

# Package ‘rpanel’

February 20, 2015

**Type** Package

**Title** Simple interactive controls for R using the tcltk library.

**Version** 1.1-3

**Date** 2014-02-20

**Author** Bowman, Bowman, Gibson and Crawford

**Maintainer** Adrian Bowman <adrian.bowman@glasgow.ac.uk>

**Depends** R (>= 2.14), tcltk

**Suggests** tkrplot, rgl, sp, geoR, RandomFields, akima, MASS, denstrip,  
lattice, sm, maps, mgcv

**SystemRequirements** BWidget

**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. Some functions for specific graphical tasks, referred to as 'cartoons', are provided.

**License** GPL (>= 2)

**LazyData** TRUE

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2014-02-28 12:13:07

## R topics documented:

rpanel-package	3
aircond	5
Clyde	6
CofE	6
gullweight	7
luthor	7

poisons	8
river	9
rodent	9
rp.ancova	10
rp.anova	11
rp.block	13
rp.bubbleplot	14
rp.button	15
rp.cartoons	17
rp.checkbox	18
rp.ci	20
rp.clearlines	21
rp.colour.key	22
rp.combo	23
rp.control	25
rp.control.put	26
rp.deleteline	27
rp.do	28
rp.doublebutton	29
rp.firth	31
rp.geosim	33
rp.grid	34
rp.gulls	35
rp.image	36
rp.likelihood	38
rp.line	39
rp.listbox	41
rp.logistic	43
rp.menu	44
rp.messagebox	46
rp.mururoa	47
rp.normal	48
rp.notebook	49
rp.panel	50
rp.plot3d	51
rp.plot4d	53
rp.pos	56
rp.power	57
rp.radiogroup	59
rp.regression	60
rp.rmplot	62
rp.screenresolution	64
rp.slider	65
rp.surface	67
rp.tables	69
rp.text	70
rp.textentry	72
rp.timer	74

rp.tkrplot . . . . .	75
rp.var.get . . . . .	76
rp.var.put . . . . .	77
rp.widget.dispose . . . . .	78
SO2 . . . . .	79
worldbank . . . . .	79

<b>Index</b>	<b>81</b>
--------------	-----------

---

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

---

## Description

**rpanel** provides a set of functions to build simple GUI controls for R functions. Uses include changing a parameter on a graph (and animating it) with a slider, or a "doublebutton", up to more sophisticated mini-applications. In addition to functions which create controls, a number of ‘cartoon’ functions built on these controls are also available.

## Details

Package: rpanel  
 Type: Package  
 Version: 1.1-3  
 Date: 2014-02-20  
 License: GNU

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

### Cartoon functions

[rp.gulls](#): An interactive problem-solving exercise on deciding the sex of a herring gull  
[rp.ci](#): Confidence intervals  
[rp.anova](#): Analysis of variance  
[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.plot4d](#): Interactive display of a plot of four variables  
[rp.spacetime](#): A version of rp.plot4d designed for space-time data  
[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  
`rp.surface`: Displaying the uncertainty in an estimate of a surface

Functions to create individual controls

`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.combo`: adds a combo box to the panel  
`rp.doublebutton`: adds a widget with '+' and '-' buttons, to increment and decrement a variable  
`rp.menu`: adds a menu to the panel  
`rp.text`: adds a text box to the panel  
`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 refreshed in a panel.  
`rp.timer`: executes an action function repeatedly until a condition is satisfied  
`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.var.put`: place an object into the rpanel environment, usually within a panel  
`rp.var.get`: retrieve an object from the rpanel environment, usually from a 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  
`rp.colour.key`: a colour key to associate with a plot

Generally speaking these functions have a parameter, `name`, which is used to later delete or modify a widget.

### Author(s)

E. Crawford & A. Bowman

Maintainer: Adrian Bowman <adrian.bowman@glasgow.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

```
## Not run:
  rp.gulls()

## End(Not run)
```

---

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. They 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

```
## Not run:
  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))

## End(Not run)
```

---

Clyde

*Water quality in the River Clyde*

---

### Description

These data record the water quality, in terms of dissolved oxygen (DO) on a percentage scale, at a number of sampling stations (Station) on the River Clyde. The date (Day, \codeMonth, Year) is also available, along with the day of the year (Doy between 1 and 365) and an identified (id) of the survey on which each measurement was made.

The data are used in the [rp.plot4d](#) 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

```
## Not run:
  with(Clyde, {
    rp.plot4d(cbind(Doy, DO), Station, location.plot = FALSE)
    rp.plot4d(cbind(Station, DO), Doy, location.plot = FALSE)
  })

## End(Not run)
```

---

CofE

*Giving in the Church of England*

---

### 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](#) example script.

### References

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

**Examples**

```
## Not run:
with(CofE, {
  rp.regression(cbind(Employ, Attend), Giving)
})

## End(Not run)
```

---

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**

```
## Not run:
with(gullweight, {
  rp.ancova(hab, weight, month)
})

## End(Not run)
```

---

luthor

*Repeated measurements on leutinizing hormone in cows*

---

**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

```
## Not run:
LH <- luthor[,2:16]
gp   <- factor(luthor[,1])
times <- c(1:5, (5+(1:10)/2))
rp.rmpplot(log(LH), fac = gp, timept = times)

## End(Not run)
```

---

poisons

*Survival times of animals subjected to different poisons and treatment*

---

## Description

These data record the survival times (in units of 10 hours) of animals in a 3 x 4 factorial experiment. Four animals were allocated to each combination of three poisons and four treatments, using a randomisation procedure.

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

The data were reported in the paper by Box and Cox (1964) referenced below.

## References

Box, GEP and Cox, DR (1964), An analysis of transformations. Journal of the Royal Statistical Society Series B - Statistical Methodology, 26, 211-252.

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

## Examples

```
## Not run:
with(poisons, {
  rp.anova(1/stime, treatment, poison)
})

## End(Not run)
```

---

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

```
## Not run:  
rp.logistic(river$Temperature, river$Low)  
  
## End(Not run)
```

---

rodent

*The mass and speed of quadrupedal rodents*

---

### Description

In an investigation of the relationship between mass (kg) and speed (km/hr) in mammals, 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

```
## Not run:
  with(rodent, {
    rp.regression(log(Mass), log(Speed))
  })

## End(Not run)
```

---

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 = NA, model0 = NA,
  xlab, ylab, glab, hscale = NA, vscale = hscale, style = "new")
```

### Arguments

x	a vector of covariate values.
y	a vector of response values.
group	a vector of group indicators. If this is not already a factor it will be converted into one.
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 tkplot library is required.
model, model0	logical vectors of length 4 defining the initial and comparison models to be fitted. The four values determine whether each of the four terms intercept, x, z and x : z appear. This is appropriate only for style = "new".
xlab	a character variable used for the covariate axis label.
ylab	a character variable used for the response axis label.
glab	a character variable used for the group variable 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.
- style a character variable used to determine whether the style of controls in earlier version of the rpanel package is to be used. The value "new" activates the new style and any other value activates the old one.

### 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

Nothing is returned.

### References

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

### Examples

```
## Not run:
  with(gullweight, {
    rp.ancova(hab, weight, month)
  })

## End(Not run)
```

---

rp.anova

*Interactive analysis of variance*

---

### Description

This function plots response data, separated by one or two factors. It also creates a panel which controls the models which can be fitted to the data and displayed on the plot. A comparison model can also be selected and the results of an F-test are displayed graphically.

### Usage

```
rp.anova(y, x, z, model = NA, model0 = NA, ylab = NA, xlab = NA, zlab = NA,
  panel = TRUE, panel.plot = TRUE, hscale = 1.3, vscale = hscale / 1.3)
```

**Arguments**

<code>y</code>	a vector of response values.
<code>x</code>	a factor which splits <code>y</code> into different groups.
<code>z</code>	an optional second factor which splits <code>y</code> into a second set of groups.
<code>model, model0</code>	logical vectors of length 2 or 4, for one or two factors respectively, defining the initial and comparison models to be fitted. For one factor, the two values determine whether each of the terms for the intercept and <code>x</code> appear. For two factors, the four values determine whether each of the four terms intercept, <code>x</code> , <code>z</code> and <code>x:z</code> appear.
<code>ylab</code>	a character name used for the response variable.
<code>xlab</code>	a character name used for the first factor.
<code>zlab</code>	a character variable used for the response axis label.
<code>panel</code>	a logical variable which determines whether a panel is created to allow interactive control of the fitted models.
<code>panel.plot</code>	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.
<code>hscale, vscale</code>	scaling parameters for the size of the plot when <code>panel.plot</code> is set to TRUE.

**Details**

The data are displayed as points superimposed on a density strip created by the **denstrip** package. Selected models are displayed through the fitted values for each group. When a valid comparison model is selected, its fitted values are displayed along with a shaded regions expressing the contribution of the differences between the two sets of fitted values to the F-statistic. The F-test is displayed in graphical form with a density strip to represent the F-distribution and a point to indicate the observed value of the F-statistic.

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

**Value**

Nothing is returned.

**References**

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

**Examples**

