

Package ‘gamlss.util’

May 18, 2016

Description Extra functions for GAMLSS and others models.

Title GAMLSS Utilities

LazyLoad yes

Version 4.3-4

Date 2016-06-18

Depends R (>= 2.15.0), gamlss.dist, gamlss (>= 4.3.3), zoo, stats,
methods, graphics, grDevices

Suggests colorspace

Author Mikis Stasinopoulos
los <d.stasinopoulos@londonmet.ac.uk>, Bob Rigby <r.rigby@londonmet.ac.uk>, Paul Eilers
<p.eilers@erasmusmc.nl>

Maintainer Mikis Stasinopoulos <d.stasinopoulos@londonmet.ac.uk>

License GPL-2 | GPL-3

URL <http://www.gamlss.org/>

NeedsCompilation no

Repository CRAN

Date/Publication 2016-05-18 14:22:29

R topics documented:

| | |
|----------------------------|-----------|
| centiles.ts | 2 |
| gammaFit | 4 |
| lagPlot | 6 |
| penReg | 7 |
| plotSimpleGamlss | 10 |
| scattersmooth | 12 |
| Index | 15 |

centiles.ts

*Plots the centile curves for a time series GAMLSS object***Description**

This function `centiles.ts()` plots centiles curves for time series response variables whose distributions belong to the GAMLSS family of distributions. The function also tabulates the sample percentages below each centile curve (for comparison with the model percentages given by the argument `cent`.)

Usage

```
centiles.ts(obj, xvar = NULL, cent = c(0.5, 2.5, 50, 95.5, 99.5), legend = TRUE,
  ylab = "y", xlab = "x", main = NULL, main.gsub = "@",
  xleg = min(xvar), yleg = max(obj$y), xlim = range(xvar),
  ylim = range(obj$y), save = FALSE, plot = TRUE, type = "l",
  points = TRUE, pch = "+", col = "blue", col.centiles = 1:length(cent) + 2,
  lty.centiles = 1, lwd.centiles = 1, ...)
```

Arguments

| | |
|------------------------|--|
| <code>obj</code> | a fitted <code>gamlss</code> object which has a time series response variable |
| <code>xvar</code> | the time of the time series |
| <code>cent</code> | a vector with elements the % centile values for which the centile curves have to be evaluated |
| <code>legend</code> | whether a legend is required in the plot or not, the default is <code>legend=TRUE</code> |
| <code>ylab</code> | the y-variable label |
| <code>xlab</code> | the x-variable label |
| <code>main</code> | the main title here as character. If <code>NULL</code> the default title "centile curves using NO" (or the relevant distributions name) is shown |
| <code>main.gsub</code> | if the <code>main.gsub</code> (with default "@") appears in the main title then it is substituted with the default title. |
| <code>xleg</code> | position of the legend in the x-axis |
| <code>yleg</code> | position of the legend in the y-axis |
| <code>xlim</code> | the limits of the x-axis |
| <code>ylim</code> | the limits of the y-axis |
| <code>save</code> | whether to save the sample percentages or not with default equal to <code>FALSE</code> . In this case the sample percentages are printed but are not saved |
| <code>plot</code> | whether to plot the centiles. This option is useful for <code>centile.split</code> |
| <code>type</code> | type of line |
| <code>pch</code> | the character to be used as the default in plotting points see <code>par</code> |
| <code>col</code> | plotting colour see <code>par</code> |

| | |
|--------------|--|
| col.centiles | Plotting colours for the centile curves |
| lty.centiles | line type for the centile curves |
| lwd.centiles | The line width for the centile curves |
| points | whether the data points should be plotted, default is TRUE for centiles() and FALSE for centiles.fan() |
| ... | for extra arguments |

Details

Centiles are calculated using the fitted values in obj and xvar must correspond exactly to the time of the response time series object

Value

A centile plot is produced and the sample centiles below each centile curve are printed (or saved)

Author(s)

Mikis Stasinopoulos <d.stasinopoulos@londonmet.ac.uk>, Bob Rigby <r.rigby@londonmet.ac.uk> with contribution from Majid Djennad

References

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.

