Package: ContourFunctions (via r-universe)

September 12, 2024

Type Package

Title Create Contour Plots from Data or a Function

Version 0.1.2.9000

Description Provides functions for making contour plots. The contour plot can be created from grid data, a function, or a data set. If non-grid data is given, then a Gaussian process is fit to the data and used to create the contour plot.

License GPL-3

RoxygenNote 7.3.1

Imports rlang

Depends ggplot2, lhs, rmarkdown

Suggests covr, knitr, laGP, locfit, mgcv, mlegp, testthat

VignetteBuilder knitr

URL https://github.com/CollinErickson/ContourFunctions

Encoding UTF-8

Repository https://collinerickson.r-universe.dev

RemoteUrl https://github.com/collinerickson/contourfunctions

RemoteRef HEAD

RemoteSha 715d77f00b5536ca35aff1a7c665ded7e72c96d6

Contents

`cf` *Make contour plot from data or function*

Description

Simpler function for making contours with cf package. Won't give argument completion, so all must be specified

Usage

```
cf(..., gg = FALSE)
```

Arguments

`...`	Arguments to be passed to cf_func or cf_data based on data type of first argument. If D is given as argument, then it is passed to cf_highdim.
`gg`	Should ggplot2 be used instead of base graphics?

Value

Whatever is returned from other function, probably nothing. Will be a ggplot2 object if using gg=TRUE.

Examples

```
cf(function(x){x[1]^2 - x[2]})
x <- runif(20)
y <- runif(20)
z <- exp(-(x-.5)^2-5*(y-.5)^2)# + rnorm(20,0,.05)
cf(x,y,z)
cf(function(x){x[1]^2 - x[2]}, D=3)
```

cf_4dim *Plot 2D contour slices of four dimensional functions*

Description

Plots a grid of contour plots. Each contour plot is a contour over two dimensions with the remaining two dimensions set to a value. See cf_highdim for functions with more than 4 dimensions.

Usage

```
cf_4dim(
  func,
  over = c(1, 2),
  nover = 5,
  nover1 = nover,
  nover2 = nover,
  low = rep(0, 4),
  high = rep(1, 4),
  same_scale = TRUE,
  n = 20,
  batchmax = 1,
  var_names = c(expression(), lapply(1:4, function(ti) bquote(x[.(ti)]))),
  pts = NULL,
  axes = TRUE,
  key.axes,
  key.title,
  nlevels = 20,
  color.palette = cm.colors.strong,
  edge_width = 0.04,
  cex.var_names = 1.3,
  bar = TRUE,
  bar_width = 0.2,
  over_srt = c(0, 90),
  ...
)
```

Arguments

func	A four-dimensional function to plot contours of
over	Indices of the dimensions used for the outer grid
nover	Number of grid points for the outer grid dimensions
nover1	Number of grid points for the first outer grid dimension
nover2	Number of grid points for the second outer grid dimension
low	Low input value for each dimension
high	High input value for each dimension

same_scale	Should all contour plots be on the same scale? Takes longer since it has to precalculate range of outputs.
n	Number of points in grid on each dimension
batchmax	number of datapoints that can be computed at a time
var_names	Variable names to add to plot
pts	Matrix of points to show on plot
axes	axes
key.axes	key for bar plot
key.title	statements which add titles for the plot key.
nlevels	Number of levels in contour scale
color.palette	Color palette used for contour plots
edge_width	How wide should edges with variable names be? As proportion of screen section to left of bar. Either single value for both edges, or length two vector.
cex.var_names	Size of var_names printed on edges.
bar	Should a bar be added on right when all on same_scale?
bar_width	How wide should bar section of plot be?
over_srt	Degrees of rotation for the axis labels. Vector of length two.
...	Arguments passed to cf_func, and then probably through to cf_grid

Examples

```
cf_4dim(
  function(x) {x[1] + x[2]^2 + sin(2*pi*x[3])}
)

cf_4dim(function(x) x[1]*x[3] + sin(x[2]*x[4]), color.palette=heat.colors,
        nover1=3, nover2=8, cex.var_names = .5)

cf_4dim(function(x) x[1]*x[3] + sin(x[2]*x[4]), color.palette=topo.colors,
        nover1=3, nover2=8, cex.var_names = 1, over_srt = c(90,0),
        edge_width=c(.1, .2), nlevels = 5)
```

---

cf_data *Contour plot from data*

Description

Makes filled contour plot from data without sidebar by interpolating with a Gaussian process model. The model is passed to cf_func to make the contour plot.

