the panel then the tkrplot library is required.
hscale, vscale	scaling parameters for the size of the plot when panel.plot is set to TRUE. The default values are 1 on Unix platforms and 1.4 on Windows platforms.

Details

The interactive controls allow a normal density curve to be added to the histogram, with double-buttons used to control the values of the normal mean and standard deviation. The fitted normal density based on the sample mean and standard deviation can also be displayed.

Value

Nothing is returned.

References

rpanel: Simple interactive controls for R functions using the tcltk package. *Journal of Statistical Software*, 17, issue 9.

Examples

```
if (interactive()) {  
  y <- rnorm(50, mean = 10, sd = 0.5)  
  rp.normal(y)  
}
```

rp.panel *Returns a panel*

Description

Returns the most recently created panel or a named (by passing the name as a string parameter) panel.

Usage

```
rp.panel(panelname = rp.panelname(new = FALSE))
```

Arguments

panelname optional string parameter. If set the panel of that name is returned, if not set the most recently created panel is returned.

Value

If panelname is set the panel of that name is returned, if not set the most recently created panel is returned.

Warning

Note: returning of the most recent panel may fail when running R on a Windows machine in DOS. A warning is contained within the function.

References

rpanel: Simple interactive controls for R functions using the tcltk package. Journal of Statistical Software, 17, issue 9.

See Also

[rp.control](#)

Examples

```
if (interactive()) {  
  # create a panel - will be created in .rpenv as "newpanel"  
  rp.control(realname = "newpanel")  
  # creates the panel, but does not return a handle to it - created as ".rpanel2"  
  rp.control()  
  # pick up the first panel  
  panel2 <- rp.panel("newpanel")  
  # gets a handle to the latest panel created  
  panel <- rp.panel()  
}
```

rp.panelname	<i>Automatic generation of a panel name</i>
--------------	---

Description

This creates the automatically assigned internal 'realname' parameter of a panel. 'realname' is the internally used name of the panel used by callback functions. In rpanel these will always have the form 'rpanelxxxxxxx' where 'xxxxxxx' is generated randomly whenever a panel is created.

Usage

```
rp.panelname(new = TRUE)
```

Arguments

`new` optional parameter. When set to FALSE this will return the most recently created panel name.

Value

A string of the form 'rpanelx', where x is an integer.

Warning

Note: returning of the most recent panel may fail when running R on a Windows machine in DOS. A warning is contained within the function.

References

rpanel: Simple interactive controls for R functions using the tcltk package. Journal of Statistical Software, 17, issue 9.

See Also

[rp.control](#)

Examples

```

if (interactive()) {
  hist.draw <- function(panel) {
    hist(rnorm(50))
    panel
  }
  hist.sample <- function() {
    panel.name <- rp.panelname()
    rp.control(realname = panel.name)
    rp.button(panel.name, title = "Sample", action = hist.draw)
  }
  hist.sample()
}

```

rp.plot3d

*Interactive display of a plot of three variables***Description**

This function produces a scatterplot of three variables, using the rgl package for three-dimensional display.

Usage

```

rp.plot3d(x, y, z, xlab = NA, ylab = NA, zlab = NA,
          axes = TRUE, new.window = TRUE, type = "p", size = 3, col = "red",
          xlim = NA, ylim = NA, zlim = NA, plot = TRUE, ...)

```

Arguments

<code>x, y, z</code>	vectors of observed values.
<code>xlab</code>	a character variable used for the first axis label.
<code>ylab</code>	a character variable used for the second axis label.
<code>zlab</code>	a character variable used for the third axis label.
<code>axes</code>	a logical variable determining whether the axes are shown.
<code>new.window</code>	a logical variable which determines whether a new window is opened (TRUE) or the current plot is clear and the new plot is drawn in the existing window (FALSE).
<code>type</code>	a character variable controlling the type of plotting. If the value is set to "n", the points are not plotted.
<code>size</code>	the size of the plotted points.
<code>col</code>	the colour of the plotted points.
<code>xlim</code>	the plotting range for the first variable.
<code>ylim</code>	the plotting range for the second variable.

zlim	the plotting range for the third variable.
plot	a logical variable which determines whether a plot is drawn. It can be useful to set this to FALSE when only the scaling function is required.
...	other rgl parameters which control the appearance of the plotted points.

