

Package ‘monoProc’

January 15, 2012

Type Package

Title strictly monotone smoothing procedure

Version 1.0-6

Date 2007-10-23

Author Regine Scheder

Maintainer Regine Scheder <regine.scheder@rub.de>

Description monotonizes a given fit in one or two variables.

Depends R (>= 2.4.1), methods

Suggests locfit, UsingR, KernSmooth

LazyLoad yes

Imports graphics

License GPL (>= 2)

Repository CRAN

Date/Publication 2009-10-11 18:30:13

R topics documented:

cv	2
ksmooth	3
ksmooth-class	4
locfit-class	4
locpoly	5
locpoly-class	5
loess-class	6
mono.1d	6
mono.2d	8
monofit-class	9

monoproc	10
monoproc-class	12
monoproc.1d-class	13
monoproc.2d-class	14

Index	15
--------------	-----------

cv	<i>Cross validation function</i>
----	----------------------------------

Description

computes the cross validation value (leave one out method) for the monotized and the unconstrained fit

Usage

```
cv(fit, ...)
```

Arguments

fit	an object of class "monoproclocfit.1d" or "monoproclocfit.2d"
...	currently not in use

Details

so far, this function can only be used for objects of class "monoproc" if the originally fit came from the locfit-function. This function is currently not really computational efficient.

Value

returns a matrix where column 1 and 2 represents the values for the new fit and the fitold, respectively, under leave one out cross validation. The rows correspond to the observation number.

See Also

[monoproc](#)

Examples

```
if(require(UsingR)&&require(locfit)){
  data(fat)
  fat<-fat[-39,] ##two extreme observations
  fat<-fat[-41,] ##are deleted
  attach(fat)
  x<-as.matrix(cbind(weight, height))
  fit<-locfit.raw(x,body.fat.siri, alpha=0.3, deg=1, kern="epan")
  fitmono<-monoproc(fit,bandwidth=1,mono1="increasing", mono2="decreasing", dir="xy", gridsize=30)
  nf<- layout(matrix(c(1,1,1,2,2,3,3,3,4,4), 2, 5, byrow = TRUE))
```

```
layout.show(nf)
plot(fit, type="persp", theta = 135, phi = 30,col="lightblue",
     cex=0.7,main="unconstraint Bodyfat estimate")
plot(fit)
plot(fitmono,theta = 135, phi = 30, col="lightblue",cex=0.7,
     main="monotone Bodyfat estimate")
plot(fitmono, type="contour")
t<-cv(fitmono)
#Cross Validation for the unconstraint estimator
sum((t[,2]-body.fat.siri)^2)/250
#Cross Validation for the monotone estimator
sum((t[,1]-body.fat.siri)^2)/250
plot(seq(1:250),rep(1,250), type="l",col=2, xlab="observation index",
     ylab="Ratio of CV-unconstraint over CV-monotone")
points((t[,2]-body.fat.siri)^2/(t[,1]-body.fat.siri)^2)
}
```

ksmooth

ksmooth with S4-Class value

Description

changes the function [ksmooth](#) to S4-Class value

Usage

```
ksmooth(...)
```

Arguments

... usage described in the original help file.

Value

an object of class "ksmooth".

See Also

for the class "ksmooth" see [ksmooth](#). For details of the usage, please see [ksmooth](#).

ksmooth-class	<i>Class "ksmooth"</i>
---------------	------------------------

Description

S4-Class for the value of the ksmooth function

Objects from the Class

Objects can be created by calls of the form `new("ksmooth", ...)`. This class is the value of the function `ksmooth` transferred to a S4-Class.

Slots

x: Object of class "numeric". x-values of the smoothed fit in increasing order.

y: Object of class "numeric". The corresponding fitted y-values of the Nadaraya-Watson estimator.

call: Object of class "call". The call of the function `ksmooth` for this object.

Methods

lines signature(x = "ksmooth"): line method to plot this class.

plot signature(x = "ksmooth", y = "missing"): plot method for this class.

See Also

[ksmooth](#)

locfit-class	<i>Class "locfit"</i>
--------------	-----------------------

Description

S4-Class for the old class `locfit`

Methods

monoproc signature(fit = "locfit", bandwidth= "numeric", xx= "missing", dir= "missing"): monotizing method.

monoproc signature(fit = "locfit", bandwidth= "numeric", xx= "missing", dir= "character"): monotizing method.

monoproc signature(fit = "locfit", bandwidth= "numeric", xx= "numeric", dir= "missing"): monotizing method.

monoproc signature(fit = "locfit", bandwidth= "numeric", xx= "list", dir= "character"): monotizing method.

