Package 'ICSsmoothing'

January 14, 2021

Type Package

Title Data Smoothing by Interpolating Cubic Splines

Version 1.2.5

Description We construct the explicit form of clamped cubic interpolating spline (both uniform - knots are equidistant and non-uniform - knots are arbitrary). Using this form, we propose a linear regression model suitable for real data smoothing. This package is based on a diploma thesis Hudak (2017) <https://opac.crzp.sk/?fn=detailBiblioForm&sid=7FE4E0810F66E41DF9A9CA23557F&seo=CRZP-detail-kniha>.

Depends R (>= 3.5.0), polynom, ggplot2

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

Suggests knitr, rmarkdown, testthat

VignetteBuilder knitr

NeedsCompilation no

Author Juraj Hudak [aut],
Viliam Kacala [cre],
Csaba Torok [ctb]

Maintainer Viliam Kacala <viliam.kacala.ml@gmail.com>

Repository CRAN

Date/Publication 2021-01-14 09:10:09 UTC

R topics documented:

`CERN` *277 measurements of the cross sections for $\pi\hat{}-p$ collision (nuclear physics).*

Description

277 measurements of the cross sections for $\pi^- p$ collision (nuclear physics).

Usage

```
CERN
```

Format

A data frame with 277 elements.

Source

https://link.springer.com/article/10.1007/BF02683433

`cics_explicit` *Construct the explicit form of non-uniform clamped interpolating cubic spline (NcICS).*

Description

`cics_explicit` constructs the explicit form of non-uniform clamped interpolating cubic spline (via Hermite cubic spline) for nodes `uu`, function values `yy` and exterior-node derivatives `d`.

Usage

```
cics_explicit(
  uu,
  yy,
  d,
  clrs = c("blue", "red"),
  xlab = NULL,
  ylab = NULL,
  title = NULL
)
```

Arguments

uu	a vector of arbitrary nodes (ordered ascendingly), with magnitude n+2, n$\geq$1.
yy	a vector of function values pertaining to nodes in uu.
d	a vector of two values of derivative, in the first and the last node of uu.
clrs	a vector of colours that are used alternately to plot the graph of spline's components.
xlab	a title (optional parameter) for the x axis.
ylab	a title (optional parameter) for the y axis.
title	a title (optional parameter) for the plot.

Value

a list with components

spline_coeffs	matrix, whose i-th row contains coefficients of non-uniform ICS's i-th component.
spline_polynomials	list of NcICS's components string representations.
B	4-element array of (n+1)x(n+4) matrices, whereas element in i-th row and j-th column of l-th matrix contains coefficient by x^{l-1} of cubic polynomial that is in i-th row and j-th column of matrix B from spline's explicit form $$S = B.\gamma.$$
gamma	γ = vector of spline coefficients - function values and exterior-node derivatives that takes part in the explicit form $S = B.\gamma$.
aux_BF	A basis function of the spline
aux_tridiag_inverse	An inverse of the tridiagonal matrix used for spline derivatives construction

Examples

```
cics_explicit(
 uu = c(1, 2.2, 3, 3.8, 7),
 CERN$y[1:5],
 d=c(0,-2),
 xlab="X axis",
 ylab="Y axis"
)

uu <- c(0, 1, 4, 6);
yy <- c(4, 5, 2, 1.8);
sp <- cics_explicit(uu, yy, c(1,0))
sp$spline_polynomials
### <~~>
### Spline components' coefficients
explicit_spline(sp$B, sp$gamma)
sp$spline_coeffs == .Last.value
```

cics_explicit_smooth *Smooth given data set by k-component non-uniform clamped interpolating spline (NcICS).*

Description

cics_explicit_smooth constructs the non-uniform clamped interpolating spline with k components that smoothes given data set {(xx[i],yy[i]),i=1..length(xx)}.

Usage

```
cics_explicit_smooth(
  xx,
  yy,
  uu,
  clrs = c("blue", "red"),
  d,
  xlab = NULL,
  ylab = NULL,
  title = NULL
)
```

Arguments

xx	a vector of data set's x-coordinates (that are in increasing order).
yy	a vector of data set's y-coordinates.
uu	a vector of arbitrary nodes, based on which we construct the smoothing spline. uu[1] and uu[length(uu)] must be equal to xx[1] and xx[length(xx)], respectively.
clrs	a vector of colours that are used alternately to plot the graph of spline's components.
d	a vector (optional parameter) that contains two values of derivative, in the first and the last node from uu. If missing, values of derivative are estimated by given linear regression model. If present, their contribution is removed from linear model and only function values are estimated.
xlab	a title (optional parameter) for the x axis.
ylab	a title (optional parameter) for the y axis.
title	a title (optional parameter) for the plot.