Details

The plot is produced by appropriate calls to the rgl package. This allows interactive control of the viewing position. Other objects may subsequently be added to the plot by using rgl functions and data which are centred and scaled by the returned values indicated below.

Value

A scaling function is returned to allow further objects to be added to the plot. The function accepts x, y, z vector arguments and returns a list with x, y, z components defining the co-ordinates for plotting. An illustration is given in the example below.

References

rpanel: Simple interactive controls for R functions using the tcltk package. Journal of Statistical Software, 17, issue 9.

See Also

[rp.regression](#)

Examples

```
if (interactive()) {
  x <- rnorm(50)
  y <- rnorm(50)
  z <- rnorm(50)
  scaling <- rp.plot3d(x, y, z, xlim = c(-3, 3))
  # In addition you may add a line to the plot with these two lines;
  # a <- scaling(c(-3,3), c(0,0), c(0,0))
  # lines3d(a$x, a$y, a$z, col = "green", size = 2)
}
```

rp.pos

Positioning controls in an rpanel

Description

This function provides demonstrations of the use of the pos argument in functions to create controls.

Usage

```
rp.pos(layout = "default")
```

Arguments

`layout` the type of panel layout to be demonstrated. Valid options are "default", "pack", "place" and "grid".

Details

The various functions to create controls accept a parameter called `pos` which can be used to specify the layout of the controls. It has various modes of operation and the mode is determined from the type of information provided in the `pos` argument. The different modes are outlined below.

- `default` If `pos` is not specified, controls are arranged in a column with the most recent added to the bottom. Each control is aligned to the left hand side.
- `pack` If `pos` is set to "left", "right", "top" or "bottom" then the control is aligned to the left/right/top/bottom edge of the window. If there is already a control in that position, the new control is placed beside that control, closer to the centre. (This uses Tk's "pack" layout manager.)
- `place` If `pos` is set to a vector of four integer values, these are interpreted as `c(x, y, width, height)` where all dimensions are in pixels. `x` and `y` define the coordinates in from the left-hand side and down from the top respectively. When using this mode of laying out objects, it usually helps to specify the size of the panel in `rp.control`. (This uses Tk's "place" layout manager.)
- `grid` This mode provides greater flexibility in layout. The following arguments can be passed to `pos` in any of the function calls to create controls. Alternatively, `pos` can be passed a list which has these named components.
 - `column` An integer which specifies the column number. Columns count from 0. This is a mandatory field for grids.
 - `row` An integer which specifies the row number. Rows count from 0. This is a mandatory field for grids.
 - `grid` A string which gives the name of the grid the control has to be placed in. This field is optional. If omitted the default grid belonging to the panel is used.
- `columnspan` An integer which specifies how many columns the control should span. Columns are counted to the right from the start column specified by `column`. This field is optional. If omitted one column is assumed.
- `rowspan` An integer which specifies how many rows the control should span. Rows are counted down from the start row specified by `row`. This field is optional. If omitted one row is assumed.
- `width` An integer which specifies the width of the control. For controls with writing (buttons, listboxes etc) this is in characters and for images this is in pixels. This field is optional. If omitted the control is sized horizontally to fill the cell the control is placed within.
- `height` An integer which specifies the height of the control. For controls with writing (buttons, listboxes etc) this is in characters and for images this is in pixels. This field is optional. If omitted the control is sized vertically to fill the cell the control is placed within.
- `sticky` A string which specifies how the control expands to fill the cell. This is a string with any combination of 'n', 'e', 'w', 's', representing north/east/west/south expansions. An empty string assignment (") will centre the control. If the argument is not assigned a value then the control is 'w' (west) aligned by default.

bg Specifies the background colour of the grid. If left blank this defaults to the operating system's standard background colour.