See Also

[centiles](#)

Examples

```
## Not run:
library(gamlss.add)
dax <- EuStockMarkets[, "DAX"]
# returns
rdax <- diff(dax,1)
w1 <- wlag(rdax,30)
# garch type
f1<- gamlss(rdax~ la(rdax, lags=30, from.lag=1), sigma.fo~la(rdax^2,
  lags=30, from.lag=1), weights=w1, bf.cyc=10, family=TF)
tiR <- as.numeric(time(rdax))
```

```
centiles.ts(f1, xvar=tiR, cent=c(2.5,50,97.5), col.cent="black")

## End(Not run)
```

gammaFit

A function to fit a GARMA model

Description

This function is for fitting a GARMA model, see Benjamin et al. (2003).

Usage

```
gammaFit(formula = formula(data), order = c(0, 0),
         weights = NULL, data = sys.parent(),
         family = NO(), alpha = 0.1,
         phi.start = NULL, theta.start = NULL,
         tail = max(order), control = list())
```

Arguments

| | |
|-------------|---|
| formula | A formula for linear terms i.e. like in <code>lm()</code> |
| order | order specify the order of the generalised arm model |
| weights | prior weighs, they are working like in <code>gamlss</code> |
| data | the relevant data. frame |
| family | A <code>gamlss.family</code> distribution |
| alpha | This parameter is used in the definition of the link function of the response variable i.e. $\log(y^*)$ will be $y^* = \max(y, \alpha)$ |
| phi.start | starting values for the AR parameters |
| theta.start | starting values for the MA part |
| tail | how many observation from the tall of the response variable should be suppressed |
| control | control for <code>optim()</code> or <code>nlmnb()</code> function use for optimisation. |

Details

The model is described in Benjamin et al. (2003). The implementation here is more general that it allows all the `gamlss.family` distributions to be fitted rather than only for the exponential family which was described in the original paper. Note that in this formulation only the mu can be modelled as ARMA.

Value

It returns a fitted gamma model.

Note

There is no check done whether the fitted model is stationary.

Author(s)

Mikis Stasinopoulos <d.stasinopoulos@londonmet.ac.uk>, Bob Rigby <r.rigby@londonmet.ac.uk> and Vlasios Voudouris

References

Benjamin M. A., Rigby R. A. and Stasinopoulos D.M. (2003) Generalised Autoregressive Moving Average Models. *J. Am. Statist. Ass.*, 98, 214-223.

Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.

Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).

Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.

See Also

[gamlss.family](#), [gamlss](#)