Value

a list with components

`est_spline_coeffs`
: 4-element array of `(k)x(k+3)` matrices, whereas element in `i`-th row and `j`-th of `l`-th matrix contains coefficient by `x^{l-1}` of cubic polynomial, which is in `i`-th row and `j`-th column of matrix `B` from smoothing spline's explicit form

$$S = B.\gamma.$$

`est_spline_polynomials`
: list of string representations of smoothing NcICS.

`est_gamma`
: vector of estimated smoothing spline's coefficients (function values and exterior-node derivatives).

`aux_BF`
: A basis function of the spline

`aux_tridiag_inverse`
: An inverse of the tridiagonal matrix used for spline derivatives construction

`aux_M`
: An estimation matrix used to compute `est_gamma`

Examples

```
cics_explicit_smooth(
xx = CERN$x,
yy = CERN$y,
d = c(0, 1),
uu = c(1, sort(runif(20,1,277)), 277),
xlab = "X axis",
ylab = "Y axis"
)

yy <- c(1, 2, 3, 4, 3, 2, 2, 3, 5, 6, 7, 6, 5, 5, 4, 3, 2, 1, 0)
xx <- c(1:length(yy))
uu <- c(1,7,10,19)
sp <- cics_explicit_smooth(xx,yy,uu)
### We can change the derivatives at the end nodes:
sp <- cics_explicit_smooth(xx,yy, uu, d=c(3,-7/10))

### CERN:
uu <- c(1, 15, 26, 63, 73, 88, 103, 117, 132, 200, 203, 219, 258, 277)
sp <- cics_explicit_smooth(
  xx = CERN$x,
  yy = CERN$y,
  d = c(1, 0),
  uu
)
```

`cics_unif_explicit` *Construct the explicit form of uniform clamped interpolating cubic spline (UcICS).*

Description

cics_unif_explicit constructs the explicit form of uniform clamped interpolating cubic spline (via Hermite cubic spline) for nodes uu, function values yy and exterior-node derivatives d.

Usage

```
cics_unif_explicit(
  uumin,
  uumax,
  yy,
  d,
  clrs = c("blue", "red"),
  xlab = NULL,
  ylab = NULL,
  title = NULL
)
```

Arguments

uumin	a starting node.
uumax	an ending node.
yy	a vector of function values pertaining to nodes in uu.
d	a vector of two values of derivative, in the first and the last node of uu.
clrs	a vector (optional parameter) of colours that are used alternately to plot the graph of spline's components.
xlab	a title (optional parameter) for the x axis.
ylab	a title (optional parameter) for the y axis.
title	a title (optional parameter) for the plot.

Value

A list of spline components

spline_coeffs	matrix, whose i-th row contains coefficients of uniform ICS's i-th component.
spline_polynomials	list of UcICS's components string representations.
B	4-element array of (n+1)x(n+4) matrices, whereas element in i-th row and j-th column of l-th matrix contains coefficient by x^{l-1} of cubic polynomial that is in i-th row and j-th column of matrix B from spline's explicit form $$S = B.\gamma.$$
gamma	γ = vector of spline coefficients - function values and exterior-node derivatives that takes part in the explicit form $S = B.\gamma$.
aux_BF	A basis function of the spline
aux_tridiag_inverse	An inverse of the tridiagonal matrix used for spline derivatives construction

Examples

```
yy <- c(4, 5, 2, 1.8);
sp <- cics_unif_explicit(0, 6, yy, c(2, 0.9))
sp$spline_polynomials
### <~~>
### Spline components' coefficients
explicit_spline(sp$B, sp$gamma)
sp$spline_coeffs == .Last.value
```

`cics_unif_explicit_smooth` *Smooth given data set by k-component uniform clamped interpolating spline (UcICS).*

Description

`cics_unif_explicit_smooth` constructs the uniform clamped interpolating spline with `k` components that smoothes given data set `{(xx[i],yy[i]),i=1..length(xx)}`.

Usage

```
cics_unif_explicit_smooth(
  xx,
  yy,
  k,
  clrs = c("blue", "red"),
  d,
  xlab = NULL,
  ylab = NULL,
  title = NULL,
  plotTF = TRUE
)
```

Arguments

`xx`	a vector of data set's `x`-coordinates (that are in increasing order).
`yy`	a vector of datanvidi set's `y`-coordinates.
`k`	a chosen number of components of smoothing UcICS (integer ≥ 2).
`clrs`	a vector of colours that are used alternately to plot the graph of spline's components.
`d`	a vector (optional parameter) that contains two values of derivative, in the first and the last computed node. If missing, values of derivative are estimated by given linear regression model. If present, their contribution is removed from linear model and only function values are estimated.
`xlab`	a title (optional parameter) for the `x` axis.

ylab	a title (optional parameter) for the y axis.
title	a title (optional parameter) for the plot.
plotTF	a boolean value (optional parameter), if TRUE then plot.