(This uses Tk's "grid" layout manager.)

The "grid" mode of layout should not be mixed with the other modes.

The example below illustrates the use of pos. Try resizing the windows to explore the behaviour.

Value

Nothing is returned.

Warning

This may not work properly on the command line (as opposed to R-GUI) under Mac OS X due to the `rp.block` function.

References

rpanel: Simple interactive controls for R functions using the tcltk package. Journal of Statistical Software, 17, issue 9.

See Also

[rp.control](#)

Examples

```
if (interactive()) {  
  # example function with different parameters for "pos" documentation.  
  rp.pos("default")  
  rp.pos("pack")  
  rp.pos("place")  
  rp.pos("grid")  
}
```

rp.power

Interactive power calculations for a two-sample t-test

Description

This function creates a panel which allows the sample size, population means and common standard deviation to be set. The corresponding power curve is displayed in the graphics window.

Usage

```
rp.power()
```

Arguments

None.

Details

The population parameters and sample size are controlled by doublebuttons. The sample size refer to the total sample size, assuming two groups of equal size. A checkbox allows plots of the population distributions also to be displayed.

Value

the panel object.

References

rpanel: Simple interactive controls for R functions using the tcltk package. Journal of Statistical Software, 17, issue 9.

Examples

```
if (interactive()) {
  rp.power()
}
```

rp.radiogroup	<i>Radiobuttons for a panel</i>
---------------	---------------------------------

Description

This function adds a set of radiobuttons to the panel. When a radiobutton is pressed, a variable is set and an action function is called.

Usage

```
rp.radiogroup(panel, var, values, labels=values, initval = values[1],
              parent = window, pos = NULL, title = deparse(substitute(var)), action)
```

Arguments

panel	the panel in which the radiobuttons should appear. This may be passed as a panelname string or the panel object itself.
var	the name of the variable whose values are set by the buttons.
values	the values that var is allocated by the buttons.
labels	the labels for the radiobuttons.
initval	the initial value for the variable (optional). The initial value can also be specified in the call to rp.control.

parent	this specifies the widget inside which the radiobuttons should appear. In the current version of rpanel, it should not normally be used.
pos	the layout instructions. Please see the rp.pos example and help for full details.
title	the label for the group of radiobuttons.
action	the function which is called when a button is pressed.
...	information for <code>pos</code> can be passed individually as additional arguments.

Details

The function `action` should take one argument, which should be the panel to which the radiobuttons are attached.

See [rp.grid](#) for details of the grid layout system.

Value

If the parameter `panel` is the `panelname` string the same string is returned. If the `panel` object is used the altered panel is assigned to both the calling level and `panel`'s environment level.

Warning

The `action` function should return the panel. Without this assignment any widgets added or alterations made to panel parameters within the `action` function will be lost.

References

rpanel: Simple interactive controls for R functions using the tcltk package. Journal of Statistical Software, 17, issue 9.

See Also

[rp.checkbox](#), [rp.control](#)

Examples

```
if (interactive()) {
  data.plotfn <- function(panel) {
    if (panel$plot.type == "histogram")
      hist(panel$x)
    else
      if (panel$plot.type == "boxplot")
        boxplot(panel$x)
      else
        plot(density(panel$x))
    panel
  }
  panel <- rp.control(x = rnorm(50))
  rp.radiogroup(panel, plot.type,
    c("histogram", "boxplot", "density estimate"),
    action = data.plotfn, title = "Plot type")
}
```

rp.regression *Interactive display of regression with one or two covariates*

Description

The function `rp.regression` plots a response variable against one or two covariates and creates a panel which controls the model which is fitted to the data and displayed on the plot. In the case of two covariates, a three-dimensional display is created. The function `rp.regression2` is retained simply for compatibility with earlier releases of the package.