See Also[loffit](#)

locpoly	<i>locpoly with S4-Class value</i>
---------	------------------------------------

Description

changes the function [locpoly](#) to S4-Class value

Usage

```
locpoly(...)
```

Arguments

... usage described in the original help file.

Value

an object of class "locpoly"

See Also

for the class "locpoly" see [locpoly](#). For the usage, see [locpoly](#).

locpoly-class	<i>Class "locpoly"</i>
---------------	------------------------

Description

S4-Class for the value of the [locpoly](#) function

Objects from the Class

Objects can be created by calls of the form `new("locpoly", ...)`. This class is the value of the function [locpoly](#) transferred to a S4-Class.

Slots

x: Object of class "numeric". x-values of the smoothed fit in increasing order.

y: Object of class "numeric". The corresponding fitted y-values of the local polynomial estimator.

call: Object of class "call". The call of the function [locpoly](#) for this object.

Methods

lines signature(x = "locpoly"): line method to plot the class.

plot signature(x = "locpoly", y = "missing"): to plot this class.

See Also

[locpoly](#)

loess-class	<i>Class "loess"</i>
-------------	----------------------

Description

S4-Class for the old class loess

Methods

monoproc signature(fit = "loess", bandwidth= "numeric", xx= "missing", dir= "missing"): monotizing method.

monoproc signature(fit = "loess", bandwidth= "numeric", xx= "numeric", dir= "missing"): monotizing method.

See Also

[loess](#)

mono.1d	<i>strictly monotone and smooth function</i>
---------	--

Description

this function applies a kernel smoothing method to monotize a given fit with one independent variable

Usage

```
mono.1d(fit, bandwidth, xx, kernel = "epanech", mono1)
```

Arguments

fit	a list containing x-values and their corresponding y-values. The length of fit[[1]] determines degree of acuteness of the monotizing procedure
bandwidth	a single number which represents the kernel bandwidth smoothing parameter. Missing values are not accepted.
xx	an additional vector of x-values where the monotizing procedure is to be evaluated. If missing fit[[1]] is used instead.
kernel	"epanech" - the Epanechnikov kernel
mono1	either "increasing" or "decreasing"

Details

this function is used within "monoproc"

Value

returns an object of class "monofit"

Author(s)

Regine Scheder <Regine.Scheder@rub.de>

References

Dette, H., Neumeyer, N., and Pilz, K. (2004) *A simple nonparametric estimator of a monotone regression function.*

Dette, H. and Scheder, R. (2005) *Strictly monotone and smooth nonparametric regression for two or more variable.*

See Also

[mono.2d](#) and [monoproc](#)

Examples

```
data(cars)
speed<-cars$speed
dist<-cars$dist
fit1<-ksmooth(speed, dist, "normal", bandwidth=2)
##computes the Nadaraya-Watson estimate
fit2<-mono.1d(list(x=fit1@x, y=fit1@y),bandwidth=0.7, mono1="increasing") ##calculates the monotone estimates
plot(speed, dist)
lines(fit1, col=2)
lines(fit2, col=3)
```

 mono.2d

strictly monotone and smooth function

Description

this function applies a kernel smoothing method to monotone a given fit with two independent variable

Usage

```
mono.2d(fit, bandwidth, xx, kernel = "epanech", dir = "xy", mono1, mono2)
```

Arguments

fit	a list containing x-values and y-values as independent variables and their corresponding z-values. The length of fit[[1]] determines degree of acuteness of the monotone procedure
bandwidth	a single number which represents the kernel bandwidth smoothing parameter. Missing values are not accepted.
xx	an additional list of x-values and y-values where the monotone procedure is to be evaluated. If missing list(x=fit[[1]],y=fit[[2]]) is used instead.
kernel	"epanech" - the Epanechnikov kernel
dir	with respect to which variable and in which order the function is to be monotone. Possible is "x", "y", "xy", and "yx"
mono1	either "increasing" or "decreasing" for x-variable
mono2	either "increasing" or "decreasing" for y-variable

Details

this function is used within "monoproc"

Value

returns an object of class "monofit"

Author(s)

Regine Scheder <Regine.Scheder@rub.de>

References

Dette, H., Neumeyer, N., and Pilz, K. (2004) *A simple nonparametric estimator of a monotone regression function.*

Dette, H. and Scheder, R. (2005) *Strictly monotone and smooth nonparametric regression for two or more variable.*

See Also

[mono.1d](#) and [monoproc](#)

monofit-class

Class "monofit"

Description

A object of class `monofit` is the output of either [mono.1d](#) or [mono.2d](#)

Details

This Class is usually used within the function `monoproc`. It is the value of the functions `mono.1d` and `mono.2d`, respectively, which are called in the `monoproc` function. For one independent variable in the regression function, the matrix in slot `z` is a 0 x 0 matrix and slot `y` corresponds to the values of the regression function.