```
## Not run:
  with(poisons, {
    rp.anova(1/stime, treatment, poison)
  })

## End(Not run)
```

---

`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`. Note that the panelname needs to have been set appropriately in [rp.control](#).

### 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 argument `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 the MacOS command line.

### References

rpanel: Simple interactive controls for R functions using the tcltk package (<http://www.stats.gla.ac.uk/~adrian/rpanel/>)

### See Also

[rp.control](#)

**Examples**

```
## Not run:
# 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(panelname="panel")
rp.button(panel, action = boxp.sim, title = "Simulate")
rp.block(panel)

## End(Not run)
```

---

rp.bubbleplot

*Animated scatterplot*


---

**Description**

This function produces a scatterplot of two variables, with the values of third and fourth variables represented by size and colour of the plotted points. In addition, the scatterplot is animated over a fifth variable, such as time.

**Usage**

```
rp.bubbleplot(x, y, year, size, col, col.palette = topo.colors(20),
              interpolate = FALSE, fill.in = FALSE, labels = rownames(x),
              hscale = 1, vscale = hscale)
```

**Arguments**

x	a matrix of values, whose columns correspond to time points, to be plotted on the horizontal axis.
y	a matrix of values, whose columns correspond to time points, to be plotted on the vertical axis.
year	a vector of values, usually years, over which the scatterplot will be animated. The values in this vector correspond to the columns of x and y.
size	a vector or matrix of values used to scale the sizes of the plotted points.
col	a vector or matrix of values which will be translated into the colours of the plotted points.
col.palette	the colour palette used to colour the points.
interpolate	a logical variable controlling whether interpolation is used to create data for plotting at year values which do not correspond to an exact values of year.
fill.in	a logical variable which controls whether gaps resulting from missing data are filled in with the largest previous value.

labels            the labels of the plotted points, used to highlight individual points on the scatterplot.

hscale, vscale   scaling parameters for the size of the plot when panel.plot is set to TRUE.

### Details

This plot mimics the plots made famous by Hans Rosling through the Gapminder project (see [www.gapminder.org](http://www.gapminder.org)). This type of plot is also available at the Google Public Data Explorer ([www.google.com/publicdata/directory](http://www.google.com/publicdata/directory)). The aim of this function is to make this type of plot available directly from within R. The controls provide a slider or button for animation, plus a list of country names for individual identification.

### Value

Nothing is returned.

### References

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

### Examples

```
## Not run:
  rp.bubbleplot(log(gdp), log(co2.emissions), 1960:2007, size = population,
               col = life.expectancy, interpolate = TRUE)

## End(Not run)
```

---

<code>rp.button</code>	<i>Button control for rpanel</i>
------------------------	----------------------------------

---

### 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)), repeatdelay=0,
          repeatinterval=0, quitbutton=FALSE, pos=NULL, foreground=NULL,
          background=NULL, font=NULL, parentname=deparse(substitute(panel)),
          name=paste("button", .nc(), sep=""), ...)
```

**Arguments**

panel	the panel in which the button should appear.
action	the function executed when the button is pressed.
title	the text displayed on the button.
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 an <code>rp.block</code> call. Before the window is destroyed the action function will be called.
pos	the layout instructions. Please see the <a href="#">rp.pos</a> example and help for full details.
foreground	this sets the colour of text e.g. "navy"
background	this sets the background colour of text e.g. "white"
font	this sets the text font e.g. "Arial"
parentname	this specifies the widget inside which the button should appear.
name	the name of the button.
...	...

**Details**

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

**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**

The arguments `id` and `parent` have been discontinued in version 1.1.

**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

```
## Not run:
# 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")

## End(Not run)
```

---

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(hscale = 1)
```

## Arguments

**hscale** a scaling parameter for the size of the plot which will be passed to all relevant menu items.

## Value

Nothing.

## References

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

## Examples

```
## Not run:
rp.cartoons()

## End(Not run)
```

rp.checkbox

*A checkbox control for rpanel***Description**

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

**Usage**

```
rp.checkbox(panel, variable, action=I, labels=NULL, names=NULL, title=NULL,
  initval=rep(FALSE, length(labels)), pos=NULL, doaction=FALSE, foreground=NULL,
  background=NULL, font=NULL, parentname=deparse(substitute(panel)),
  name=paste("checkbox", .nc(), sep=""), ...)
```

**Arguments**

panel	the panel in which the checkbox(es) should appear.
variable	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. The length of labels determines the number of checkboxes created. This default value for labels is the name of variable, and therefore a single checkbox.
names	the names attached to the elements of variable. These provide a helpful means of referring to particular items in multiple checkboxes when defining the action function. If names were not specified in the call to rp.control then names is set to labels.
title	the title of the checkbox group. This defaults to the name of the variable variable.
initval	the initial value for variable (optional). The initial value can also be specified in the call to rp.control.
pos	the layout instructions. Please see the <a href="#">rp.pos</a> 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 FALSE, so that the rp.do function should be called after all widgets have been created, to initialise the state of the panel display.
foreground	this sets the colour of text e.g. "navy"
background	this sets the background colour of text e.g. "white"
font	this sets the text font e.g. "Arial"
parentname	this specifies the widget inside which the checkbox(es) should appear.
name	the name of the checkbox.
...	...

**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.

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

**Examples**

```
## Not run:
  plot.hist <- function(panel) {
    with(panel, {
      xlim <- range(c(x, mean(x) + c(-3, 3) * sd(x)))
      if (panel$cbox[3])
        clr <- "lightblue" else clr <- NULL
      hist(x, freq = FALSE, col = clr, xlim = xlim)
      if (panel$cbox[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$cbox[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)

## End(Not run)
```

---

`rp.ci`*Simulations of normal-based confidence intervals*

---

### 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 and the size of the simulated sample.

### Usage

```
rp.ci(mu = 0, sigma = 1, sample.sizes = c(30, 50, 100, 200, 500), confidence = 0.95,  
      panel = TRUE, panel.plot = TRUE, hscale = NA, vscale = hscale)
```

### Arguments

<code>mu</code> , <code>sigma</code>	the population mean and standard deviation.
<code>sample.sizes</code>	the available sample sizes (30, 50, 100, 200, 500) for simulated data.
<code>confidence</code>	the available confidence levels (0.90, 0.95, 0.99).
<code>panel</code>	a logical parameter which determines whether interactive controls are provided or a simple static plot is produced.
<code>panel.plot</code>	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.
<code>hscale</code> , <code>vscale</code>	scaling parameters for the size of the plot when <code>panel.plot</code> 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**

```
## Not run:  
  rp.ci()  
  
## End(Not run)
```

---

rp.clearlines	<i>Remove lines from an rpanel image</i>
---------------	--

---

**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, imagename)
```

**Arguments**

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

**Note**

In version 1.1 "id" has been renamed "name" to be consistent with the rest of rpanel.

**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

```
## Not run:
panel <- rp.control()
image.file <- file.path(system.file(package = "rpanel"), "images", "gulllmks.gif")
panel <- rp.image(panel, image.file, imagename="gulls.image")
rp.line(panel, imagename=gulls.image, 10, 10, 100, 100, color = "green")
rp.line(panel, imagename=gulls.image, 100, 100, 100, 10, color = "blue")
rp.clearlines(panel, imagename=gulls.image)

## End(Not run)
```

---

rp.colour.key	<i>Creates a colour key.</i>
---------------	------------------------------

---

## Description

A colour key is created using the specified colours (`cols`) and an axis defined by the specified breaks (`brks`). This is usually an additional component of a panel which allows the colours on the main plot to be interpreted. The function is used in that way in the function `rp.plot4d`.

## Usage

```
rp.colour.key(cols, brks, par.mar = c(5, 0, 4, 3) + 0.1, natural = TRUE, margin = FALSE)
```

## Arguments

<code>cols</code>	a vector of colours.
<code>brks</code>	a vector of values which defines the positions on the axis between which each colour is placed.
<code>par.mar</code>	a vector of four values which are passed to the <code>mar</code> argument of the <code>par</code> function to control the marginal space around the key.
<code>natural</code>	a logical value which, when <code>TRUE</code> , causes the usual form of axis to be constructed from the values in <code>brks</code> . When <code>natural</code> is <code>FALSE</code> , the values in <code>brks</code> are associated with a regularly spaced set of locations along the axis.
<code>margin</code>	a logical value which determines whether a marginal plotting area is placed on the left of the key. This can be useful in allowing relevant information to be plotted alongside the key, such as the confidence intervals in <code>rp.surface</code> . Specifically, if <code>margin</code> is <code>FALSE</code> , the horizontal axis has range <code>c(0, 1)</code> while if <code>margin</code> is <code>TRUE</code> the the range is <code>c(-1, 1)</code> . In both cases the key is plotted over the horizontal range <code>c(0, 1)</code> .