Usage

```
rp.regression(x, y, ylab = NA, x1lab = NA, x2lab = NA, xlab = NA,
              panel = TRUE, panel.plot = TRUE, hscale = NA, vscale = hscale,
              model = "None", line.showing = TRUE, residuals.showing = FALSE, si
rp.regression2(y, x1, x2, ylab = NA, x1lab = NA, x2lab = NA,
               panel = TRUE, model = "None", residuals.showing = FALSE, size = 3,
```

Arguments

<code>x</code>	a vector or two column matrix of covariate values.
<code>y</code>	a vector of response values.
<code>x1, x2</code>	vectors of covariate values.
<code>ylab</code>	a character variable used for the response axis label.
<code>x1lab</code>	a character variable used for the first covariate axis label.
<code>x2lab</code>	a character variable used for the second covariate axis label.
<code>xlab</code>	a character variable used for the first covariate axis label. This is provided for convenience as a more natural argument name when there is only one covariate.
<code>panel</code>	a logical variable which determines whether a panel is created to allow interactive control of the fitted models. This is relevant only to the case of two covariates.
<code>panel.plot</code>	a logical variable which determines whether the plot is placed inside the control panel. This is relevant only to the case of one covariate.
<code>hscale, vscale</code>	scaling parameters for the size of the plot when there is one covariate and <code>panel.plot</code> is set to <code>TRUE</code> . The default values are 1 on Unix platforms and 1.4 on Windows platforms.
<code>model</code>	a character variable defining the model to be fitted when the function starts. The valid values are "None", the name of the first and second covariates and the combination of these names with an "&". This is relevant only to the case of two covariates.
<code>line.showing</code>	a logical value determining whether a regression line is shown on the plot when the function starts. This is relevant only to the case of one covariates.

<code>residuals.showing</code>	a logical value determining whether the residuals are shown on the plot when the function starts.
<code>size</code>	the size of the plotted points. This is relevant only to the case of two covariates.
<code>col</code>	the colour of the plotted points. This is relevant only to the case of two covariates.

Details

In the case of one covariate, the control panel allows a line to be drawn on the plot and its intercept and slope altered interactively. The residuals and the least squares fitted line can be displayed. When the fitted line is displayed, the effects of moving individual points can be viewed by clicking and dragging.

In the case of two covariates, the plot is constructed with the aid of the `rgl` package for three-dimensional display, through the `rpanel` function [rp.plot3d](#). This display can be rotated and linear models involving one, two or none of the covariates can be displayed. Residuals can also be superimposed.

In the case of two covariates, static plots, for printing or other purposes can be created by setting the panel argument to `FALSE` and specifying `model` and `residuals.showing` as required.

Value

Nothing is returned.

References

`rpanel`: Simple interactive controls for R functions using the `tecltk` package. *Journal of Statistical Software*, 17, issue 9.

See Also

[rp.plot3d](#)

Examples

```
if (interactive()) {
  data(CofE)
  attach(CofE)
  rp.regression(Employ, Giving)
  rp.regression(cbind(Employ, Attend), Giving)
}
```

rp.rmpplot

*Interactive plotting of repeated measurement data***Description**

This function creates a panel which controls the display of data which have a repeated measurement structure across time. Groups, means and standard errors can be displayed. Individual profiles can also be inspected.

