Package 'interpolators'

November 10, 2023

Title Some Interpolation Methods

Version 1.0.1

Description Some interpolation methods taken from 'Boost': barycentric rational interpolation, modified Akima interpolation, PCHIP (piecewise cubic Hermite interpolating polynomial) interpolation, and Catmull-Rom splines.

License GPL-3

URL https://github.com/stla/interpolators

BugReports https://github.com/stla/interpolators/issues

Imports Rcpp

LinkingTo BH, Rcpp

Encoding UTF-8

RoxygenNote 7.2.3

NeedsCompilation yes

Author Stéphane Laurent [aut, cre]

Maintainer Stéphane Laurent <laurent_step@outlook.fr>

Repository CRAN

Date/Publication 2023-11-10 19:33:19 UTC

R topics documented:

evalInterpolator *Interpolator evaluation*

Description

Evaluation of an interpolator at some given values.

Usage

```
evalInterpolator(ipr, x, derivative = 0)
```

Arguments

ipr	an interpolator
x	numeric vector giving the values to be interpolated; missing values are not allowed; for Catmull-Rom splines, the values must be between 0 and 1
derivative	order of differentiation, 0 or 1

Value

Numeric vector of interpolated values, or numeric matrix of interpolated points for the Catmull-Rom interpolator.

iprBarycentricRational *Barycentric rational interpolator*

Description

Barycentric rational interpolator.

Usage

```
iprBarycentricRational(x, y, ao = 3)
```

Arguments

x, y	numeric vectors giving the coordinates of the known points, without missing value
ao	approximation order, an integer greater than or equal to 3

Details

See Barycentric rational interpolation.

Value

An interpolator, for usage in evalInterpolator.

Examples

```
library(interpolators)
x <- c(1, 2, 4, 5)
y <- x^2
ipr <- iprBarycentricRational(x, y)
evalInterpolator(ipr, c(2, 3))
evalInterpolator(ipr, c(2, 3), derivative = 1)
```

---

iprCatmullRom *Catmull-Rom interpolator*

Description

Catmull-Rom interpolator for 2-dimensional or 3-dimensional points.

Usage

```
iprCatmullRom(points, closed = FALSE, alpha = 0.5)
```

Arguments

points	numeric matrix of 2D or 3D points, one point per row
closed	Boolean, whether the curve is closed
alpha	parameter between 0 and 1; the default value 0.5 is recommended

Details

See Catmull-Rom splines.

Value

An interpolator, for usage in evalInterpolator.

Examples

```
library(interpolators)
points <- rbind(
  c(0, 2.5),
  c(2, 4),
  c(3, 2),
  c(4, 1.5),
  c(5, 6),
  c(6, 5),
  c(7, 3),
```

```
  c(9, 1),
  c(10, 2.5),
  c(11, 7),
  c(9, 5),
  c(8, 6),
  c(7, 5.5)
)
ipr <- iprCatmullRom(points)
s <- seq(0, 1, length.out = 400)
Curve <- evalInterpolator(ipr, s)
head(Curve)
plot(Curve, type = "l", lwd = 2)
points(points, pch = 19)

# a closed example (pentagram) ####
rho <- sqrt((5 - sqrt(5))/10)
R <- sqrt((25 - 11*sqrt(5))/10)
points <- matrix(NA_real_, nrow = 10L, ncol = 2L)
points[c(1, 3, 5, 7, 9), ] <- t(vapply(0:4, function(i){
  c(rho*cospi(2*i/5), rho*sinpi(2*i/5))
}, numeric(2L)))
points[c(2, 4, 6, 8, 10), ] <- t(vapply(0:4, function(i){
  c(R*cospi(2*i/5 + 1/5), R*sinpi(2*i/5 + 1/5))
}, numeric(2L)))
ipr <- iprCatmullRom(points, closed = TRUE)
s <- seq(0, 1, length.out = 400L)
Curve <- evalInterpolator(ipr, s)
plot(Curve, type = "l", lwd = 2, asp = 1)
points(points, pch = 19)
```

`iprMakima` *Modified Akima interpolator*

Description

Modified Akima interpolator.

Usage

```
iprMakima(x, y)
```

Arguments

`x, y`	numeric vectors giving the coordinates of the known points, without missing value

Details

See Modified Akima interpolation.

Value

An interpolator, for usage in evalInterpolator.

Examples

```
library(interpolators)
x <- seq(0, 4*pi, length.out = 9L)
y <- x - sin(x)
ipr <- iprMakima(x, y)
curve(x - sin(x), from = 0, to = 4*pi, lwd = 2)
curve(
  evalInterpolator(ipr, x),
  add = TRUE, col = "blue", lwd = 3, lty = "dashed"
)
points(x, y, pch = 19)
```

---

iprPCHIP *PCHIP interpolator*

Description

PCHIP interpolator. It is monotonic.

Usage

```
iprPCHIP(x, y)
```

Arguments

x, y	numeric vectors giving the coordinates of the known points, without missing value

Details

See PCHIP interpolation.

Value

An interpolator, for usage in evalInterpolator.

Examples

```
library(interpolators)
x <- seq(0, 4*pi, length.out = 9L)
y <- x - sin(x)
ipr <- iprPCHIP(x, y)
curve(x - sin(x), from = 0, to = 4*pi, lwd = 2)
curve(
  evalInterpolator(ipr, x),
```

```
  add = TRUE, col = "blue", lwd = 3, lty = "dashed"
)
points(x, y, pch = 19)
```

Index