Usage

```
cf_data(
  x,
  y = NULL,
  z = NULL,
  xlim = NULL,
  ylim = NULL,
  xylim = NULL,
  fit = "",
  gg = FALSE,
  show_points,
  family = "gaussian",
  ...
)
```

Arguments

x	either just x data, x and y data, or x, y and z data
y	either y data, z data, or null
z	either z data or null
xlim	x limits for the contour plot, will be set to data limits +- 5% if not specified
ylim	y limits for the contour plot, will be set to data limits +- 5% if not specified
xylim	x and y limits for the contour plot
fit	Method to fit a model with. Current options are laGP (default), mlegp, gam (uses mgcv), and locfit. laGP is faster but might cause trouble.
gg	If TRUE, will use ggplot2 by calling gcf_func
show_points	Whether the input data points should be shown on the plot. If missing, is TRUE when there are more than 300 points.
family	The distribution/link to be used in fitting. Only available when fit is locfit or mgcv.
...	passed to cf_func

Examples

```
x <- runif(20)
y <- runif(20)
z <- exp(-(x-.5)^2-5*(y-.5)^2)
cf_data(x,y,z)
```

cf_func *Makes filled contour plot from function*

Description

A contour plot of the given function without sidebar by default. It calls the function 'cf_grid' to make the actual plot.

Usage

```
cf_func(
  fn0,
  n = 100,
  xlim = c(0, 1),
  ylim = c(0, 1),
  xylim = NULL,
  batchmax = 1,
  out.col.name = NULL,
  out.name = NULL,
  pts = NULL,
  gg = FALSE,
  ...
)
```

Arguments

fn0	function to plot, first argument must be two-dimensional
n	number of points in each dimension
xlim	x limits for the contour plot
ylim	y limits for the contour plot
xylim	x and y limits for the contour plot, use when both are same #@param mainminmax whether the min and max values should be shown in the title of plot
batchmax	number of datapoints that can be computed at a time
out.col.name	if a column needs to be selected from the function, specify it
out.name	Selects with a $ the name from output to be used, for lists and data frames #@param pretitle Text to be preappended to end of plot title #@param posttitle Text to be appended to end of plot title #@param title Title for the plot #@param mainminmax_minmax Whether [min,max]= should be shown in title or just the numbers
pts	Points to plot on top of contour
gg	Should ggplot2 be used? Will use gcf_grid() instead of cf_grid().
...	Passed to cf_grid

References

[1] filled.contour R function, copied function but removed part for sidebar

[2] http://stackoverflow.com/questions/16774928/removing-part-of-a-graphic-in-r, answer by P Lapointe

Examples

```
cf_func(function(x){x[1]*x[2]})
cf_func(function(x)(exp(-(x[1]-.5)^2-5*(x[2]-.5)^2)))
cf_func(function(xx){exp(-sum((xx-.5)^2/.1))}, bar=TRUE)
cf_func(function(xx){exp(-sum((xx-.5)^2/.1))}, bar=TRUE, mainminmax=TRUE)
cf_func(function(x)(exp(-(x[1]-.5)^2-5*(x[2]-.5)^2)), with_lines=TRUE)
```

---

cf_grid *Create a contour plot from a grid of data*

Description

Makes filled contour plot with an optional sidebar, essentially filled.contour function. This version uses the split.screen() function to add the sidebar if bar is TRUE. By default it won't show the bar but will show the min and max values in the plot title along with their colors. Using this function will make other functions such as points() called afterwards not put points where you expect. Pass anything you want added to the plot area to afterplotfunc as a function to get it to work properly.