Usage

```
rp.rmpplot(y, id = NA, timept = NA, fac = NA, type = "all",
           xlab = NA, ylab = NA, xlabel = NA, add = FALSE,
           lwd = 1, col = NA, lty = NA, panel = TRUE,
           panel.plot = TRUE, hscale = NA, vscale = hscale, ...)
```

Arguments

y	a vector, matrix or dataframe of response data. If y is a matrix or dataframe, the rows should correspond to cases and the columns to the repeated measurements.
id	when y is a vector, id should contain the identifiers for the individual profiles.
timept	when y is a vector, timept should contain the time value associated with each repeated measurement. When y is a matrix or dataframe timept may identify the values associated with the repeated measurements (columns); in this case the default value is the sequence from 1 to the number of repeated measurements.
fac	an optional factor to split the data into groups.
type	when the function is not running in interactive panel mode, this character variable determines the type of plot produced. It can be set to "all", "mean", "mean+bar" or "band". The last option is applicable only when there are two groups of data.
xlab	the x-axis label.
ylab	the y-axis label.
xlabels	labels for the repeated measurements, to be printed on the x-axis.
add	a logical variable which determines whether the repeated measurements graph is added to an existing plot. This is only appropriate when panel = FALSE.
lwd	the width of the lines drawn for each repeated measurements profile.
col	a vector of colours associated with each of the factor levels in fac.
lty	a vector of linetypes associated with each of the factor levels in fac.
panel	a logical variable controlling whether an interactive panel is created.
panel.plot	a logical parameter which determines whether the plot is placed inside the panel (TRUE) or the standard graphics window (FALSE). If the plot is to be placed inside the panel then the tkrplot library is required.

```
hscale, vscale
```

scaling parameters for the size of the plot when `panel.plot` is set to `TRUE`. The default values are 1 on Unix platforms and 1.4 on Windows platforms.

```
...
```

further arguments which will be passed to the `plot` call in the construction of the graph.

Details

This function is designed principally for repeated measurements over time, with common time points for each profile. A set of radiobuttons allows all the individual profiles to be plotted, or summaries in the form of means and two standard errors. A checkbox allows the data to be split into groups identified by the variable `fac`. When there are only two groups, a band can be displayed to indicate time points at which the distance between the observed means is more than two standard errors of the differences between the means.

Value

the name of the panel object.

References

`rpanel`: Simple interactive controls for R functions using the `tcltk` package. *Journal of Statistical Software*, 17, issue 9.

Examples

```
if (interactive()) {
  data(luthor)
  LH   <- luthor[,2:16]
  gp   <- factor(luthor[,1])
  times <- c(1:5, (5+(1:10)/2))
  rp.rmplot(log(LH), fac = gp, timept = times)
}
```

rp.slider

Slider for an rpanel

Description

Add a slider to the panel, to graphically control a numeric variable.

Usage

```
rp.slider(panel, var, from, to, action = I, title = deparse(substitute(var)),
  log = FALSE, showvalue = FALSE, resolution = 0, initval = NULL,
  parent = window, pos = NULL, horizontal = TRUE, ...)
```

Arguments

<code>panel</code>	the panel in which the slider appears. This may be passed as a <code>panelname</code> string or the panel object itself.
<code>var</code>	the name of the variable that the slider controls.
<code>from</code>	the lower limit of the range of values to which the slider can be set.
<code>to</code>	the upper limit of the range of values to which the slider can be set.
<code>log</code>	a logical variable which controls whether the scale of the slider is logarithmic.
<code>showvalue</code>	a logical variable which determines whether the present value of "var" is shown. This is forced to <code>FALSE</code> when <code>log</code> is <code>TRUE</code> .
<code>resolution</code>	the resolution of the slider scale. If > 0 , all values are rounded to an even multiple of this value. The default is 0.
<code>action</code>	the function which is called when the slider is moved.
<code>title</code>	the label of the slider.
<code>initval</code>	the initial value of <code>var</code> (optional). The initial value can also be specified in the call to <code>rp.control</code> .
<code>parent</code>	this specifies the widget inside which the slider should appear. In the current version of <code>rpanel</code> , it should not normally be used.
<code>pos</code>	the layout instructions. Please see the <code>rp.pos</code> example and help for full details.
<code>horizontal</code>	a logical variable determining whether the slider is displayed horizontally (or vertically).
<code>...</code>	information for <code>pos</code> can be passed individually as additional arguments.

Details

The function `action` should take one argument, which should be the panel to which the slider is attached.