Examples

```
data(polio)
ti <- as.numeric(time(polio))
mo <- as.factor(cycle(polio))
x1 <- 0:167 #Index used in Tutz p197
x2 <- cos(2*pi*x1/12)
x3 <- sin(2*pi*x1/12)
x4 <- cos(2*pi*x1/6)
x5 <- sin(2*pi*x1/6)
# all the data here
da <-data.frame(polio,x1,x2,x3,x4,x5, ti, mo)
rm(ti,mo,x1,x2,x3,x4,x5)

#-----
# with linear trend
m00 <- gammaFit(polio~x1+x2+x3+x4+x5, data=da, order=c(0,0), family=NBI, tail=3) #
m10 <- gammaFit(polio~x1+x2+x3+x4+x5, data=da, order=c(1,0), family=NBI, tail=3) #

## Not run:
m01 <- gammaFit(polio~x1+x2+x3+x4+x5, order=c(0,1), data=da, family=NBI, tail=3)
m20 <- gammaFit(polio~x1+x2+x3+x4+x5, order=c(2,0), data=da, family=NBI, tail=3)
m11 <- gammaFit(polio~x1+x2+x3+x4+x5, order=c(1,1), data=da, family=NBI, tail=3)
m02 <- gammaFit(polio~x1+x2+x3+x4+x5, order=c(0,2), data=da, family=NBI, tail=3)
m30 <- gammaFit(polio~x1+x2+x3+x4+x5, order=c(3,0), data=da, family=NBI, tail=3)
```

```

m21 <- garmaFit(polio~x1+x2+x3+x4+x5, order=c(2,1), data=da, family=NBI, tail=3)
m12 <- garmaFit(polio~x1+x2+x3+x4+x5, order=c(1,2), data=da, family=NBI, tail=3)
m03 <- garmaFit(polio~x1+x2+x3+x4+x5, order=c(0,3), data=da, family=NBI, tail=3)
AIC(m00,m10,m01,m20,m11,m02,m30,m21,m12,m03 , k=0)
AIC(m00,m10,m01,m20,m11,m02,m30,m21,m12,m03 , k=log(168))
# without linear trend
n00 <- garmaFit(polio~x2+x3+x4+x5, data=da, order=c(0,0), family=NBI, tail=3) #
n10 <- garmaFit(polio~x2+x3+x4+x5, data=da, order=c(1,0), family=NBI, tail=3) # OK
n01 <- garmaFit(polio~x2+x3+x4+x5, order=c(0,1), data=da, family=NBI, tail=3)
n20 <- garmaFit(polio~x2+x3+x4+x5, order=c(2,0), data=da, family=NBI, tail=3)
n11 <- garmaFit(polio~x2+x3+x4+x5, order=c(1,1), data=da, family=NBI, tail=3)
n02 <- garmaFit(polio~x2+x3+x4+x5, order=c(0,2), data=da, family=NBI, tail=3)
n30 <- garmaFit(polio~x2+x3+x4+x5, order=c(3,0), data=da, family=NBI, tail=3)
n21 <- garmaFit(polio~x2+x3+x4+x5, order=c(2,1), data=da, family=NBI, tail=3)
n12 <- garmaFit(polio~x2+x3+x4+x5, order=c(1,2), data=da, family=NBI, tail=3)
n03 <- garmaFit(polio~x2+x3+x4+x5, order=c(0,3), data=da, family=NBI, tail=3)

AIC(m00,n10,n01,n20,n11,n02,n30,n21,n12, k=0)
AIC(m00,n10,n01,n20,n11,n02,n30,n21,n12, k=log(168))

## End(Not run)

```

lagPlot

Lag plot for time series data

Description

The function `lagPlot()` plots a time series variable against its lagged values or against the lagged values of an explanatory variable.

Usage

```
lagPlot(y, x = NULL, lags = 0, corr = TRUE, smooth = TRUE)
```

Arguments

| | |
|---------------------|--|
| <code>y</code> | time-series (univariate) |
| <code>x</code> | explanatory variable |
| <code>lags</code> | number of lag plots desired |
| <code>corr</code> | whether to include the correlation in the plot |
| <code>smooth</code> | whether to plot the smoothing curve |

Details

The function uses the functions `lag.plo1()` and `lag.plo2()` described in Shumway and Stoffer (2011) page 56.

Value

A plot is produced.

Author(s)

Mikis Stasinopoulos

References

Shumway R. H. and Stoffer D. S. (2011) *Time Series Analysis and Its Applications, With R Examples*. (third edition), Springer, New York, .

See Also

[lag.plot](#)

Examples

```
dax<-EuStockMarkets[,"DAX"]
ftse<-EuStockMarkets[,"FTSE"]
lagPlot(dax, lags=9)
lagPlot(dax, ftse, lags=8)
```

 penReg

Function to fit penalised regression

Description

The function `penReg()` can be used to fit a P-spline. It can be used as demonstration of how the penalised B-splines can be fitted to one explanatory variable. For more than one explanatory variables use the function `pb()` in **gamlss**. The function `penRegQ()` is similar to the function `penReg()` but it estimates the "random effect" sigmas using the Q-function (marginal likelihood). The Q-function estimation takes longer but it has the advantage that standard errors are provided for $\log(\sigma_e)$ and $\log(\sigma_b)$, where the sigmas are the standard errors for the response and the random effects respectively. The function `pbq()` is a smoother within GAMLSS and should give identical results to the additive function `pb()`. The function `gamlss.pbq` is not for use.