Usage

```
cf_grid(
  x = seq(0, 1, length.out = nrow(z)),
  y = seq(0, 1, length.out = ncol(z)),
  z,
  xlim = range(x, finite = TRUE),
  ylim = range(y, finite = TRUE),
  zlim = range(z, finite = TRUE),
  levels = pretty(zlim, nlevels),
  nlevels = 20,
  color.palette = cm.colors.strong,
  col = color.palette(length(levels) - 1),
  plot.title,
  plot.axes,
  key.title,
  key.axes,
  asp = NA,
  xaxs = "i",
  yaxs = "i",
  las = 1,
  axes = TRUE,
  frame.plot = axes,
```

```
  bar = F,
  pts = NULL,
  reset.par = TRUE,
  pretitle = "",
  posttitle = "",
  main = NULL,
  mainminmax = !bar,
  mainminmax_minmax = TRUE,
  afterplotfunc = NULL,
  cex.main = par()$cex.main,
  par.list = NULL,
  xaxis = TRUE,
  yaxis = TRUE,
  with_lines = FALSE,
  lines_only = FALSE,
  ...
)
```

Arguments

x	x values, must form grid with y. If not given, it is assumed to be from 0 to 1.
y	y values, must form grid with x. If not given, it is assumed to be from 0 to 1.
z	z values at grid locations
xlim	x limits for the plot.
ylim	y limits for the plot.
zlim	z limits for the plot.
levels	a set of levels which are used to partition the range of z. Must be strictly increasing (and finite). Areas with z values between consecutive levels are painted with the same color.
nlevels	if levels is not specified, the range of z, values is divided into approximately this many levels.
color.palette	A color palette function to be used to assign colors in the plot. Defaults to cm.colors.strong. Other options include rainbow, heat.colors, terrain.colors, topo.colors, and function(x) {gray((1:x)/x)}.
col	an explicit set of colors to be used in the plot. This argument overrides any palette function specification. There should be one less color than levels
plot.title	statements which add titles to the main plot.
plot.axes	statements which draw axes (and a box) on the main plot. This overrides the default axes.
key.title	statements which add titles for the plot key.
key.axes	statements which draw axes on the plot key. This overrides the default axis.
asp	the y/x aspect ratio, see plot.window.
xaxs	the x axis style. The default is to use internal labeling.
yaxs	the y axis style. The default is to use internal labeling.

las	the style of labeling to be used. The default is to use horizontal labeling.
axes	logical indicating if axes should be drawn, as in plot.default.
frame.plot	logical indicating if a box should be drawn, as in plot.default.
bar	Should a bar showing the output range and colors be shown on the right?
pts	Points to plot on top of contour
reset.par	Should the graphical parameters be reset before exiting? Usually should be unless you need to add something to the plot afterwards and bar is TRUE.
pretitle	Text to be preappended to end of plot title
posttitle	Text to be appended to end of plot title
main	Title for the plot
mainminmax	whether the min and max values should be shown in the title of plot
mainminmax_minmax	Whether [min,max]= should be shown in title or just the numbers
afterplotfunc	Function to call after plotting, such as adding points or lines.
cex.main	The size of the main title. 1.2 is default.
par.list	List of options to pass to par
xaxis	Should x axis be added?
yaxis	Should y axis be added?
with_lines	Should lines be added on top of contour to show contours?
lines_only	Should no fill be used, only contour lines?
...	additional graphical parameters, currently only passed to title().