See `rp.grid` for details of the grid layout system.

Value

If the parameter `panel` is the `panelname` string the same string is returned. If the panel object is used the altered panel is assigned to both the calling level and panel's environment level.

Warning

The `action` function should return the panel. Without this assignment any widgets added or alterations made to panel parameters within the `action` function will be lost.

Note that setting `log=TRUE` and `showvalue=TRUE` is not allowed. The slider value shown would be incorrect (it wouldn't be the log value) and so `showvalue` is over-ridden and set to `FALSE`. A new widget `rp.label` is under development which would be used in these circumstances.

References

`rpanel`: Simple interactive controls for R functions using the `tcltk` package. *Journal of Statistical Software*, 17, issue 9.

See Also

[rp.radiogroup](#), [rp.control](#)

Examples

```
if (interactive()) {
  density.draw <- function(panel) {
    plot(density(panel$x, bw = panel$h))
    panel
  }
  panel <- rp.control(x = rnorm(50))
  rp.slider(panel, h, 0.5, 5, log = TRUE, action = density.draw)
}
```

rp.tables

Interactive statistical tables

Description

This function launches a panel which allows standard normal, t, chi-squared and F distributions to be plotted, with interactive control of parameters, tail probability and p-value calculations.

Usage

```
rp.tables(panel.plot = TRUE, hscale = NA, vscale = hscale)
```

Arguments

`panel.plot` a logical parameter which determines whether the plot is placed inside the panel (TRUE) or the standard graphics window (FALSE). If the plot is to be placed inside the panel then the tkrplot library is required.

`hscale, vscale`

horizontal and vertical scaling factors for the size of the plot when `panel.plot` is set to TRUE. It can be useful to adjust these for projection on a screen, for example. The default values are 1 on Unix platforms and 1.4 on Windows platforms.

Details

The panel contains radiobuttons to select the standard normal, t, chi-squared or F distributions. Doublebutton are available to control the degrees of freedom. An observed value can be added to the plot, with optional determination of the corresponding p-value. Alternatively, shaded areas corresponding to tail probabilities of specified value can be displayed.

Value

the name of the panel object.

References

rp.panel: Simple interactive controls for R functions using the tcltk package. Journal of Statistical Software, 17, issue 9.

Examples

```
if (interactive()) {
  rp.tables()
}
```

rp.textentry	<i>Text entry boxes for a panel</i>
--------------	-------------------------------------

Description

This function adds one or more boxes which allow text to be entered.

Usage

```
rp.textentry(panel, var, action = I, labels = NA, names = labels, title = NA,
             initval = NA, parent = window, pos = NULL, ...)
```

Arguments

panel	the panel in which the text entry box(es) should appear. This may be passed as a panelname string or the panel object itself.
var	the name of the variable which will be assigned the text entered into the box(es).
action	the function which is called when the text has been entered.
labels	a character string of labels for the text entry boxes.
names	the names attached to the elements of var. These provide a helpful means of referring to particular items in multiple textentry boxes.
title	a title for the group of text entry boxes. If there is only one box, and labels is not specified, title is taken as the label.
initval	the initial value(s) for var (optional). The initial value(s) can also be specified in the call to rp.control.
parent	this specifies the widget inside which the text entry widget should appear. In the current version, it should not normally be used.
pos	the layout instructions. Please see the rp.pos example and help for full details.
...	information for pos can be passed individually as additional arguments.

Details

The function `action` should take one argument, which should be the panel to which the text entry box is attached.

See [rp.grid](#) for details of the grid layout system.

Value

If the parameter `panel` is the `panelname` string the same string is returned. If the `panel` object is used the altered panel is assigned to both the calling level and `panel`'s environment level.

Warning

The `action` function should return the panel. Without this assignment any widgets added or alterations made to panel parameters within the `action` function will be lost.

References

`rpanel`: Simple interactive controls for R functions using the `tcltk` package. *Journal of Statistical Software*, 17, issue 9.