Objects from the Class

Objects can be created by calls of the form `new("monofit", ...)`.

Slots

x: x-variable is of class "numeric"

y: y-variable is of class "numeric"

z: z-variable is of class "matrix"

Methods

lines signature(x = "monofit"): ...

plot signature(x = "monofit", y = "missing"): ...

See Also

[mono.1d](#), [mono.2d](#), [monoproc.1d](#), or [monoproc.2d](#)

monoproc

*Monotonizing procedure***Description**

this procedure monotonizes a given fit in either one variable or two

Usage

```
monoproc(fit, bandwidth, xx, dir,...)
```

Arguments

fit	so far an object either of class "list", "loess", "ksmooth", "locpoly", or "locfit"
bandwidth	a single smoothing parameter of class "numeric". Missing values are not accepted.
xx	points where the monotone fit is to be evaluated. If missing the independent variables of fit are used instead.
dir	gives the order, direction, and the variable for twodimensional problems. This variable has to be missing for onedimensional problems otherwise there is an error. Possible values for dir are "x", "y", "xy", and "yx".
...	further parameters about the monotonization procedure. See also in Details.

Details

If fit is of class "list", the independent variable should be equidistant. The length of a list object can be either 2 (one independent variable as first element) or 3 (two independent variables as first two elements).

In twodimensional problems, the value "x" for dir refers to a single monotonization with respect to the first variable and "y" correspondingly to the second variable. With the values "xy" and "yx", the order of monotonization can be determined. The difference of the resulting monotone fits may be marginal, but in some cases of interest.

The kernel of the monotonizing procedure can be specified as well. Currently, only the Epanechnikov kernel is implemented which is the default value of kernel.

With the arguments mono1 and mono2, the kind of monotonization can be specified - either "increasing" or "decreasing". mono1 refers to the first independent variable and correspondingly mono2 to the second one. For an onedimensional problem only mono1 can be specified. The default value for mono1 and mono2 is "increasing".

The gridsize can be further specified, if the fit is not of class "list". However, do not use too large gridsizes. A gridsize larger than 50 is for twodimensional problems not recommended.

Value

An object of class [monoproc.1d-class](#) or [monoproc.2d-class](#) according to the number of independent variables

Author(s)

Regine Scheder <Regine.Scheder@rub.de>

References

Dette, H., Neumeyer, N., and Pilz, K. (2004) *A simple nonparametric estimator of a monotone regression function.*

Dette, H. and Scheder, R. (2005) *Strictly monotone and smooth nonparametric regression for two or more variable.*

See Also

[monoproc.1d](#) and [monoproc.2d](#)

