Package 'frenchCurve'

June 26, 2020

Type Package

Title Generate Open or Closed Interpolating Curves

Version 0.1.0

Author Bill Venables

Maintainer Bill Venables <Bill.Venables@gmail.com>

Description Functions for finding smooth interpolating curves connecting a series of points in the plane. Curves may be open or closed, that is, with the first and last point of the curve at the initial point.

License GPL-2

Imports graphics, stats, sp

Depends grDevices

Encoding UTF-8

LazyData true

RoxygenNote 7.1.0

NeedsCompilation no

Repository CRAN

Date/Publication 2020-06-26 08:40:02 UTC

R topics documented:

as_polygon *Make a Simple Polygon or Points*

Description

A simple polygon is here defined as a data frame with numeric components x and y without any duplicate rows. The order of rows is significant in defining the associated figure.

Usage

```
as_polygon(x, y = NULL, ...)

## Default S3 method:
as_polygon(x, y = NULL, ...)

## S3 method for class 'curve'
as_polygon(x, y = NULL, ...)

as_points(x, y = NULL)
```

Arguments

x, y	any specification of 2-d points, or a "curve" object
...	additional arguments not currently used

Details

A 'points' object is defined as a data frame with numeric columns x and y.

Value

a data frame with components x and y

open_curve *Curved Interpolation*

Description

Interpolate between ordered 2-d points with a smooth curve. open_curve() produces an open curve; closed_curve() produces a closed curve. Bezier curves are also provided.

Usage

```
open_curve(x, y = NULL, n = 100 * length(z), asp = 1, ...)

## S3 method for class 'curve'
plot(x, y = NULL, type = "l", lty = "solid", ...)

## S3 method for class 'curve'
points(x, pch = 20, ...)

## S3 method for class 'curve'
lines(x, ...)

closed_curve(x, y = NULL, n0 = 100 * length(z0), ...)

bezier_curve(x, y = NULL, n = 500, t = seq(0, 1, length.out = n), ...)
```

Arguments

x, y	Any of the forms used to specify a 2-d set of points or an object of class "curve"
n, n0	number of points in the interpolating curve
asp	the relative scale for x versus that of y
...	additional arguments currently ignored
pch, type, lty	plot arguments or traditional graphics parameters
t	for Bezier curves, parameter value sequence ranging from 0 to 1

Value

a list with components x, y, and points, of S3 class "curve"

Examples

```
oldPar <- par(pty = "s", mfrow = c(2, 2), mar = c(1,1,2,1), xpd = NA)
z <- (complex(argument = seq(-0.9*base::pi, 0.9*base::pi, length = 20)) +
     complex(modulus = 0.125, argument = runif(20, -base::pi, base::pi))) *
     complex(argument = runif(1, -base::pi, base::pi))

plot(z, asp=1, axes = FALSE, ann = FALSE, panel.first = grid())
title(main = "Open")
segments(Re(z[1]), Im(z[1]), Re(z[20]), Im(z[20]), col = "grey", lty = "dashed")
lines(open_curve(z), col = "red")

plot(z, asp=1, axes = FALSE, ann = FALSE, panel.first = grid())
title(main = "Closed")
lines(closed_curve(z), col = "royal blue")

plot(z, asp=1, axes = FALSE, ann = FALSE, panel.first = grid())
title(main = "Bezier")
lines(bezier_curve(z), col = "dark green")
```

```
plot(z, asp=1, axes = FALSE, ann = FALSE, panel.first = grid())
title(main = "Circle")
lines(complex(argument = seq(-base::pi, base::pi, len = 500)),
      col = "purple")

par(oldPar)
```

`%inside%` *Check if points lie inside a simple polygon*

Description

Check if points lie inside a simple polygon

Usage

```
points %inside% polygon
```

Arguments

`points`	a data.frame with components x,y specifying the points
`polygon`	a data.frame with components x,y specifying the polygon

Value

a logical value matching the number of points, TRUE = "inside"

Examples

```
oldPar <- par(pty = "s", las = 1, xpd = NA)
pts <- expand.grid(x = seq(0, 1, len=25), y = seq(0, 1, len=25))
pol <- (1 + 1i)/2 + complex(argument = seq(-base::pi, base::pi, len=100))/3
show_red <- as_points(pts) %inside% as_polygon(pol)
plot(pts, col = ifelse(show_red, "red", "royal blue"), ann = FALSE, bty = "n",
     pch = ".", cex = ifelse(show_red, 4, 2.5), asp = 1)
polygon(pol, lwd = 0.5)
par(oldPar)
```

Index