Usage

```
penReg(y, x, weights = rep(1, length(y)), df = NULL, lambda = NULL, start = 10,
       inter = 20, order = 2, degree = 3, plot = FALSE,
       method = c("ML", "ML-1", "GAIC", "GCV", "EM"), k = 2, ...)
penRegQ(y, x, weights = rep(1, length(y)), order = 2, start = 10,
        plot = FALSE, lambda = NULL, inter = 20, degree = 3,
        optim.proc = c("nlminb", "optim"),
        optim.control = NULL)
pbq(x, control = pbq.control(...), ...)
gamlss.pbq(x, y, w, xeval = NULL, ...)
```

Arguments

| | |
|---------------|---|
| y | the response variable |
| x | the unique explanatory variable |
| weights | prior weights |
| w | weights in the iteration withing GAMLSS |
| df | effective degrees of freedom |
| lambda | the smoothing parameter |
| start | the lambda starting value if the local methods are used |
| inter | the no of break points (knots) in the x-axis |
| order | the required difference in the vector of coefficients |
| degree | the degree of the piecewise polynomial |
| plot | whether to plot the data and the fitted function |
| method | The method used in the (local) performance iterations. Available methods are "ML", "ML-1", "EM", "GAIC" and "GCV" |
| k | the penalty used in "GAIC" and "GCV" |
| optim.proc | which function to be use to optimise the Q-function, options are c("nlminb", "optim") |
| optim.control | options for the optimisation procedures |
| control | arguments for the fitting function. It takes one two: i) order the order of the B-spline and plot whether to plot the data and fit. |
| xeval | this is use for prediction |
| ... | for extra arguments |

Value

Returns a fitted object of class penReg. The object contains 1) the fitted coefficients 2) the fitted.values 3) the response variable y, 4) the label of the response variable ylabel 5) the explanatory variable x, 6) the label of the explanatory variable 7) the smoothing parameter lambda, 8) the effective degrees of freedom df, 9) the estimate for sigma sigma, 10) the residual sum of squares rss, 11) the Akaike information criterion aic, 12) the Bayesian information criterion sbc and 13) the deviance

Author(s)

Mikis Stasinopoulos <d.stasinopoulos@londonmet.ac.uk>, Bob Rigby <r.rigby@londonmet.ac.uk> and Paul Eilers

References

- Eilers, P. H. C. and Marx, B. D. (1996). Flexible smoothing with B-splines and penalties (with comments and rejoinder). *Statist. Sci*, **11**, 89-121.
- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.

Examples

```

set.seed(1234)
x <- seq(0,10,length=200); y<-(yt<-1+2*x+.6*x^2-.1*x^3)+rnorm(200, 4)
library(gamlss)
#-----
# df fixed
g1<-gamlss(y~pb(x, df=4))
m1<-penReg(y,x, df=4)
cbind(g1$mu.coefSmo[[1]]$lambda, m1$lambda)
cbind(g1$mu.df, m1$edf)
cbind(g1$aic, m1$aic)
cbind(fitted(g1), fitted(m1))[1:10,]
# identical
#-----
# estimate lambda using ML
g2<-gamlss(y~pb(x))
m2<-penReg(y,x)
cbind(g2$mu.df, m2$edf)
cbind(g2$mu.lambda, m2$lambda)
cbind(g2$aic, m2$aic) # different lambda
cbind(fitted(g2), fitted(m2))[1:10,]
# identical
#-----
# estimate lambda using GCV
g3 <- gamlss(y~pb(x, method="GCV"))
m3 <- penReg(y,x, method="GCV")
cbind(g3$mu.df, m3$edf)
cbind(g3$mu.lambda, m3$lambda)
cbind(g3$aic, m3$aic)
cbind(fitted(g3), fitted(m3))[1:10,]
# almost identical
#-----
# estimate lambda using GAIC(#=3)
g4<-gamlss(y~pb(x, method="GAIC", k=3))
m4<-penReg(y,x, method="GAIC", k=3)
cbind(g4$mu.df, m4$edf )
cbind(g4$mu.lambda, m4$lambda)
cbind(g4$aic, m4$aic)
cbind(g4$mu.df, m4$df)
cbind(g4$mu.lambda, m4$lambda)
cbind(fitted(g4), fitted(m4))[1:10,]

#-----
plot(y~x)
lines(fitted(m1)~x, col="green")
lines(fitted(m2)~x, col="red")
lines(fitted(m3)~x, col="blue")
lines(fitted(m4)~x, col="yellow")
lines(fitted(m4)~x, col="grey")
# using the Q function

# the Q-function takes longer