References

[1] filled.contour R function, copied function but removed part for sidebar

[2] http://stackoverflow.com/questions/16774928/removing-part-of-a-graphic-in-r, answer by P Lapointe

Examples

```
x <- y <- seq(-4*pi, 4*pi, len = 27)
r <- sqrt(outer(x^2, y^2, "+"))
cf_grid(cos(r^2)*exp(-r/(2*pi)))
cf_grid(r, color.palette=heat.colors, bar=TRUE)
cf_grid(r, color.palette=function(x) {gray((1:x)/x)}, bar=TRUE)
```

cf_highdim *Plot 2D contour slices of higher dimensional functions*

Description

Plots a grid of contour plots. Each contour plot is a contour over two dimensions with the remaining dimensions set to the baseline value. Similar to plots created in Hwang et al. (2018).

Usage

```
cf_highdim(
  func,
  D,
  low = rep(0, D),
  high = rep(1, D),
  baseline = (low + high)/2,
  same_scale = TRUE,
  n = 20,
  batchmax = 1,
  var_names = c(expression(), lapply(1:D, function(ti) bquote(x[.(ti)]))),
  pts = NULL,
  average = FALSE,
  average_reps = 10000,
  axes = TRUE,
  key.axes,
  key.title,
  nlevels = 20,
  levels = pretty(zlim, nlevels),
  color.palette = cm.colors.strong,
  col = color.palette(length(levels) - 1),
  edge_width = 0.04,
  cex.var_names = 1.3,
  bar = TRUE,
  ...
)
```

Arguments

func	Function to plot contours of
D	Input dimension of function
low	Low input value for each dimension
high	High input value for each dimension
baseline	Baseline input value for each dimension
same_scale	Should all contour plots be on the same scale?
n	Number of points in grid on each dimension

batchmax	number of datapoints that can be computed at a time
var_names	Variable names to add to plot Takes longer since it has to precalculate range of outputs.
pts	Matrix of points to show on plot
average	Should the background dimensions be averaged over instead of set to baseline value? Much slower.
average_reps	Number of points to average over when using average
axes	logical indicating if axes should be drawn, as in plot.default.
key.axes	statements which draw axes on the plot key. This overrides the default axis.
key.title	statements which add titles for the plot key.
nlevels	if levels is not specified, the range of z, values is divided into approximately this many levels.
levels	a set of levels which are used to partition the range of z. Must be strictly increasing (and finite). Areas with z values between consecutive levels are painted with the same color.
color.palette	A color palette function to be used to assign colors in the plot. Defaults to cm.colors.strong. Other options include rainbow, heat.colors, terrain.colors, topo.colors, and function(x) {gray((1:x)/x)}.
col	an explicit set of colors to be used in the plot. This argument overrides any palette function specification. There should be one less color than levels.
edge_width	How wide should edges with variable names be? As proportion of full screen.
cex.var_names	Size of var_names printed on edges.
bar	Should a bar showing the output range and colors be shown on the top right?
...	Arguments passed to cf_func, and then probably through to cf_grid

References

Hwang, Yongmoon, Sang-Lyul Cha, Sehoon Kim, Seung-Seop Jin, and Hyung-Jo Jung. "The Multiple-Update-Infill Sampling Method Using Minimum Energy Design for Sequential Surrogate Modeling." Applied Sciences 8, no. 4 (2018): 481.