## References

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

**Examples**

```
## Not run:
key.plot <- function(panel) {
  rp.colour.key(topo.colors(12), 0:12)
  panel
}
panel <- rp.control()
rp.tkrplot(panel, key, key.plot, hscale = 0.15)

## End(Not run)
```

rp.combo

*A 'combo' for a panel***Description**

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

**Usage**

```
rp.combo(panel, variable, prompt=NULL, vals, initval=vals[1], pos=NULL, action=I,
  foreground=NULL, background=NULL, font=NULL, editable=FALSE,
  parentname=deparse(substitute(panel)), name=paste("combo", .nc(), sep=""), ...)
```

**Arguments**

panel	the panel in which the combobox should appear.
variable	the name of the variable whose value is set by the combobox.
prompt	the label for the combobox.
vals	the values of variable used by the combo.
initval	the initial value of variable (optional). The initial value can also be specified in the call to rp.control.
pos	the layout instructions. Please see the <a href="#">rp.pos</a> example and help for full details.
action	the function which is called when an item is chosen.
foreground	colour of the text
background	colour of the text background
font	font to be used
editable	whether the combobox can be edited or not.
parentname	this specifies the widget inside which the combobox should appear.
name	name assigned to the combobox, used for disposing of the widget
...	...

## Details

The function `action` should take one argument, which should be the panel to which the combobox 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

Parameters `parent` and `...` have been discontinued in version 1.1. Note that the argument previously named `var` has been renamed `variable` to avoid reserved word issues.

## 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

```
## Not run:
  callback <- function(panel) {
    print(panel$option)
    panel
  }
  panel <- rp.control()
  rp.combo(panel, option, "Pick an option:",
           c("Option1", "Option2", "Other options"), action=callback)

## End(Not run)
```

---

rp.control                      *Create or dispose of an rpanel*

---

### Description

The function `rp.control` creates a panel window into which `rpanel` widgets can be placed. It can also set up variables within the `rpanel` object. The function `rp.control.dispose` disposes of an `rpanel`.

### Usage

```
rp.control(title = "", size = c(100, 100), panelname, background, ...)
rp.control.dispose(panel)
```

### Arguments

<code>title</code>	the title of the panel displayed in the banner.
<code>size</code>	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.
<code>panelname</code>	the name of the panel. It is important to set this parameter for <code>rp.block</code> and <code>rp.control.dispose</code> .
<code>background</code>	the background colour of the control e.g. "white". (New parameter with version 2.0.)
<code>...</code>	additional arguments which are treated as variable initialisations and are stored within the returned <code>rpanel</code> object. For example inserting <code>x=3</code> creates a variable <code>x</code> in the <code>rpanel</code> 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> .
<code>panel</code>	the panel to be disposed of. This represents the object and its parameters

### Details

Objects passed into `rp.control` are then available to be used by action functions.

### Value

The list object which defines the panel.

### Note

Previous arguments `realname` and `aschar` have been discontinued in version 1.1.

### References

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

**See Also**

[rp.button](#), [rp.checkbox](#), [rp.combo](#), [rp.doublebutton](#), [rp.grid](#), [rp.image](#), [rp.listbox](#), [rp.menu](#), [rp.radiogroup](#), [rp.slider](#), [rp.text](#), [rp.textentry](#), [rp.tkrplot](#), [rp.widget.dispose](#)

**Examples**

```
## Not run:
hist.or.boxp <- function(panel) {
  if (panel$plot.type == "histogram")
    hist(panel$x)
  else
    boxplot(panel$x)
  panel
}
panel <- rp.control(x=rnorm(50), panelname="panel")
rp.radiogroup(panel, plot.type, c("histogram", "boxplot"),
              title="Plot type", action = hist.or.boxp)

# 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.control.dispose(panel, "panel")

## End(Not run)
```

---

rp.control.put	<i>Updates the panel environment with the current value of the panel list object.</i>
----------------	---

---

**Description**

Sometimes an action function makes changes to the panel list object. When the action function is completed, the panel environment is updated. However, if there are other calls to action functions within the original action function, then the panel environment needs to be updated before these calls. This function achieves that.

**Usage**

```
rp.control.put(panelname, panel)
```

**Arguments**

panelname	the panelname of the relevant panel. This is usually identified as panel\$panelname.
panel	the relevant panel.

**References**

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

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

```
## Not run:
action1 <- function(panel) {
  panel$x <- rnorm(1)
  rp.control.put(panel$panelname, panel)
  rp.do(panel, action2)
  panel
}
action2 <- function(panel) {
  print(panel$x)
  panel
}
panel <- rp.control(x = 0)
rp.button(panel, action1, "new x")

## End(Not run)
```

---

`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, imagename, id)
```

**Arguments**

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

**Value**

If the argument `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.

**Note**

In version 1.1, the former argument `image` has been renamed `name` to be consistent with the rest of **rpanel**.

## References

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

## Examples

```
## Not run:
panel <- rp.control()
image.file <- file.path(system.file(package = "rpanel"), "images", "gulllmks.gif")
panel <- rp.image(panel, image.file, imagename="gulls.image")
rp.line(panel, imagename=gulls.image, 10, 10, 100, 100, color = "green", id="first")
rp.line(panel, imagename=gulls.image, 100, 100, 100, 10, color = "blue", id="second")
rp.deleteline(panel, imagename=gulls.image, id="first")

## End(Not run)
```

---

rp.do

*Runs a user-written action function*

---

## Description

Runs a user-written action function, passing a panel to it as a parameter. This can be used to put the rpanel into its initial state. For example, 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, x = NA, y = NA)
```

## Arguments

panel	the panel to be passed as a parameter to the function.
action	the function to be executed.
x,y	additional arguments for mouse position on the plot, so that the action function can be called with these additional arguments if they are present.

## Value

If the argument 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**

```
## Not run:
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", initval="histogram")
rp.do(panel, data.plotfn)

## End(Not run)
```

---

rp.doublebutton	<i>Double-button widget for rpanel</i>
-----------------	--

---

**Description**

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

**Usage**

```
rp.doublebutton(panel, variable, step, title=deparse(substitute(variable)),
  action=I, initval=NULL, range=c(NA, NA), log=FALSE,
  showvalue=FALSE, showvaluewidth=4, repeatinterval=100,
  repeatdelay=100, pos=NULL, foreground=NULL,
  background=NULL, font=NULL, parentname=deparse(substitute(panel)),
  name=paste("doublebutton", .nc(), sep=""), ...)
```

**Arguments**

panel	the panel in which the doublebutton should appear.
variable	the name of the variable within the panel that the doublebutton should control.
step	the value by which the variable "variable" is incremented or decremented on pressing a button. When log is TRUE this is a factor instead.
title	the label for the doublebutton. This defaults to the name of var.

action	the function which is called when a button is pressed.
initval	the initial value for var (optional). The initial value can also be specified in the call to rp.control.
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 "variable" 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 <a href="#">rp.pos</a> example and help for full details.
foreground	colour of the text
background	colour of the text background
font	font to be used
parentname	this specifies the widget inside which the doublebutton widget.
name	name assigned to the doublebutton; used for disposal etc
...	...

### 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 argument 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.

### Note

The former arguments parent and ... have been discontinued in version 1.1. Note also that the argument var has been renamed variable to avoid reserved word issues.

## 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

```
## Not run:
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)

## End(Not run)
```

---

 rp.firth

*Geostatistical sampling and analysis simulation tool*


---

## Description

This function gives access to a sampling scenario which is based on the mapping of 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 = NA,
  parameters = NA)
```

**Arguments**

hscale	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.
col.palette	the colour palette used to display the predicted and true spatial surfaces.
col.se	the colour used to draw the standard error contours on the predicted surface.
file	the name of a file to which the sampled data will be written.
parameters	a list which can be used to change the parameters which control the simulated measurement data.

**Details**

The use of the function is discussed in detail in the paper by Bowman et al. (2008) 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.

**Value**

Nothing is returned.

**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. *Journal of Statistical Software*, 36, 13, 1–17.

**See Also**

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

**Examples**

```
## Not run:
  rp.firth()

## End(Not run)
```

**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 = 0.5, max.pSill = 1, max.Nugget = 1, max.Kappa = 10,
          max.aniso.ratio = 5,
          min.ngrid = 10, max.ngrid = 25, hscale = NA, vscale = hscale,
          col.palette = terrain.colors(40))
```

**Arguments**

max.Range, max.pSill, max.Nugget	the maximum values of the range, sill and nugget parameters. These define the end-points of the corresponding slider scales.
max.Kappa	The maximum value of the kappa parameter in the Matern family of spatial covariance functions.
max.aniso.ratio	The maximum value of the anisotropy ratio parameter, which controls the degree of anisotropy in the simulated field.
min.ngrid, max.ngrid	the minimum and maximum values of the grid size for sampling points.
hscale, vscale	horizontal and vertical scaling factors for the size of the plots. It can be useful to adjust these for different screen resolutions or for projection in a lecture setting. The default values are 1.2 on Unix platforms and 1.4 on Windows platforms.
col.palette	the colour palette used to display the random fields.