See Also

[rp.control](#)

Examples

```
if (interactive()) {
  plotf <- function(panel) {
    with(panel, {
      pars <- as.numeric(pars)
      xgrid <- seq(0.1, max(c(pars[3], 5), na.rm = TRUE), length = 50)
      dgrid <- df(xgrid, pars[1], pars[2])
      plot(xgrid, dgrid, type = "l", col = "blue", lwd = 3)
      if (!is.na(pars[3])) {
        lines(rep(pars[3], 2), c(0, 0.95 * max(dgrid)), lty = 2, col = "red")
        text(pars[3], max(dgrid), as.character(pars[3]), col = "red")
      }
    })
    panel
  }

  panel <- rp.control(pars = c(5, 10, NA))
  rp.textentry(panel, pars, plotf, labels = c("df1", "df2", "observed"),
    initval = c(10, 5, 3))
  rp.do(panel, plotf)
}
```

rp.tkrplot

rpanel calls for *tkrplot* and *tkrreplot*

Description

These functions call Luke Tierney's `tkrplot` and `tkrreplot` functions to allow R graphics to be displayed in a panel.

Usage

```
rp.tkrplot(panel, name, plotfun, action = NA, mousedrag = NA,
           mouseup = NA, hscale = 1, vscale = 1, parent = window, pos = NULL,
           foreground = NULL, margins = c(0, 0, 0, 0), ...)
rp.tkrreplot(panel, name)
```

Arguments

panel	the panel in which the plot should appear. This may be passed as a panelname string or the panel object itself.
name	the name of the plot. This is subsequently used in tkreplot to specify the plot to be redrawn.
plotfun	the function used to create the plot.
action	the function called when the plot is clicked.
mousedrag	the function called when the mouse is dragged.
mouseup	the function called when the mouse is released.
parent	this specifies the widget inside which the plot should appear. In the current version of rpanel, it should not normally be used.
hscale	horizontal scaling factor to control the width of the plot.
vscale	vertical scaling factor to control the height of the plot.
pos	the layout instructions. Please see the rp.pos example and help for full details.
foreground	the filename of a transparent gif file. This will be overlaid on the tkrplot image.
margins	an integer vector of length 4 giving the margin sizes, in pixels and in the usual order, for the placing of the foreground image.
...	information for <code>pos</code> can be passed individually as additional arguments.

Details

The function `action` should take one argument, which should be the panel to which the tkrplot is attached.

See [rp.grid](#) for details of the grid layout system.

Value

If the parameter `panel` is the panelname string the same string is returned. If the panel object is used the altered panel is assigned to both the calling level and panel's environment level.

Warning

The `action` function should return the panel. Without this assignment any widgets added or alterations made to panel parameters within the `action` function will be lost.

References

rpanel: Simple interactive controls for R functions using the tcltk package. *Journal of Statistical Software*, 17, issue 9.

See Also[rp.image](#)**Examples**

```
if (interactive()) {
  draw <- function(panel) {
    plot(1:20, (1:20)^panel$h)
    panel
  }

  redraw <- function(panel) {
    rp.tkrreplot(panel, tkrep)
    panel
  }

  rpplot <- rp.control(title = "Demonstration of rp.tkrplot", h = 1)
  rp.tkrplot(rpplot, tkrep, draw)
  rp.slider(rpplot, h, action = redraw, from = 0.05, to = 2.00, resolution = 0.05)
}
```