Examples

```
## Not run:
# Only use 4 dims of 8 for borehole function
cf_highdim(function(x) TestFunctions::borehole(c(x,.5,.5,.5,.5)), 4)
# Add points
cf_highdim(function(x) TestFunctions::borehole(c(x,.5,.5,.5,.5)), 4,
           pts=matrix(c(.1,.3,.6,.9),1,4))

# Full 8D borehole function
cf_highdim(TestFunctions::borehole, 8)

# Putting each plot on separate scale
cf_highdim(TestFunctions::borehole, 8, n=10, same_scale = FALSE)
```

```
## End(Not run)

cf_highdim(function(x) {x[1]^2 + exp(x[2])}, D=3)

friedman <- function(x) {
  10*sin(pi*x[1]*x[2]) + 20*(x[3]-.5)^2 + 10*x[4] + 5*x[5]
}
cf_highdim(friedman, 5, color.palette=topo.colors)
cf_highdim(friedman, 5,
           color.palette=function(x) {gray((1:x)/x)},
           nlevels=10)

## Not run:
# Recreate Plate 1 or Figure 1.1 from Engineering Design via Surrogate
# Modelling by Forrester, Sobester, and Keane (2008).
cf_highdim(function(x)TestFunctions::wingweight(x, scale_it=FALSE),
  D=10, low = c(150,220,6,-10,16,.5,.08,2.5,1700,.025),
  high = c(200,300,10,10,45,1,.18,6,2500,.08),
  baseline=c(174,252,7.52,0,34,.672,.12,3.8,2000,.064),
  color.palette=topo.colors,
  var_names=c('SW', 'Wtw', 'A', 'Lambda', 'q', 'lambda', 'tc', 'Nz', 'Wdg'))

## End(Not run)

# Average over background dimensions, use higher reps to reduce noise.
f1 <- function(x) {x[1] + x[2]^2 + x[3]^3}
cf_highdim(f1, 4, average=TRUE, average_reps=1e2, n=10)
f1b <- function(x) {x[,1] + x[,2]^2 + x[,3]^3}
cf_highdim(f1b, 4, average=TRUE, average_reps=1e2, n=10, batchmax=Inf)
cf_highdim(f1b, 4, average_reps=1e2, n=10, batchmax=Inf,
           color.palette = topo.colors, nlevels=3)

# This was giving bad result
csa()
split.screen(c(2,1))
screen(2)
cf_highdim(f1b, 4, n=10, batchmax=Inf)
csa()
```

`cm.colors.strong` *Strong version of cm.colors color palette*

Description

Altered version of cm.colors that uses full saturation to get stronger colors.

Usage

```
cm.colors.strong(n, alpha = 1)
```

Arguments

n	Number of color groups
alpha	Alpha level

Value

Character vector of colors

Examples

```
# Character string output
cm.colors.strong(5)

# Plot to show these
sl <- 21
sx <- seq(0,1,l=sl)
plot(sx,sin(2*pi*sx), cex=5, col=cm.colors.strong(sl), pch=19);points(sx,sin(2*pi*sx), cex=5)
plot(sx,sin(2*pi*sx), cex=5, col=cm.colors(sl),     pch=19);points(sx,sin(2*pi*sx), cex=5)
```

---

csa *Close all open screens*

Description

Closes the screens open, which happens when plotting with ‘split.screen‘ is interrupted. It often happens when there is a error while plotting. When you try to plot the next thing it gives an error. Running this function will reset the plot screen. It just does ‘close.screen(all.screens=TRUE)‘ but is faster to type.

Usage

```
csa(silent = FALSE)
```

Arguments

silent	Should the output of ‘close.screen‘ not be returned?

Examples

```
# Split screen into fourths
split.screen(c(2,2))
hist(rnorm(100))
screen(2)
hist(runif(100))
# Use csa() to go back to normal plotting
csa()
hist(rexp(100))
```

eval_over_grid_with_batch *Evaluate function over grid of points*

Description

'batchmax' gives how many can be evaluated at a time. If more than 1, then the input is given to the function as rows of a matrix.

Usage

```
eval_over_grid_with_batch(x, y, fn, batchmax)
```

Arguments

x	Vector of x values to evaluate
y	Vector of y values to evaluate
fn	Function that takes in a length two vector if 'batchmax' is 1 or a matrix with two columns if greater than 1.
batchmax	Number of points that can evaluated simultaneously. If 1, points are passed to 'fn' as a vector of length two. If greater than 1, points are passed to 'fn' as rows of a matrix.

Value

Matrix of size 'length(x)' by 'length(y)'

Examples

```
eval_over_grid_with_batch(c(0,.5,1), c(10,20,30), function(a)a[1]+a[2], batchmax=1)
eval_over_grid_with_batch(c(0,.5,1), c(10,20,30), function(a)a[,1]+a[,2], batchmax=Inf)
```

gcf *Make contour plot from data or function using ggplot2*

Description

Simpler function for making contours with cf package. Won't give argument completion, so all must be specified

Usage

```
gcf(...)
```

Arguments

...	Arguments to be passed to cf_func or cf_data based on data type of first argument. If D is given as argument, then it is passed to cf_highdim.