**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 by Bowman et al. (2008) referenced below.

The **geoR** and **RandomFields** packages are used to generate the data.

Note that the Matern covariance function is parameterised in the form described by Handcock & Wallis (1994) which separates the effects of the shape and range parameters.

**Value**

Nothing is returned.

**References**

Adler, D. (2005). rgl: 3D Visualization Device System (OpenGL). <http://CRAN.R-project.org>.

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. *Journal of Statistical Software*, 36, 13, 1–17.

Diggle, P.J. and Ribiero, P.J. (2008). *Model-based Geostatistics*. Springer, New York.

Handcock, M.S. and Wallis, J.R. (1994). An approach to statistical spatial-temporal modeling of meteorological fields. *Journal of the American Statistical Association*, 89, 368-378.

**See Also**

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

**Examples**

```
## Not run:
  rp.geosim()

## End(Not run)
```

---

rp.grid

*Define a subsidiary grid within an rpanel*

---

**Description**

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

**Usage**

```
rp.grid(panel, name=paste("grid", .nc(), sep=""), pos=NULL, background=NULL,
  parentname=deparse(substitute(panel)), ...)
```

**Arguments**

panel	the panel to which the grid should be attached.
name	a string defining the name of the grid. For use with <a href="#">rp.widget.dispose</a>
pos	See the help information on "grid" mode in <a href="#">rp.pos</a> , for more information.
background	a character variable defining a background colour. (This is not the same as colours in R, but simple colours are available.)
parentname	this specifies the widget inside which the grid should appear.
...	...

## 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.

## Note

The former argument `parent` has been discontinued in version 1.1, while the argument `bg` has been renamed `background` for consistency with the other functions.

## References

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

## Examples

```
## Not run:
panel <- rp.control()
rp.grid(panel, pos=list(row=0, column=0, sticky="news"),
        background="red", name="g0")
rp.grid(panel, pos=list(row=1, column=1, sticky="news", width=100, height=100),
        background="navy", name="g1")
rp.grid(panel, pos=list(row=2, column=2, sticky="news", width=150, height=200),
        background="green", name="g2")
rp.button(panel, function(panel) { panel }, "press A",
         pos=list(row=1, column=1, sticky=""), parentname="g1")
rp.button(panel, function(panel) { panel }, "press B",
         pos=list(row=2, column=2, sticky="news"), parentname="g1")
rp.button(panel, function(panel) { panel }, "press C",
         pos=list("left",width=50, height=150), parentname="g2")
rp.grid(panel, pos=list(row=0, column=0, sticky="", width=10, height=10),
        background="yellow", parentname="g0")

## End(Not run)
```

---

 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 = "", panel.plot = TRUE)
```

**Arguments**

df.name	a string giving the filename where the dataframe containing the currently collected measurements will be stored using the save function. If this string is the default value of "" then no file will be saved.
panel.plot	whether to plot or not.

**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 immediately saved in the file df.name and is therefore available for inspection and analysis simply by loading this file. If the measurement is invalid or not useful, an appropriate message is given in a pop-up window.

Note that in versions of rpanel earlier than 1.1-1 the dataframe containing the collected data was previously forced into the global environment for immediate access. This has been replaced by the use of a user-nominated file.

**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**

```
## Not run:
  rp.gulls()

## End(Not run)
```

---

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, imagename, action=NA, mousedrag=NA, mouseup=NA, pos=NULL,
  parentname=deparse(substitute(panel)), ...)
```

**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.
imagename	name assigned to the image, used for disposing of the widget
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.
pos	the layout instructions. Please see the <a href="#">rp.pos</a> example and help for full details.
parentname	this specifies the widget inside which the image should appear.
...	...

**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 argument 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**

The former arguments parent and ... have been discontinued in version 1.1. Note also that the argument id has been renamed name to be consistent with the rest of rpanel.

**References**

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

**Examples**

```
## Not run:
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, gulls.image, action = gulls.click)

## End(Not run)
```

---

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

loglik.fn	This should be either the name of a function, with arguments theta and data, 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.
data	an object which contains the data. This will be referred to in likelihood contributions.
theta.low	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**

rp.panel: Statistical cartoons in R. ([http://www.mathstore.ac.uk/headocs/doc.php?doc=7403\\_bowman\\_a\\_statsr.pdf](http://www.mathstore.ac.uk/headocs/doc.php?doc=7403_bowman_a_statsr.pdf))

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

**Examples**

```
## Not run:
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))

## End(Not run)
```

---

rp.line	<i>Draws a line on an rpanel image</i>
---------	--

---

**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, imagename, 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.
imagename	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.
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 argument `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.

## Note

In version 1.0, the former argument `image` has been renamed `name` to be consistent with the rest of `rpanel`.

## 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

```
## Not run:
click.capture <- function(panel,x,y) {
  if (is.null(panel$x)) {
    panel$x <- as.numeric(x)
    panel$y <- as.numeric(y)
  }
  else {
    rp.line(panel, imagename=gulls.image, panel$x, panel$y,
            as.numeric(x), as.numeric(y), width=3, id = "current")
    panel$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, imagename="gulls.image", action = click.capture)

## End(Not run)
```

---

rp.listbox	<i>Listbox for a panel</i>
------------	----------------------------

---

### 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, variable, vals, labels = vals,
           rows=length(labels), initval=vals[1], pos=NULL,
           title=deparse(substitute(variable)), action=I, foreground=NULL,
           background=NULL, font=NULL, parentname=deparse(substitute(panel)),
           sleep = 0.01, name=paste("listbox", .nc(), sep=""), ...)
```

### Arguments

panel	the panel in which the listbox should appear.
variable	the name of the variable whose value is set by the listbox.
vals	the values of var used by the listbox. NOTE: Not currently in use, intended to be.
labels	the labels for values of var 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 <var> (optional). The initial value can also be specified in the call to rp.control.
pos	the layout instructions. Please see the <a href="#">rp.pos</a> example and help for full details.
title	the label for the listbox.
action	the function which is called when an item is chosen.
foreground	colour of the text
background	colour of the text background
font	font to be used
parentname	this specifies the widget inside which the listbox should appear.
sleep	a length of time in seconds, passed to Sys.sleep, which can be used to overcome a technical problem in some computer systems. If the listbox appears blank, then setting this parameter to a slightly value may fix the problem.
name	name assigned to the listbox, used for disposing of the widget
...	...

## 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 argument `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

The former arguments `parent` and `...` have been discontinued in version 1.1. Note also that the argument `var` has been renamed `variable` to avoid reserved word issues.

## 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

```
## Not run:
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")

## End(Not run)
```

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, panel = TRUE,
            hscale = NA, vscale = hscale, alpha = 0, beta = 0,
            display = c("jitter" = FALSE, "regression line" = FALSE,
                       "fitted model" = FALSE))
```

**Arguments**

<code>x</code>	a vector of covariate values.
<code>y</code>	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.
<code>xlab</code>	a character variable used for the covariate axis label.
<code>ylab</code>	a character variable used for the response axis label.
<code>panel.plot</code>	a logical variable which determines whether the plot is placed inside the control panel.
<code>panel</code>	a logical variable which determines whether an interactive panel is created.
<code>hscale, 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>alpha</code>	the initial value of the intercept parameter.
<code>beta</code>	the initial value of the slope parameter.
<code>display</code>	the initial settings of the checkboxes which control whether the data are ‘jittered’ for visual effect and whether the movable and fitted regression lines are displayed.

**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**

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

**See Also**

[rp.regression](#)

**Examples**

```
## Not run:
  rp.logistic(river$Temperature, river$Low)

## End(Not run)
```

---

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, variable, labels, initval=NULL, action=I,
        foreground=NULL, background=NULL, font=NULL,
        name=paste("menu", .nc(), sep=""))
```

**Arguments**

panel	the panel to which the menu should be attached should appear.
variable	the name of the variable whose value is set by the menu. (Renamed in 2.0 to <code>variable</code> from <code>var</code> as <code>var</code> is a reserved word.)
labels	the labels for the menu options. These values are returned through <code>var</code> . 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 <code>variable</code> (optional). The initial value can also be specified in the call to <a href="#">rp.control</a> .
action	the function which is called when a menu item is chosen.
foreground	this sets the colour of text e.g. "navy"

background	this sets the background colour of text e.g. "white"
font	this sets the text font e.g. "Arial"
name	the name of the widget - this is used by <a href="#">rp.widget.dispose</a>

### 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 as `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 as `list(list("File", "Quit"), list("Edit", "Copy", "Cut", "Paste"))`.

### 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.

### Note

The former argument `parent` has been discontinued in version 1.1.

### References

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

### See Also

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

### Examples

```
## Not run:
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)

## End(Not run)
```

---

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

```
## Not run:  
  rp.messagebox("Click OK to continue.", title = "Test message")  
  
## End(Not run)
```

**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 = NA,
           parameters = NA)
```

**Arguments**

hscale	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.
col.palette	the colour palette used to display the predicted and true spatial surfaces.
col.se	the colour used to draw the standard error contours on the predicted surface.
file	the name of a file to which the sampled data will be written.
parameters	a list which can be used to change the parameters which control the simulated measurement data.