Index

*Topic **dynamic**

- aircond, 3
- CofE, 4
- gullweight, 4
- luthor, 5
- river, 6
- rodent, 6
- rp.ancova, 7
- rp.block, 8
- rp.button, 9
- rp.cartoons, 11
- rp.checkbox, 12
- rp.ci, 13
- rp.clearlines, 14
- rp.control, 15
- rp.deleteline, 17
- rp.do, 18
- rp.doublebutton, 19
- rp.firth, 21
- rp.geosim, 22
- rp.grid, 24
- rp.gulls, 25
- rp.image, 26
- rp.likelihood, 27
- rp.line, 28
- rp.listbox, 30
- rp.logistic, 31
- rp.menu, 33
- rp.messagebox, 34
- rp.mururoa, 35
- rp.normal, 37
- rp.panel, 38
- rp.panelname, 39
- rp.plot3d, 40
- rp.pos, 41
- rp.power, 43
- rp.radiogroup, 44
- rp.regression, 46
- rp.rmpplot, 48

- rp.slider, 49
- rp.tables, 51
- rp.textentry, 52
- rp.tkrplot, 53

*Topic **iplot**

- aircond, 3
- CofE, 4
- gullweight, 4
- luthor, 5
- river, 6
- rodent, 6
- rp.ancova, 7
- rp.block, 8
- rp.button, 9
- rp.cartoons, 11
- rp.checkbox, 12
- rp.ci, 13
- rp.clearlines, 14
- rp.control, 15
- rp.deleteline, 17
- rp.do, 18
- rp.doublebutton, 19
- rp.firth, 21
- rp.geosim, 22
- rp.grid, 24
- rp.gulls, 25
- rp.image, 26
- rp.likelihood, 27
- rp.line, 28
- rp.listbox, 30
- rp.logistic, 31
- rp.menu, 33
- rp.messagebox, 34
- rp.mururoa, 35
- rp.normal, 37
- rp.panel, 38
- rp.panelname, 39
- rp.plot3d, 40
- rp.pos, 41

- rp.power, 43
- rp.radiogroup, 44
- rp.regression, 46
- rp.rmplot, 48
- rp.slider, 49
- rp.tables, 51
- rp.textentry, 52
- rp.tkrplot, 53
- *Topic **package**
 - rpanel-package, 1
- aircond, 3
- CofE, 4
- gullweight, 4
- luthor, 5
- river, 6
- rodent, 6
- rp.ancova, 2, 7
- rp.block, 2, 8
- rp.button, 2, 3, 9, 16
- rp.cartoons, 3, 6, 11
- rp.checkbox, 2, 3, 12, 16, 31, 34, 45
- rp.ci, 2, 13
- rp.clearlines, 2, 14
- rp.control, 2, 3, 9, 10, 13, 15, 20, 31, 34, 35, 38, 39, 43, 45, 51, 53
- rp.deleteline, 2, 17
- rp.do, 2, 18
- rp.doublebutton, 2, 3, 10, 16, 19
- rp.firth, 3, 21, 23, 36
- rp.geosim, 3, 22
- rp.grid, 2, 10, 12, 20, 24, 26, 30, 45, 50, 52, 54
- rp.gulls, 2, 25
- rp.image, 2, 15, 26, 29, 55
- rp.likelihood, 2, 27
- rp.line, 2, 15, 28
- rp.listbox, 2, 30
- rp.logistic, 2, 6, 31
- rp.menu, 33
- rp.messagebox, 2, 34
- rp.mururoa, 3, 22, 23, 35
- rp.normal, 2, 37
- rp.panel, 2, 38
- rp.panelname, 2, 39
- rp.plot3d, 2, 40, 47
- rp.pos, 2, 10, 12, 19, 24, 26, 30, 41, 45, 50, 52, 54
- rp.power, 2, 43
- rp.radiogroup, 2, 3, 13, 16, 18, 20, 44, 51
- rp.regression, 2, 32, 41, 46
- rp.regression2 (*rp.regression*), 46
- rp.rmplot, 2, 5, 48
- rp.slider, 2, 3, 16, 49
- rp.tables, 2, 51
- rp.textentry, 2, 3, 16, 52
- rp.tkrplot, 2, 29, 53
- rp.tkrreplot, 2
- rp.tkrreplot (*rp.tkrplot*), 53
- rpanel (*rpanel-package*), 1
- rpanel-package, 1