Value

Whatever is returned from other function, probably nothing. Will be a ggplot2 object if using gg=TRUE.

Examples

```
gcf(function(x){x[1]^2 - x[2]})
x <- runif(20)
y <- runif(20)
z <- exp(-(x-.5)^2-5*(y-.5)^2)# + rnorm(20,0,.05)
gcf(x,y,z)
gcf(function(x){x[1]^2 - x[2]}, D=3)
```

---

gcf_data *Contour plot from data*

Description

Makes filled contour plot from data without sidebar by interpolating with a Gaussian process model. This is the same as ‘cf_data‘ except it will use ggplot2 to make the plot.

Usage

```
gcf_data(
  x,
  y = NULL,
  z = NULL,
  xlim = NULL,
  ylim = NULL,
  xylim = NULL,
  fit = "",
  gg = TRUE,
  ...
)
```

Arguments

x	either just x data, x and y data, or x, y and z data
y	either y data, z data, or null
z	either z data or null
xlim	x limits for the contour plot, will be set to data limits +- 5% if not specified

ylim	y limits for the contour plot, will be set to data limits +- 5% if not specified
xylim	x and y limits for the contour plot
fit	Method to fit a model with. Current options are laGP (default) and mlegp. laGP is faster but might cause trouble.
gg	If FALSE, will use base graphics by calling cf_func()
...	passed to cf_func

Examples

```
x <- runif(20)
y <- runif(20)
z <- exp(-(x-.5)^2-5*(y-.5)^2)
gcf_data(x,y,z)
```

---

gcf_func *Makes filled contour plot from function*

Description

A contour plot of the given function without sidebar by default. It calls the function ‘cf_grid‘ to make the actual plot.

Usage

```
gcf_func(
  fn0,
  n = 100,
  xlim = c(0, 1),
  ylim = c(0, 1),
  xylim = NULL,
  batchmax = 1,
  out.col.name = NULL,
  out.name = NULL,
  pts = NULL,
  ...
)
```

Arguments

fn0	function to plot, first argument must be two-dimensional
n	number of points in each dimension
xlim	x limits for the contour plot
ylim	y limits for the contour plot
xylim	x and y limits for the contour plot, use when both are same #@param mainmin-max whether the min and max values should be shown in the title of plot

batchmax	number of datapoints that can be computed at a time
out.col.name	if a column needs to be selected from the function, specify it
out.name	Selects with a $ the name from output to be used, for lists and data frames #@param pretitle Text to be preappended to end of plot title #@param posttitle Text to be appended to end of plot title #@param title Title for the plot #@param mainminmax_minmax Whether [min,max]= should be shown in title or just the numbers
pts	Points to plot on top of contour
...	Passed to cf_grid

Examples

```
gcf_func(function(x){x[1]*x[2]})
gcf_func(function(x)(exp(-(x[1]-.5)^2-5*(x[2]-.5)^2)))
gcf_func(function(xx){exp(-sum((xx-.5)^2/.1))}, bar=TRUE, color.palette=terrain.colors)
gcf_func(function(xx){exp(-sum((xx-.5)^2/.1))}, bar=TRUE, mainminmax=TRUE)
gcf_func(function(x)(exp(-(x[1]-.5)^2-5*(x[2]-.5)^2)))
```

---

gcf_grid *Create contour plot from grid data using ggplot2*

Description

The same as cf_grid_screen but uses ggplot2 for the plot.