**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.

**Value**

Nothing is returned.

**References**

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. *Journal of Statistical Software*, 36, 13, 1–17.

**See Also**

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

**Examples**

```
## Not run:
rp.mururoa()

## End(Not run)
```

---

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 = hscale)
```

**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**

```
## Not run:
y <- rnorm(50, mean = 10, sd = 0.5)
rp.normal(y)

## End(Not run)
```

---

rp.notebook

*Define a notebook within an rpanel*


---

**Description**

A tabbed notebook, the location of which is defined by pos, is created within an rpanel. Further widgets, grids or even notebooks can then be placed within the notebook.

**Usage**

```
rp.notebook(panel, tabs, tabnames=tabs, width = 600, height = 400, pos = NULL,
  foreground = NULL, background = "lightgray", font = NULL,
  parentname = deparse(substitute(panel)),
  name = paste("notebook", .nc(), sep = ""), ...)
rp.notebook.raise(panel, parentname, label)
```

**Arguments**

panel	the panel in which the notebook should appear.
tabs	this is a vector of the names to appear on the tabs
tabnames	this is a vector of the labels to be used internally - used by rp.notebook.raise
width	the width, in pixels, of the notebook
height	the height, in pixels, of the notebook
pos	the position of the notebook. see <a href="#">rp.pos</a>

foreground	this sets the colour of text e.g. "navy"
background	this sets the background colour of text e.g. "white"
font	this sets the text font e.g. "Arial"
parentname	this specifies the widget inside which the notebook should appear.
name	the name of the widget - this is used by <code>rp.widget.dispose</code>
label	the name of the tab which is to be raised
...	...

### Details

The role of this function is to specify a notebook. Nesting of notebooks is permitted. `rp.notebook.raise` is used to bring the contents of a particular tab to the foreground.

### References

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

### Examples

```
## Not run:
panel <- rp.control(title="Notebook example with two notebooks")
rp.notebook(panel, c("File", "Edit"), width=600, height=400,
             pos=list(row=0, column=0), background="lightgray",
             font="Arial", name="n1")
rp.notebook.raise(panel, "n1", "Edit")
rp.button(panel, function(panel){
  rp.messagebox("Button pressed!"); panel },
          "Test this", parentname="Edit")
rp.messagebox("A second tabbed notebook can be added to the same window.")
rp.notebook(panel, c("A tab 1", "A tab 2"), width=200, height=200,
             pos=list(row=1, column=1), background="Navy", foreground="White")
rp.messagebox("A tabbed notebook can be placed inside a tabbed notebook.")
rp.notebook(panel, c("Tab within tab", "Another tab"),
             width=200, height=100, parentname="File", name="n3")
rp.notebook.raise(panel, "n1", "File")
rp.notebook.raise(panel, "n3", "Another tab")

## End(Not run)
```

---

rp.panel

*Returns a panel*

---

### Description

Returns a named (by passing the name as a string parameter) panel.

**Usage**

```
rp.panel(panelname)
```

**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 it is 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**

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

**See Also**

[rp.control](#)

**Examples**

```
## Not run:
# create a panel - will be created in .rpenv as "newpanel"
rp.control(panelname = "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")

## End(Not run)
```

---

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**

x,y,z	vectors of observed values.
xlab	a character variable used for the first axis label.
ylab	a character variable used for the second axis label.
zlab	a character variable used for the third axis label.
axes	a logical variable determining whether the axes are shown.
new.window	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).
type	a character variable controlling the type of plotting. If the value is set to "n", the points are not plotted.
size	the size of the plotted points.
col	the colour of the plotted points.
xlim	the plotting range for the first variable.
ylim	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**

```
## Not run:
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)

## End(Not run)
```

rp.plot4d

*Animated scatterplot***Description**

This function plots two covariates coloured by a response variable and animates this by a third covariate. In particular, it is useful for plotting spatiotemporal data.

**Usage**

```
rp.plot4d(x, z, y, model, col.palette, col.breaks, col.labels,
          hscale = 1, vscale = hscale, panel = TRUE,
          x1lab, x2lab, zlab, ylab,
          background.plot = NULL, foreground.plot = NULL,
          z.window = "normal", z.window.pars = c(min(z), sd(z)/5),
          coords = rep(NA, 2), radius = 0.05, location.plot = TRUE,
          retain.location.plot = FALSE, eqscplot = FALSE,
          location.plot.type = "histogram")

rp.spacetime(space, time, y, model, col.palette, col.breaks, col.labels,
             hscale = 1, vscale = hscale, panel = TRUE,
             x1lab, x2lab, zlab, ylab,
             background.plot = NULL, foreground.plot = NULL,
             time.window = "normal",
             time.window.pars = c(min(time), sd(time)/5),
             coords = rep(NA, 2), radius = 0.05, location.plot = TRUE,
             retain.location.plot = FALSE, eqscplot = TRUE,
             location.plot.type = "histogram")
```

**Arguments**

x, space	a two column matrix of covariates, in particular spatial locations.
z, time	a vector of values, such as times, over which the scatterplot will be animated.
y	a vector of response values which will be used to colour the plotted points.

- `model` a list with components `x` (a two-column matrix), `z` (a vector) and `y` (a three-dimensional array) which defines the fitted values (`y`) over a regular grid of `x` and `z` values.
- `col.palette`, `col.breaks`, `col.labels` the colour palette used to colour the points, the break points on the scale which define the range associated with the each colour and the labels associated with the break points. If `col.palette` is missing, `topo.colors(20)` will be used, or `topo.colors` with the number of colours set by the number of levels when `y` is a factor. If `col.breaks` is missing then a regular grid over the range of the observed data is used. If `col.labels` is specified then the colour key has a grid of equally spaced colour blocks labelled by `col.labels`; otherwise the scale is linear. Setting `col.breaks` and `col.labels` differently can be useful if the data `y` are on a transformed scale but labels on the original scale are desired.
- `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.
- `panel` a logical value determining whether an interactive control panel is provided.
- `x1lab`, `x2lab`, `zlab`, `ylab` the axis labels of the variables
- `background.plot`, `foreground.plot` function to add further graphical material, such as a map, onto the background or foreground of the plot.
- `z.window`, `time.window` a character string which determines whether the window in `z` is initially "normal" or "uniform". This can be changed in the interactive panel.
- `z.window.pars`, `time.window.pars` a vector of length two which sets initial values for the location and width of the `z.window`. These values can be changed in the interactive panel.
- `coords` a vector of length two which defines the location of the window in the `x` space when the function is not used interactively.
- `radius` the radius of the window in the `x` space when the function is not used interactively.
- `location.plot` a logical value which determines whether the mouse can be used to interact with the `x` plot to create a plot of `y` against `z` for a nominated neighbourhood.
- `retain.location.plot` a logical value which determines the initial state of the checkbox determining whether a plot of `y` against `z` for a nominated neighbourhood remains in place after the mouse has been released.
- `eqsplot` a logical value which determines whether the `x` plot is constructed by using the `eqsplot` function in the **MASS** package, so that the same distances on each axis represent the same changes in the corresponding axis variables.
- `location.plot.type` a character variable controlling whether a histogram or a density estimate (using the **lattice** package) is produced when `y` is a factor or absent and a location plot is requested by clicking the mouse on the plot of `x`.

## Details

The colour black should be avoided when using a normal window shape for *z*. This is because *hsv* shading is used to indicate increasing distance from the current *z* location and black has an *hsv* representation with *s* component 0, which cannot therefore be reduced further.

## Value

Nothing is returned.

## References

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

## Examples

```
## Not run:
# The quakes data

with(quakes, {
  rp.plot4d(cbind(long, lat), depth)
  rp.plot4d(cbind(long, lat), depth, mag)
})

# S02 over Europe

with(S02, {
  location <- cbind(longitude, latitude)

  if (require(mgcv) & require(maps)) {
    location1 <- location[,1]
    location2 <- location[,2]
    model <- gam(logS02 ~ s(location1, location2, year))
    loc1 <- seq(min(location1), max(location1), length = 30)
    loc2 <- seq(min(location2), max(location2), length = 30)
    yr <- seq(min(year), max(year), length = 30)
    newdata <- expand.grid(loc1, loc2, yr)
    names(newdata) <- c("location1", "location2", "year")
    model <- predict(model, newdata)
    model <- list(x = cbind(loc1, loc2), z = yr,
                 y = array(model, dim = rep(30, 3)))
    mapxy <- map('world', plot = FALSE,
                xlim = range(longitude), ylim = range(latitude))
    rp.plot4d(location, year, logS02, model,
              col.palette = rev(heat.colors(20)),
              foreground.plot = function() map(mapxy, add = TRUE))
  }
  else
    rp.plot4d(location, year, logS02, col.palette = rev(heat.colors(20)))
})