```

```

system.time(g6<-gamlss(y~pbq(x)))
system.time(g61<-gamlss(y~pb(x)))
AIC(g6, g61)
#
system.time(m6<-penRegQ(y,x))
system.time(m61<-penReg(y,x))
AIC(m6, m61)

cbind(g6$mu.df, g61$mu.df,m6$se.df, m61$se.df)
cbind(g6$mu.lambda,g61$mu.lambda, m6$lambda, m61$lambda)
cbind(g6$aic, AIC(g6), m6$aic, AIC(m6), m61$aic, AIC(m61))
cbind(fitted(g6), fitted(m6))[1:10,]

```

plotSimpleGamlss

Plotting a simple GAMLSS model for demonstration purpose

Description

This is to plot a simple GAMLSS model where only one explanatory variable exist in order to demonstrated how the distribution of the response changes according to values of the explanatory variable.

Usage

```

plotSimpleGamlss(y, x, model = NULL, formula = NULL, data = NULL,
  family = NULL, val = NULL, N = 1000, x.val = quantile(x),
  ylim = c(min(y), max(y)), xlim = c(min(x), max(x)), ...)

```

Arguments

| | |
|---------|---|
| y | The response variable |
| x | The explanatory variable (only one is allowed here) |
| model | A fitted gamlss model |
| formula | A formula for the mean model if model=NULL |
| data | The data where the response and the one explanatory can be found |
| family | The gamlss family distribution |
| val | this parameter determines how the plotted distribution is shown, increase/decrease it if the distribution is not shown properly |
| N | This parameters determine how many values of y are generated for each x. var |
| x.val | the values of the explanatory variable where we want to see the distribution |
| ylim | the y limits in the plot |
| xlim | the x limits in the plot |
| ... | extra argument to be passed to gamlss() function if model=NULL |

Details

This function is for pedagogical purpose rather than fitting models to demonstrate that the distribution of the response variable can vary according to explanatory variables. In its current form it can be used with continuous and discrete responses only.

Value

A plot is shown

Author(s)

Mikis Stasinopoulos <d.stasinopoulos@londonmet.ac.uk>

References

Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.

Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.

See Also

[scattersmooth](#)

Examples

```
## the abdominal data
m1 <- gamlss(y~pb(x), sigma.fo=~pb(x), data=abdom, family=L0)
plotSimpleGamlss(y,x, model=m1, data=abdom, x.val=seq(15, 40, 5),
                 ylim=c(0, 450), xlim=c(5, 45))

data(species)
species$ll <- log(species$lake)
m2 <- gamlss(fish~ll, data=species, trace=FALSE, family=P0 )
plotSimpleGamlss(fish,ll, model=m2, data=species, x.val=c(3,5,7, 9),
                 val=20, N=100, ylim=c(0,80))

m3 <- gamlss(fish~ll, data=species, trace=FALSE, family=NBI, sigma.fo=~ll )
plotSimpleGamlss(fish,ll, model=m3, data=species, x.val=c(3,5,7, 9),
                 val=20, N=100, ylim=c(0,100))

## Not run:
##-----
## the rent data
## fitting the model first
r1 <- gamlss(R~pb(F1), sigma.fo=~pb(F1),data=rent, family=GA, ylim=c(0, 3000))
## plot 1
plotSimpleGamlss(R,F1, model=r1, data=rent, x.val=seq(40,120, 5))
## plot 2 finer grid
plotSimpleGamlss(R,F1, model=r1, data=rent, x.val=seq(40,120, 1),
                 xlim=c(10,120))
## the same but fitting the model within the function
## note that sigma formula has to be specified
```

```
plotSimpleGam1ss(R,F1, formula= R~pb(F1), family=GA, data=rent,
                 x.val=seq(40,120, 5), sigma.fo=~pb(F1))
#-----
## End(Not run)
```

scattersmooth

Two dimensional Smooth scatter plots

Description

The function produced two dimensional smooth scatter plots. The method used is described in Eilers and Goeman (2004).