Value

a list with components

nodes	vector of equidistant nodes, based on which we construct the smoothing spline.
est_spline_coeffs	4-element array of (k)x(k+3) matrices, whereas element in i-th row and j-th of l-th matrix contains coefficient by x^{l-1} of cubic polynomial, which is in i-th row and j-th column of matrix B from smoothing spline's explicit form $$S = B.\gamma.$$
est_spline_polynomials	list of string representations of smoothing UcICS.
est_gamma	vector of estimated smoothing spline's coefficients (function values and exterior-node derivatives).
aux_BF	A basis function of the spline
aux_tridiag_inverse	An inverse of the tridiagonal matrix used for spline derivatives construction
aux_M	An estimation matrix used to compute est_gamma

Examples

```
cp <- cics_unif_explicit_smooth(
xx = CERN$x,
  yy = CERN$y,
  k = 19, #23,
  d = c(1, 0),
  xlab = "X axis",
  ylab = "Y axis"
)
```

---

explicit_spline

The function computes the coefficients of the cubic polynomials as spline components of the clamped interpolating cubic spline of class C^2 *in its explicit form* S=B * gamma.

Description

The function computes the coefficients of the cubic polynomials as spline components of the clamped interpolating cubic spline of class C^2 in its explicit form S=B * gamma.

Usage

```
explicit_spline(B, gamma)
```

Arguments

B	a 4-element array of (n+1)x(n+4) matrices, whereas element in i-th row and j-th column of l-th matrix contains coefficient by x^{l-1} of cubic polynomial that is in i-th row and j-th column of matrix B from spline's explicit form S=B.gamma.
gamma	a vector of spline coefficients - function values and exterior-node derivatives that takes part in the explicit form $S = B.\gamma$.

Value

a matrix with four columns, whose i-th row contains the coefficients of the splines's i-th component.

Examples

```
# See functions cics_explicit, cics_unif_explicit and the vignette.
```

---

forecast_demo *Forecasting demo using cics_unif_explicit_smooth.*

Description

Forecasting demo using cics_unif_explicit_smooth.

Usage

```
forecast_demo()
```

Value

a forecast result

Examples

```
# Plots as well as the process of computation of future derivatives and values using extrapolation.
ud <- forecast_demo()
```

hermite_bf_matrix *Construct 4 Hermite basis functions.*

Description

hermite_bf_matrix constructs matrix of Hermite basis functions' coefficients on [u,v], that is the matrix of 4 cubic polynomials' coefficients of one-component Hermite cubic spline.

Usage

```
hermite_bf_matrix(u, v)
```

Arguments

u	a left border of interval [u,v].
v	a right border of interval [u,v], u$\leq$v.

Value

The matrix of 4 Hermite basis functions' coefficients.

Examples

```
hermite_bf_matrix(0,1)
hermite_bf_matrix(-2,3)
```

tridiag_inv_general *Construct inverse of a general tridiagonal matrix.*

Description

tridiag_inv_general constructs inverse of a general tridiagonal matrix M of order n, using Usmani's theorem.

Usage

```
tridiag_inv_general(M, n)
```

Arguments

M	a tridiagonal matrix.
n	an order of given tridiagonal matrix.

Value

The inverse of matrix M.

Examples

```
tridiag_inv_general(matrix(c(1, 4, 0, -9), 2, 2), 2)
tridiag_inv_general(matrix(c(1, 3, 5, -2, 0, 8, 7, 6, 6), 3, 3), 3)
```

`tridiag_inv_unif_by_sums` *Construct inverse of a tridiagonal matrix M_n(a,b,a).*

Description

`tridiag_inv_unif_by_sums` constructs inverse of a regular tridiagonal matrix `M_n(a,b,a)` with constant entries by a special algorithm using sums of matrix elements.

Usage

```
tridiag_inv_unif_by_sums(n, a, b)
```

Arguments

`n`	an order of given tridiagonal matrix.
`a`	a value of tridiagonal matrix elements that are off-diagonal.
`b`	a value of tridiagonal matrix diagonal elements.

Value

The inverse of matrix `M_n(a,b,a)`.

Examples

```
tridiag_inv_unif_by_sums(5, 1, 4)
tridiag_inv_unif_by_sums(9, 10, -1)
```

Index