Examples

```
#Spencer's Mortality Dataset
if(require(locfit)){
data(spencer)
attach(spencer)
fit<-locfit.raw(age,mortality, alpha=0.3, kern="epan")
fitmono<-monoproc(fit, bandwidth=0.0003, gridsize=30)
plot(age,mortality)
lines(fit)
lines(fitmono, col=2)}

#Fat Data to predict Bodyfat (two independent variables)
if(require(UsingR)&&require(locfit)){
data(fat)
fat<-fat[-39,] ##two extreme observations
fat<-fat[-41,] ##are deleted
attach(fat)
fit<-locfit.raw(cbind(weight, height),body.fat.siri, alpha=0.3, deg=1, kern="epan")
fitmono<-monoproc(fit,bandwidth=1, mono1= "increasing", mono2="decreasing", dir="xy", gridsize=30)
nf<- layout(matrix(c(1,1,1,2,2,3,3,3,4,4), 2, 5, byrow = TRUE))
layout.show(nf)
plot(fit, type="persp", theta = 135, phi = 30,col="lightblue",
      cex=0.7,main="unconstraint Bodyfat estimate")
plot(fit)
plot(fitmono,theta = 135, phi = 30, col="lightblue",cex=0.7,
      main="monotone Bodyfat estimate")
plot(fitmono, type="contour")
t<-cv(fitmono)
CV<-sum((t[,2]-body.fat.siri)^2)/250 #Cross Validation for the unconstraint estimator
CV2<-sum((t[,1]-body.fat.siri)^2)/250 #Cross Validation for the monotone estimator
}

#Two Examples about the cars Data
#first example somehow trivial since the loess-fit is already monotone increasing
cars.lo <- loess(dist ~ speed, cars,
```

```

control = loess.control(surface = "direct")
predict<-predict(cars.lo, data.frame(speed = seq(5, 30, 1)))
plot(cars.lo, xlab="speed", ylab="dist")
lines(seq(5, 30, 1),predict)
monofit<-monoproc(cars.lo, bandwidth=0.3, gridsize=40)
lines(monofit,col=2)

data(cars)
speed<-cars$speed
dist<-cars$dist
fit1<-ksmooth(speed, dist, "normal", bandwidth=2)
##computes the Nadaraya-Watson estimate
fit2<-monoproc(fit1,bandwidth=0.7)
##calculates the monotone estimates
      plot(speed, dist)
      lines(fit1, col=2)
      lines(fit2, col=3)

#Comparison with R-function isoreg
fit1<-ksmooth(speed, dist, "normal", bandwidth=2.5)
fit2<-monoproc(fit1,bandwidth=0.7)
fit3<-isoreg(speed,dist)

plot(fit3,plot.type="single", main="monotone regression", xlab="speed", ylab="distance")
  lines(fit1, col=3, lwd=1.5)
lines(fit2, lwd=1.5)
legend(5,100, c("isoreg", "ksmooth", "monoproc"),col=c(2,3,1), lty=c(1,1,1))

```

monoproc-class	<i>Class "monoproc"</i>
----------------	-------------------------

Description

A representation of a monotonized fit with information about the original fit and more

Details

For further details please see also [monoproc.1d](#) and [monoproc.2d](#)

Objects from the Class

Objects can be created by calls of the form `new("monoproc", ...)`.

Slots

`fit`: Object of class "monofit"

`fitold`: Object of class "fit"

gridsize: Object of class "numeric"
 bandwidth: Object of class "numeric"
 kernel: Object of class "character"
 mono: Object of class "character"
 name: Object of class "character"
 call: Object of class "call"

monoproc.1d-class *Class "monoproc.1d"*

Description

A representation of a monotonized fit with information about the original fit and more

Details

the class "monoproclocfit.1d" is a "monoproc.1d"-object where the original fit is of class "locfit". Both "monoproc.1d" as well as "monoproclocfit.1d" belong to the class "monoproc"

Objects from the Class

Objects can be created by calls of the form `new("monoproc.1d", ...)`.

Slots

fit: Object of class "monofit" where `fit@z` is null. This is the new monotonized fit.
fitold: Object of class "fit". This slot represents the old fit and can be of class "list", "locfit", "ksmooth", "locpoly", or "loess".
gridsize: Object of class "numeric"
bandwidth: Object of class "numeric". The used bandwidth for the monotonizing procedure.
kernel: Object of class "character". The used kernel. Currently only "epanech".
mono: Object of class "character". Either "increasing" or "decreasing"
name: Object of class "character". Names of the variables.
call: Call of this object.

Methods

lines signature(`x = "monoproc.1d"`): adds the line of the monotone fit to a given plot.
plot signature(`x = "monoproc.1d"`, `y = "missing"`): plots a monotone fit.
print signature(`x = "monoproc.1d"`): prints a monotone fit.
summary signature(`object = "monoproc.1d"`): gives a summary of the monotone fit.
cv signature(`fit = "monoproclocfit.1d"`): calculates the cross validation value of the monotone and the unconstrained fit.

See Also

[monoproc.2d](#)

monoproc.2d-class *Class "monoproc.2d"*

Description

A representation of a monotonized fit with information about the original fit and more

Details

the class "monoproclocfit.2d" is a "monoproc.2d"-object where the original fit is of class "locfit". Both "monoproc.2d" as well as "monoproclocfit.2d" belong to the class "monoproc"

Objects from the Class

Objects can be created by calls of the form `new("monoproc.2d", ...)`.

Slots

fit: Object of class "monofit" . This is the new monotonized fit .
fitold: Object of class "fit" . This slot represents the old fit and can be of class "list", "locfit", "ksmooth" , "locpoly", or "loess".
gridsize: Object of class "numeric" .
bandwidth: Object of class "numeric". The used bandwidth for the monotonizing procedure.
kernel: Object of class "character". The used kernel. Currently only "epanech".
mono: Object of class "character".
dir: Directions and order of the monotonization. Possible values are "xy", "yx", "x", or "y".
name: Object of class "character". Names of the variables.
call: Call of this object.