Usage

```
gcf_grid(
  x = seq(0, 1, length.out = nrow(z)),
  y = seq(0, 1, length.out = ncol(z)),
  z,
  xlim = range(x, finite = TRUE),
  ylim = range(y, finite = TRUE),
  zlim = range(z, finite = TRUE),
  with_lines = FALSE,
  lines_only = FALSE,
  bins = 8,
  interpolate = TRUE,
  levels = pretty(zlim, nlevels),
  nlevels = 20,
  color.palette = cm.colors.strong,
  col = color.palette(length(levels) - 1),
  asp = NA,
  las = 1,
  bar = F,
  pts = NULL,
```

```
  reset.par = TRUE,
  pretitle = "",
  posttitle = "",
  main = NULL,
  mainminmax = !bar,
  mainminmax_minmax = TRUE,
  afterplotfunc = NULL,
  cex.main = par()$cex.main,
  ...
)
```

Arguments

x	x values, must form grid with y. If not given, it is assumed to be from 0 to 1.
y	y values, must form grid with x. If not given, it is assumed to be from 0 to 1.
z	z values at grid locations
xlim	x limits for the plot.
ylim	y limits for the plot.
zlim	z limits for the plot.
with_lines	Should lines be added on top of contour to show contours?
lines_only	Should no fill be used, only contour lines?
bins	Number of lines used when using 'with_lines' or 'lines_only'
interpolate	Will smooth out contours
levels	a set of levels which are used to partition the range of z. Must be strictly increasing (and finite). Areas with z values between consecutive levels are painted with the same color.
nlevels	if levels is not specified, the range of z, values is divided into approximately this many levels.
color.palette	a color palette function to be used to assign colors in the plot. Defaults to cm.colors. Other options include rainbow, heat.colors, terrain.colors, topo.colors, and function(x) {gray((1:x)/x)}.
col	an explicit set of colors to be used in the plot. This argument overrides any palette function specification. There should be one less color than levels
asp	the y/x aspect ratio, see plot.window.
las	the style of labeling to be used. The default is to use horizontal labeling.
bar	Should a bar showing the output range and colors be shown on the right?
pts	Points to plot on top of contour
reset.par	Should the graphical parameters be reset before exiting? Usually should be unless you need to add something to the plot afterwards and bar is TRUE.
pretitle	Text to be preappended to end of plot title
posttitle	Text to be appended to end of plot title
main	Title for the plot

mainminmax	whether the min and max values should be shown in the title of plot
mainminmax_minmax	Whether [min,max]= should be shown in title or just the numbers
afterplotfunc	Function to call after plotting, such as adding points or lines.
cex.main	The size of the main title. 1.2 is default.
...	additional graphical parameters, currently only passed to title().

Value

ggplot2 object

Examples

```
x <- y <- seq(-4*pi, 4*pi, len = 27)
r <- sqrt(outer(x^2, y^2, "+"))
gcf_grid(cos(r^2)*exp(-r/(2*pi)))
gcf_grid(r, color.palette=heat.colors, bar=TRUE)
gcf_grid(r, color.palette=function(x) {gray((1:x)/x)}, bar=TRUE)
```

---

multicolor.title *Makes plot title using specified colors for the text*

Description

Makes plot title using specified colors for the text

Usage

```
multicolor.title(main, col.main, collapse = "", cex.main = par()$cex.main)
```

Arguments

main	Text to put in main title of plot
col.main	Colors to use for the text
collapse	What to put between elements of main, defaults to "" but " " might be appropriate
cex.main	The size of the main title. 1.2 is default.

Examples

```
plot(1:4)
multicolor.title(c('Black, ','red, ','green'),c(1,2,3))
```

`text_plot` *Make a plot with only text*

Description

Make a plot with only text

Usage

```
text_plot(p, x = 0.5, y = 0.5, cex = 2, ...)
```

Arguments

`p`	Text to put on a plot
`x`	x-value of center of text, defaults to center
`y`	y-value of center of text, defaults to center
`cex`	Size of text
`...`	Arguments passed to plot

References

ZNK's answer on https://stackoverflow.com/questions/19918985/r-plot-only-text, retrieved 5/25/2018

Examples

```
text_plot("Useful?", cex=5)
```

Index