# Dissolved Oxygen in the River Clyde
```

```

with(Clyde, {

  rp.plot4d(cbind(Doy, DO), Station, location.plot = FALSE)
  rp.plot4d(cbind(Station, DO), Doy, location.plot = FALSE)

  # Highlight the data before and after a sewage treatment plant update in 1985
  ind    <- Year >= 80 & Year <= 89 & !(Year == 85)
  year   <- Year[ind] + Doy[ind] / 365
  station <- Station[ind]
  doy    <- Doy[ind]
  do     <- DO[ind]
  group  <- factor(c("before 1985", "after 1985")[1 +
                    as.numeric(year < 85)])
  rp.plot4d(cbind(doy, do), station, group,
            col.palette = c("red", "green"), location.plot = FALSE)
})

## End(Not run)

```

---

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 set to the left, right, top or bottom edge of the panel. 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 (`x`, `y`, `width`, `height`) where all dimensions are in pixels. `x` and `y` define the co-ordinates in from the left hand side and down from the top respectively. When using this mode of laying out objects, it usually helps to define 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.
  - `background` 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.

---

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 for a two-sample t-test is displayed in the graphics window.

## Usage

```
rp.power(panel = TRUE, panel.plot = TRUE, populations.showing = FALSE,  
         ngrid = seq(10, 300), mu1 = 0, mu2 = 1,  
         sigma = 1, n = 20, xgrid = seq(- 4, 5, length = 100),  
         popdens.lim = 0.7, hscale = 1, vscale = hscale)
```

## Arguments

panel	a logical value determining whether an interactive panel is created.
panel.plot	a logical value determining whether the plot is placed inside the panel.
populations.showing	a logical value determining whether the populations are initially showing.
ngrid	a vector which determines the grid a sample sizes used.
mu1, mu2	the initial values of the means of the two populations.
sigma	the initial value of the common standard deviation of the two populations.
n	the initial value of the sample size.
xgrid	the grid of values over which the populations are plotted.
popdens.lim	the upper limit on the population density scale.
hscale, vscale	scaling parameters for the size of the plot.

## 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

Nothing is returned.

## References

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

## Examples

```
## Not run:  
  rp.power()  
  
## End(Not run)
```

---

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, variable, vals, labels=NULL, initval=vals[1], pos=NULL,
  title=deparse(substitute(variable)),
  action=I, foreground=NULL, background=NULL, font=NULL,
  parentname=deparse(substitute(panel)), name=paste("radiogroup", .nc(), sep=""), ...)
```

### Arguments

panel	the panel in which the radiobuttons should appear.
variable	the name of the variable whose values are set by the buttons.
vals	the values attached to the labels for return from the action function. NOTE: for implmentation.
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 <code>rp.control</code> .
pos	the layout instructions. Please see the <a href="#">rp.pos</a> 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.
foreground	colour of the text
background	colour of the text background
font	font to be used
parentname	this specifies the widget inside which the radiogroup should appear.
name	name assigned to the listbox, used for disposing of the widget
...	...

### 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 argument `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

```
## Not run:
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")

## End(Not run)
```

---

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,
             size = 3, col = "red")
rp.regression2(y, x1, x2, ylab = NA, x1lab = NA, x2lab = NA,
              panel = TRUE, model = "None", residuals.showing = FALSE,
              size = 3, col = "red")
```

**Arguments**

x	a vector or two column matrix of covariate values.
y	a vector of response values.
x1, x2	vectors of covariate values.
ylab	a character variable used for the response axis label.
x1lab	a character variable used for the first covariate axis label.
x2lab	a character variable used for the second covariate axis label.
xlab	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.
panel	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.
panel.plot	a logical variable which determines whether the plot is placed inside the control panel. This is relevant only to the case of one covariate.
hscale, vscale	scaling parameters for the size of the plot when there is one covariate and panel.plot is set to TRUE. The default values are 1 on Unix platforms and 1.4 on Windows platforms.
model	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.
line.showing	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.
residuals.showing	a logical value determining whether the residuals are shown on the plot when the function starts.
size	the size of the plotted points. This is relevant only to the case of two covariates.
col	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 tcltk package. Journal of Statistical Software, 17, issue 9.

## See Also

[rp.plot3d](#)

## Examples

```
## Not run:
with(CofE, {
  rp.regression(Employ, Giving)
  rp.regression(cbind(Employ, Attend), Giving)
})

## End(Not run)
```

---

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 tkplot 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**

Nothing is returned.

**References**

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

**Examples**

```
## Not run:
LH  <- luthor[,2:16]
gp  <- factor(luthor[,1])
times <- c(1:5, (5+(1:10)/2))
rp.rmpplot(log(LH), fac = gp, timept = times)

## End(Not run)
```

---

rp.screenresolution    *Screen resolution*

---

**Description**

This returns the current screen resolution as a list with two components; width and height.

**Usage**

```
rp.screenresolution()
```

**Arguments**

None

**Details**

One use of this function is to identify the size of the screen so that panels can be constructed to match this using pixel co-ordinates. However, the grid layout system is likely to be a better option in general. See [rp.grid](#) for details of this.

**References**

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

**See Also**

[rp.grid](#), [rp.pos](#)

---

rp.slider	<i>Slider for an rpanel</i>
-----------	-----------------------------

---

### Description

Add a slider (or slider group) to the panel, to graphically control a numeric variable.

### Usage

```
rp.slider(panel, variable, from, to, action=I, labels=NULL, names=NULL, title=NULL,
  log=rep(FALSE, length(from)), showvalue=FALSE, showvaluewidth=4, resolution=0,
  initval=from, pos=NULL,
  horizontal=TRUE, foreground=NULL, background=NULL, font=NULL,
  parentname=deparse(substitute(panel)), name=paste("slider", .nc(), sep=""), ...)
rp.slider.change(panel, name, value, i=1, do=TRUE)
```

### Arguments

panel	the panel in which the slider appears.
variable	the name of the variable that the slider controls.
from	the lower limit of the range of values to which the slider can be set.
to	the upper limit of the range of values to which the slider can be set.
action	the function which is called when the slider is moved.
labels	displayed labels
names	the names of the elements of <code>variable</code> , for reference by action functions.
title	the label of the slider.
log	a logical variable which controls whether the scale of the slider is logarithmic.
showvalue	a logical variable which determines whether the present value of "var" is shown. This is forced to FALSE when log is TRUE.
showvaluewidth	the number of significant digits in the shown value
resolution	the resolution of the slider scale. If > 0, all values are rounded to an even multiple of this value. The default is 0.
initval	the initial value of var (optional). The initial value can also be specified in the call to <code>rp.control</code> .
pos	the layout instructions. Please see the <a href="#">rp.pos</a> example and help for full details.
horizontal	a logical variable determining whether the slider is displayed horizontally (or vertically).
foreground	colour of the text
background	colour of the text background
font	font to be used
parentname	this specifies the widget inside which the slider should appear.

name	name assigned to the slider, used for disposing of the widget
...	...
value	new value for the slider
i	which slider to alter
do	whether to call the action event

### 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.

### 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.

### Note

New for version 2.0 is support for multiple sliders in a group. See `demo(rp.slider)`.

### 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

```
## Not run:
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)

printer <- function(panel) {
  print(panel$h)
  panel
}
panel <- rp.control(x = rnorm(50), h=c(1,2,3))
```

```

rp.slider(panel, h, c(0.5,0.5,0.5), c(5,5,5),
  log = c(TRUE,TRUE,TRUE), action = printer,
  title=c('h','h1','h2'), initval=c(1,2,3))

# An example which changes the slider position through another widget

draw <- function(panel) {
  hist(panel$x)
  abline(v=panel$v, col="red", lty=2)
  panel
}

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

redraw1 <- function(panel) {
  rp.tkrreplot(panel, plot)
  rp.slider.change(panel, "slider", panel$v)
  panel
}

x <- rnorm(25)
panel <- rp.control(v = 0, x = x)
rp.tkrplot(panel, plot, draw, pos="right")
rp.slider(panel, v, min(x), max(x), redraw, name = "slider")
rp.doublebutton(panel, v, diff(range(x))/100, action=redraw1)

## End(Not run)

```

---

rp.surface

*Interactive visualisation of a surface and its uncertainty*


---

## Description

This function plots a surface and uses interactive interrogation by the mouse, or a sequence of animations, to indicate the uncertainty in the surface as an estimate of the true surface.

## Usage