Methods

plot signature(x = "monoproc.2d", y = "missing"):plots the monotone fit. Three types are available: "persp", "contour", "image". Default is "persp".
print signature(x = "monoproc.2d"): prints the monotone fit.
summary signature(object = "monoproc.2d"): gives a summary of the monotone fit.
cv signature(fit= "monoproclocfit.2d"): calculates the cross validation value for the monotone and the unconstraint fit.

See Also

[monoproc.1d](#)

Index

*Topic **classes**

- ksmooth-class, 4
- locfit-class, 4
- locpoly-class, 5
- loess-class, 6
- monofit-class, 9
- monoproc-class, 12
- monoproc.1d-class, 13
- monoproc.2d-class, 14

*Topic **nonparametric**

- cv, 2
- mono.1d, 6
- mono.2d, 8
- monoproc, 10

*Topic **regression**

- cv, 2
- ksmooth, 3
- locpoly, 5
- mono.1d, 6
- mono.2d, 8
- monoproc, 10

*Topic **smooth**

- ksmooth, 3
- locpoly, 5
- mono.1d, 6
- mono.2d, 8
- monoproc, 10

cv, 2

cv, monoproc.1d-method (cv), 2

cv, monoproc.2d-method (cv), 2

cv, monoproclocfit.1d-method (cv), 2

cv, monoproclocfit.2d-method (cv), 2

ksmooth, 3, 3, 4

ksmooth-class, 4

lines, ksmooth-method (ksmooth-class), 4

lines, locpoly-method (locpoly-class), 5

lines, monofit-method (monofit-class), 9

lines, monoproc.1d-method
(monoproc.1d-class), 13

locfit, 5

locfit-class, 4

locpoly, 5, 5, 6

locpoly-class, 5

loess, 6

loess-class, 6

mono.1d, 6, 9

mono.2d, 7, 8, 9

monofit-class, 9

monoproc, 2, 7, 9, 10

monoproc, fit, missing, ANY, ANY-method
(monoproc), 10

monoproc, ksmooth, numeric, missing, missing-method
(monoproc), 10

monoproc, ksmooth, numeric, numeric, missing-method
(monoproc), 10

monoproc, list, numeric, list, character-method
(monoproc), 10

monoproc, list, numeric, missing, character-method
(monoproc), 10

monoproc, list, numeric, missing, missing-method
(monoproc), 10

monoproc, list, numeric, numeric, missing-method
(monoproc), 10

monoproc, locfit, numeric, list, character-method
(monoproc), 10

monoproc, locfit, numeric, missing, character-method
(monoproc), 10

monoproc, locfit, numeric, missing, missing-method
(monoproc), 10

monoproc, locfit, numeric, numeric, missing-method
(monoproc), 10

monoproc, locpoly, numeric, missing, missing-method
(monoproc), 10

monoproc, locpoly, numeric, numeric, missing-method
(monoproc), 10

monoproc,loess,numeric,missing,missing-method
 (monoproc), 10

monoproc,loess,numeric,numeric,missing-method
 (monoproc), 10

monoproc-class, 12

monoproc.1d, 9, 11, 12, 14

monoproc.1d-class, 10

monoproc.1d-class, 13

monoproc.2d, 9, 11, 12, 14

monoproc.2d-class, 10

monoproc.2d-class, 14

monoproclocfit.1d-class
 (monoproc.1d-class), 13

monoproclocfit.2d-class
 (monoproc.2d-class), 14

plot,ksmooth,missing-method
 (ksmooth-class), 4

plot,locpoly,missing-method
 (locpoly-class), 5

plot,monofit,missing-method
 (monofit-class), 9

plot,monoproc.1d,missing-method
 (monoproc.1d-class), 13

plot,monoproc.2d,missing-method
 (monoproc.2d-class), 14

print,monoproc.1d-method
 (monoproc.1d-class), 13

print,monoproc.2d-method
 (monoproc.2d-class), 14

show,monoproc.1d-method
 (monoproc.1d-class), 13

show,monoproc.2d-method
 (monoproc.2d-class), 14

summary,monoproc.1d-method
 (monoproc.1d-class), 13

summary,monoproc.2d-method
 (monoproc.2d-class), 14