Usage

```
scattersmooth(x, y, nbin = 100, lambda = 1, ndot = 500,
              csize = 0.3, ticks = TRUE, xlim = c(min(x),
              max(x)), ylim = c(min(y), max(y)), show = TRUE,
              save = FALSE, data = NULL, xlab = NULL,
              ylab = NULL, cols = heat.colors(10:200),
              col.points = "blue", ...)
```

Arguments

| | |
|------------|--|
| x | the x-variable |
| y | the y-variable |
| nbin | the number of bins required for smoothing |
| lambda | the smoothing parameter |
| ndot | how many data points to show in the plot |
| csize | the size of the data points |
| ticks | whether ticks in the x and y axis appear in the plot |
| xlim | the x limit |
| ylim | the y limit |
| show | whether to show the graph or not |
| save | whether to save the output as a list or not |
| data | the data file data |
| xlab | the x label as character string |
| ylab | the y label as character string |
| cols | for changing the color scheme, the default is <code>heat.colors(10:200)</code> . Other suggestions are <code>gray(0:100/100)</code> , <code>heat.colors(101)</code> , <code>rainbow(100:200)</code> , <code>terrain.colors(101)</code> , <code>topo.colors(101)</code> , <code>cm.colors(101)</code> . Note that if you have the package colorspace in R you can use <code>heat_hcl(100)</code> which was the default before. |
| col.points | the colours of the points |
| ... | for extra arguments |

Details

The function is similar to the function `smoothScatter()` in **graphics** but it used penelized bin smoother as described in Eilers and Goeman (2004) rather than kernel smoother.

Value

the function produces a two dimensional smooth plot and saves if `save=TRUE` a list with the following components:

| | |
|----------------------|---|
| <code>Hraw</code> | A <code>nbib</code> by <code>nbib</code> matrix containing the bin row data |
| <code>Hsmooth</code> | A <code>nbib</code> by <code>nbib</code> matrix containing the smooth two dimensional histogram |
| <code>xgrid</code> | the x-grid |
| <code>ygrid</code> | the y-grid |
| <code>xbin</code> | the bin for x values |
| <code>ybin</code> | the bin for y values |
| <code>nmiss</code> | number of missing values |
| <code>seldots</code> | the values of the plotted dots |

Author(s)

Paul Eilers <p.eilers@erasmusmc.nl>

References

- Eilers, P. H. C. and Goeman, J. J. (2004). Enhancing scatterplots with smoothed density. *Bioinformatics*, Vol **20** no 5, pp 623-628.
- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.

See Also

[smoothScatter,gamlss](#)

Examples

```
m <- 1000
set.seed(pi)
phi <- 2 * pi * runif(m)
rho <- rchisq(m, df = 6)
x <- cos(phi) * rho
y <- sin(phi) * rho
H <- scattersmooth(x, y)
H1 <- scattersmooth(x, y, cols=rainbow(100:200))
# If you have the package colorspace use instead
```

```
# library(colorspace)
# H <- scattersmooth(x, y, cols=heat_hcl(100))
# H1 <- scattersmooth(x, y, cols=rainbow_hcl(100))
data(db)
scattersmooth(age, head, data=db, cols=terrain.colors(101), ndot=2000, lambda=1)
# or if you have colorspace
#scattersmooth(age, head, data=db, cols=terrain_hcl(100), ndot=2000, lambda=1)
```

Index

*Topic **models**

gammaFit, [4](#)

*Topic **regression**

centiles.ts, [2](#)

gammaFit, [4](#)

penReg, [7](#)

plotSimpleGamlss, [10](#)

scattersmooth, [12](#)

*Topic **ts**

lagPlot, [6](#)

centiles, [3](#)

centiles.ts, [2](#)

gamlss, [5](#), [13](#)

gamlss.family, [5](#)

gamlss.pbq (penReg), [7](#)

gammaFit, [4](#)

lag.plot, [7](#)

lagPlot, [6](#)

pbq (penReg), [7](#)

penReg, [7](#)

penRegQ (penReg), [7](#)

plotSimpleGamlss, [10](#)

scattersmooth, [11](#), [12](#)

smoothScatter, [13](#)