```

rp.surface(surface, covariance, x1grid, x2grid, x, y, Display = "persp",
  hscale = 1, vscale = hscale, Speed = 5, ntime = 10, ninterp = 50,
  zlim = NULL, col.palette = topo.colors(100))

```

## Arguments

surface            a matrix of estimated surface values over a regular grid.

covariance	the covariance matrix for the estimates in surface, corresponding to the estimates in vector form <code>c(surface)</code> .
x1grid, x2grid	vectors defining the regular grids over each margin of surface.
x	an optional two-column matrix of observed covariate values.
y	an optional vector of response values.
Display	a character value which determines the initial type of surface plot. Options are "image" (the default) and "persp".
hscale, vscale	scaling parameters for the size of the plot.
Speed	this determines the initial value of the speed of animations by setting the value of the sleep time (in hundredths of a second, with an offset of 2) between displayed surfaces.
ntime	the number of interpolated surfaces displayed between successive simulated surfaces, to control the smoothness of the animation.
ninterp	the number of grid values in each dimension when constructing a surface for the "image" display option. This is used because the input grid of surface may have quite low resolution which produces a rather chunky image display. A finer grid is constructed if the <code>akima</code> package is available.
zlim	a vector of length two which defines the range of plotting on the surface scale. By default, <code>zlim</code> is determined by the range of surface plus and minus three standard deviations (available from <code>covar</code> ).
col.palette	the colour palette used to paint the surface. The colours are determined simply by the height of the surface.

### Details

The interactive controls allow the surface to be plotted using `image` or `persp` displays, and with the display of uncertainty through mouse click and drag on the `image` plot or animation.

### Value

Nothing is returned.

### References

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

### Examples

```
## Not run:
if (require(sm)) {
  provide.data(trawl, options = list(describe = FALSE))
  location <- cbind(Longitude, Latitude)
  model <- sm.regression(location, Score1, ngrid = 15, display = "none")
  longitude <- model$eval.points[ , 1]
  latitude <- model$eval.points[ , 2]
  xgrid <- as.matrix(expand.grid(longitude, latitude))
}
```

```

S      <- sm.weight2(location, xgrid, model$h)
covar  <- tcrossprod(S) * model$sigma^2
rp.surface(model$estimate, covar, longitude, latitude, location, Score1)
}
## End(Not run)

```

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 = TRUE, panel.plot = TRUE, hscale = NA, vscale = hscale,
          distribution = "normal", degf1 = 5, degf2 = 30,
          observed.value = " ", observed.value.showing = !is.na(observed.value),
          probability = 0.05, tail.probability, tail.direction)

```

**Arguments**

panel	a logical parameter which determines whether interactive controls are provided or a simple static plot is produced.
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 package 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.
distribution	a character string which determines which distribution is to be plotted. Current options are "normal" (default), "t", "chi-squared" and "F".
degf1, degf2	The degrees of freedom used for the chi-squared (degf1) and F (degf1, degf2) distributions.
observed.value	a numerical value, or a character string which will be converted by as.numeric, which identifies an observed value whose location within the distribution is of interest.
observed.value.showing	a logical value which determines whether the observed value (if any) is displayed on the plot.
probability	the value of the tail probability used when tail area is shaded.
tail.probability	a character string which determines whether the tail area is drawn from the observed value ("from observed value"), using a fixed probability ("fixed probability") or not shown ("none").

`tail.direction` a character string which determines whether the lower ("lower"), upper ("upper") or two-sided ("two-sided") tail area is drawn.

### 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

Nothing is returned.

### References

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

### Examples

```
## Not run:
  rp.tables()

## End(Not run)
```

---

rp.text

*Text boxes for a panel*

---

### Description

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

### Usage

```
rp.text(panel, text, pos=NULL, action=I, foreground=NULL, background=NULL,
        font=NULL, width=NULL, parentname=deparse(substitute(panel)),
        name = paste("text", .nc(), sep=""), ...)
rp.text.change(panel, name, text)
```

### Arguments

<code>panel</code>	the panel on which the text should appear.
<code>text</code>	the text to be displayed
<code>pos</code>	the layout instructions. Please see the <a href="#">rp.pos</a> example and help for full details.
<code>action</code>	the function which is called when the text has been entered.
<code>foreground</code>	colour of the text

background	colour of the text background
font	font to be used
width	character width of the textboxes
parentname	this specifies the widget inside which the text entry widget should appear.
name	name assigned to the textentries; used for disposal etc
...	...

### Details

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

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

### 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

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

### See Also

[rp.control](#)

### Examples

```
## Not run:
panel <- rp.control(x=1)
callback <- function(panel)
{
  rp.text.change(panel, "t2", panel$x)
  panel$x = panel$x+1
  panel
}
rp.text(panel, "This is a test", name="t1")
rp.text(panel, "And so is this", font="Arial", foreground="white",
  background="navy", action=callback, name="t2")
rp.text(panel, "Here is some more text, this time across several lines.\n
  Here is some more text, this time across several lines.\n
  Here is some more text, this time across several lines.", name="t3")

## End(Not run)
```

---

rp.textentry                      *Text entry boxes for a panel*

---

### Description

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

### Usage

```
rp.textentry(panel, variable, action = I, labels = NULL, names = labels,
             title = NULL, initval = rep(NA, length(labels)), pos = NULL,
             foreground = NULL, background = NULL, font = NULL, width = 20, keydown = FALSE,
             parentname = deparse(substitute(panel)), name = paste("textentry", .nc(), sep=""), ...)
```

### 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.
variable	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	a character string of the names of the elements of variable which can be referred to within action functions.
title	title above multiple textentries
initval	the initial value(s) for var (optional). The initial value(s) can also be specified in the call to rp.control.
pos	the layout instructions. Please see the <a href="#">rp.pos</a> example and help for full details.
foreground	colour of the text
background	colour of the text background
font	font to be used
width	character width of the textboxes
keydown	if TRUE the action function will be called on every keypress - this may not be wise
parentname	this specifies the widget inside which the text entry widget should appear. In the current version, it should not normally be used.
name	name assigned to the textentries; used for disposal etc
...	...

### 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 argument `panel` is set to 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**

The former arguments `names`, `title` and `parent` have been discontinued in version 1.1. Note also that the argument `var` has been renamed `variable` to avoid reserved word issues.

**References**

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

**See Also**

[rp.control](#)

**Examples**

```
## Not run:
  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)

## End(Not run)
```

---

rp.timer *Creates a series of timed actions*

---

### Description

This creates an interval timer and allows the user to set the criteria to stop the timer.

### Usage

```
rp.timer(panel, microseconds, action, where)
```

### Arguments

panel	the panel which has some relevant variables.
microseconds	time between each call of action.
action	function to be executed on each timer tick.
where	a function which should return true or false, taking parameter panel. When false the loop will stop.

### Details

This allows the user to setup an interval timer and the function to be called at each 'tick'.

Care should be taken when writing code to anticipate interactions with the panel while activity controlled by a timer is underway, as these interactions may cause changes in the state of the panel.

### References

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

### See Also

[rp.control](#)

### Examples

```
## Not run:
stopme <- function(panel) panel$count<=20
callme <- function(panel) {
  print(panel$count)
  panel$count = panel$count+1
  panel
}
panel <- rp.control(count=1)
rp.timer(panel, 500, callme, stopme)

## End(Not run)
```

---

rp.tkrplot                      *rpanel calls for tkrplot and tkreplot*

---

### Description

These functions call Luke Tierney's tkrplot and tkreplot functions from the **tkrplot** package 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, pos=NULL, foreground=NULL, background=NULL, margins=c(0, 0, 0, 0),
           parentname=deparse(substitute(panel)), mar= par()$mar, ...)
rp.tkreplot(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.
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 <a href="#">rp.pos</a> example and help for full details.
background	the colour used for the background of the plot.
foreground	the filename of a transparent gif file. This will be overlaid on the tkrplot image after plotting takes place.
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.
parentname	this specifies the widget inside which the plot should appear. In the current version of rpanel, it should not normally be used.
mar	mar parameter for specifying the margins.
...	...

### 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 argument `panel` is set to 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**

```
## Not run:
draw <- function(panel) {
  plot(1:20, (1:20)^panel$h)
  panel
}

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

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

## End(Not run)
```

---

rp.var.get

*Retrieves an object from the rpanel environment, usually from a panel.*

---

**Description**

The management of objects within the rpanel environment is usually handled ‘behind the scenes’ but it can occasionally be useful to retrieve an object there explicitly.

**Usage**

```
rp.var.get(panelname, name)
```

**Arguments**

panelname	the panelname of the relevant panel. This is usually identified as panel\$panelname. If this argument is set to NULL then the object is not retrieved from a panel.
name	the name of the variable in character form.

**References**

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

**See Also**

[rp.var.get](#)

---

rp.var.put	<i>Places an object in the rpanel environment, usually within a panel.</i>
------------	--

---

**Description**

The management of objects within the rpanel environment is usually handled ‘behind the scenes’ but it can occasionally be useful to place an object there explicitly.

**Usage**

```
rp.var.put(panelname, name, val, labels = NULL)
```

**Arguments**

panelname	the panelname of the relevant panel. This is usually identified as panel\$panelname. If this argument is set to NULL then the object is not placed inside a panel.
name	the name of the variable.
val	the contents of the variable as a numeric or character vector.
labels	labels for var.

**References**

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

**See Also**

[rp.var.get](#)

---

rp.widget.dispose	<i>Removes a widget</i>
-------------------	-------------------------

---

### Description

This will dispose/remove a widget from a panel.

### Usage

```
rp.widget.dispose(panel, name)
```

### Arguments

panel	the panel on which the text should disappear.
name	the name assigned to the widget on creation.

### Details

This will dispose of a widget and its memory usage.

### References

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

### See Also

[rp.control](#)

### Examples

```
## Not run:  
p1 <- rp.control()  
rp.button(p1, I, "press me", name="b1")  
rp.widget.dispose(p1, b1 )  
  
## End(Not run)
```

**Description**

The data document values of SO<sub>2</sub>, on a log scale, from monitoring stations across Europe from 1990 to 2001. The data were collected through the 'European monitoring and evaluation programme' (EMEP) and they are available at [www.emep.int](http://www.emep.int). The data recorded here have been organised into a convenient form for analysis.

The data file consists of six variables: `site`: a site code for the monitoring station `longitude`: longitude of the monitoring station `latitude`: latitude of the monitoring station `year`: year of measurement `month`: month of measurement `logSO2`: SO<sub>2</sub> measurement on a log scale

**References**

Spatiotemporal smoothing and sulphur dioxide trends over Europe A. W. Bowman, M. Giannitrapani and E. M. Scott. *Applied Statistics*, 58 (2009), 737–752

**Examples**

```
## Not run:
Month    <- SO2$month + (SO2$year - 1990) * 12
Year     <- SO2$year + (SO2$month - 0.5) / 12
Location <- cbind(SO2$longitude, SO2$latitude)
back     <- I
if (require(maps)) {
  mapxy <- map('world', plot = FALSE,
              xlim = range(SO2$longitude), ylim = range(SO2$latitude))
  back <- function() map(mapxy, add = TRUE)
}
rp.plot4d(Location, Year, SO2$logSO2, col.palette = rev(heat.colors(12)),
           background.plot = back)

## End(Not run)
```

**Description**

Loading this file makes the dataframes `co2.emissions`, `gdp`, `life.expectancy` and `population` available. These contain the CO<sub>2</sub> emissions, gross domestic product, life expectancy and population data for each country of the world (rows indexed by rownames) for the years 1960-2007.

These data are provided by the World Bank through the database at <http://data.worldbank.org/data-catalog/world-d>  
The data are also used use in the Google Public Data Explorer <http://www.google.com/publicdata/directory>  
and by the Gapminder project <http://www.gapminder.org> )

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

## References

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

## Examples

```
## Not run:  
  rp.bubbleplot(log(gdp), log(co2.emissions), 1960:2007, size = population,  
               col = life.expectancy,  
               interpolate = TRUE, hscale = 1.5, vscale = 1.5)  
  
## End(Not run)
```

# Index

## \*Topic **dynamic**

- aircond, 5
- Clyde, 6
- CofE, 6
- gullweight, 7
- luthor, 7
- poisons, 8
- river, 9
- rodent, 9
- rp. ancova, 10
- rp. anova, 11
- rp. block, 13
- rp. bubbleplot, 14
- rp. button, 15
- rp. cartoons, 17
- rp. checkbox, 18
- rp. ci, 20
- rp. clearlines, 21
- rp. colour.key, 22
- rp. combo, 23
- rp. control, 25
- rp. control.put, 26
- rp. deleteline, 27
- rp. do, 28
- rp. doublebutton, 29
- rp. firth, 31
- rp. geosim, 33
- rp. grid, 34
- rp. gulls, 35
- rp. image, 36
- rp. likelihood, 38
- rp. line, 39
- rp. listbox, 41
- rp. logistic, 43
- rp. menu, 44
- rp. messagebox, 46
- rp. mururoa, 47
- rp. normal, 48
- rp. notebook, 49

- rp. panel, 50
- rp. plot3d, 51
- rp. plot4d, 53
- rp. power, 57
- rp. radiogroup, 59
- rp. regression, 60
- rp. rmpplot, 62
- rp. screenresolution, 64
- rp. slider, 65
- rp. tables, 69
- rp. text, 70
- rp. textentry, 72
- rp. timer, 74
- rp. tkplot, 75
- rp. var.get, 76
- rp. var.put, 77
- rp. widget.dispose, 78
- rpanel-package, 3
- S02, 79
- worldbank, 79

## \*Topic **iplot**

- aircond, 5
- Clyde, 6
- CofE, 6
- gullweight, 7
- luthor, 7
- poisons, 8
- river, 9
- rodent, 9
- rp. ancova, 10
- rp. anova, 11
- rp. block, 13
- rp. bubbleplot, 14
- rp. button, 15
- rp. cartoons, 17
- rp. checkbox, 18
- rp. ci, 20
- rp. clearlines, 21
- rp. colour.key, 22

- rp.combo, 23
- rp.control, 25
- rp.control.put, 26
- rp.deleteline, 27
- rp.do, 28
- rp.doublebutton, 29
- rp.firth, 31
- rp.geosim, 33
- rp.grid, 34
- rp.gulls, 35
- rp.image, 36
- rp.likelihood, 38
- rp.line, 39
- rp.listbox, 41
- rp.logistic, 43
- rp.menu, 44
- rp.messagebox, 46
- rp.mururoa, 47
- rp.normal, 48
- rp.notebook, 49
- rp.panel, 50
- rp.plot3d, 51
- rp.plot4d, 53
- rp.power, 57
- rp.radiogroup, 59
- rp.regression, 60
- rp.rmpplot, 62
- rp.screenresolution, 64
- rp.slider, 65
- rp.tables, 69
- rp.text, 70
- rp.textentry, 72
- rp.timer, 74
- rp.tkrplot, 75
- rp.var.get, 76
- rp.var.put, 77
- rp.widget.dispose, 78
- rpanel-package, 3
- S02, 79
- worldbank, 79
- \*Topic package**
- rpanel-package, 3
  
- aircond, 5
  
- Clyde, 6
- co2.emissions (worldbank), 79
- CofE, 6
  
- gdp (worldbank), 79
- gullweight, 7
  
- life.expectancy (worldbank), 79
- luthor, 7
  
- poisons, 8
- population (worldbank), 79
  
- river, 9
- rodent, 9
- rp.ancova, 3, 7, 10
- rp.anova, 3, 8, 11
- rp.block, 4, 13, 25
- rp.bubbleplot, 14
- rp.button, 4, 5, 15, 26
- rp.cartoons, 3, 9, 17
- rp.checkbox, 4, 5, 18, 24, 26, 42, 45, 60
- rp.ci, 3, 20
- rp.clearlines, 4, 21
- rp.colour.key, 4, 22
- rp.combo, 4, 23, 26
- rp.control, 4, 5, 13, 16, 19, 24, 25, 27, 31, 42, 44–46, 51, 60, 66, 71, 73, 74, 78
- rp.control.put, 26
- rp.deleteline, 4, 27
- rp.do, 4, 28
- rp.doublebutton, 4, 5, 16, 26, 29
- rp.firth, 4, 31, 34, 48
- rp.geosim, 4, 32, 33, 48
- rp.grid, 4, 16, 19, 24, 26, 30, 34, 37, 42, 59, 64, 66, 71, 72, 75
- rp.gulls, 3, 35
- rp.image, 4, 21, 26, 36, 40, 76
- rp.likelihood, 3, 5, 38
- rp.line, 4, 21, 39
- rp.listbox, 4, 26, 41
- rp.logistic, 3, 9, 43
- rp.menu, 4, 26, 44
- rp.messagebox, 4, 46
- rp.mururoa, 4, 32, 34, 47
- rp.normal, 3, 48
- rp.notebook, 49
- rp.panel, 4, 50
- rp.plot3d, 3, 51, 62
- rp.plot4d, 3, 6, 53
- rp.pos, 4, 16, 18, 23, 30, 34, 35, 37, 41, 49, 56, 59, 64, 65, 70, 72, 75
- rp.power, 3, 57

rp.radiogroup, [4](#), [5](#), [19](#), [26](#), [29](#), [31](#), [59](#), [66](#)  
rp.regression, [3](#), [6](#), [44](#), [52](#), [60](#)  
rp.regression2 (rp.regression), [60](#)  
rp.rmplot, [3](#), [7](#), [62](#)  
rp.screenresolution, [64](#)  
rp.slider, [4](#), [5](#), [26](#), [65](#)  
rp.spacetime, [3](#)  
rp.spacetime (rp.plot4d), [53](#)  
rp.surface, [4](#), [67](#)  
rp.tables, [3](#), [69](#)  
rp.text, [4](#), [26](#), [70](#)  
rp.textentry, [4](#), [5](#), [26](#), [72](#)  
rp.timer, [4](#), [74](#)  
rp.tkrplot, [4](#), [26](#), [40](#), [75](#)  
rp.tkrreplot, [4](#)  
rp.tkrreplot (rp.tkrplot), [75](#)  
rp.var.get, [4](#), [76](#), [77](#)  
rp.var.put, [4](#), [77](#)  
rp.widget.dispose, [26](#), [34](#), [45](#), [50](#), [78](#)  
rpanel (rpanel-package), [3](#)  
rpanel-package, [3](#)

S02, [79](#)

worldbank, [79](#)