

Package ‘gamlss.add’

February 21, 2012

Description Extra additive terms for GAMLSS models.

Title GAMLSS Additive.

LazyLoad yes

Version 4.1-0

Date 2012-02-21

Depends R (>= 2.4.0), gamlss, mgcv, nnet, gamlss.util, gamlss.nl,lattice

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

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

License GPL-2 | GPL-3

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

Repository CRAN

Date/Publication 2012-02-21 10:40:47

R topics documented:

fk	2
ga	4
gamlss.fk	6
gamlss.ga	7
gamlss.nn	8
nn	9
Index	13

`fk`*A function to fit break points within GAMLSS*

Description

The `fk()` function is an additive function to be used for GAMLSS models. It is an interface for the `fitFreeKnots()` function of package **`gamlss.util`**. The functions `fitFreeKnots()` was first based on the `curfit.free.knot()` function of package `DierckxSpline` of Sundar Dorai-Raj and Spencer Graves. The function `fk()` allows the user to use the free knots function `fitFreeKnots()` within `gamlss`. The great advantage of course comes from the fact GAMLSS models provide a variety of distributions and diagnostics.

Usage

```
fk(x, degree = 3, start = NULL, ...)
```

Arguments

<code>x</code>	the x-variable
<code>degree</code>	the degree of the spline function fitted
<code>start</code>	starting values for the breakpoints. If are set the number of break points is also determined by the length of <code>start</code>
<code>...</code>	for extra arguments

Details

Note that `fk` itself does no smoothing; it simply sets things up for the function `gamlss()` which in turn uses the function `additive.fit()` for backfitting which in turn uses `gamlss.fk()`. Note that, finding the break points is not a trivial problem and therefor multiple maximum points can occur. More details about the free knot splines can be found in Dierckx, (1991).

The `gamlss` algorithm used a modified backfitting in this case, that is, it fits the linear part first. Note that trying to predict outside the x-range can be dangerous as the example below shows.

Value

The `gamlss` object saved contains the last fitted object which can be accessed using `obj$par.coefSmo` where `obj` is the fitted `gamlss` object `par` is the relevant distribution parameter.

Author(s)

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

References

- Dierckx, P. (1991) *Curve and Surface Fitting with Splines*, Oxford Science Publications
- 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.fk](#)

Examples

```
library(gamlss.util)
# creating a linear + linear function
x <- seq(0,10, length.out=201)
knot <- 5
set.seed(12543)
mu <- ifelse(x<=knot,5+0.5*x,5+0.5*x+(x-knot))
y <- rNO(201, mu=mu, sigma=.5)
# plot the data
plot(y~x, xlim=c(-1,13), ylim=c(3,17))
# fit model using curfit
m1 <- fitFreeKnots(x, y, knots=4, degree=1)
# fitted values
lines(fitted(m1)~x, col="red", lwd="3")
# predict
pm1<-predict(m1, newdata=-1:12)
points(-1:12,pm1, col="red",pch = 21, bg="blue")
#-----
# now gamlss
#-----
# get the fk function
# fit model
g0 <- gamlss(y~fk(x, degree=1, start=c(4)))
# creating data frame
da <- data.frame(y,x)
g1 <- gamlss(y~fk(x, degree=1, start=c(4)), data=da)
nd<-data.frame(x=-1:12)
pg1<-predict(g1, newdata=nd)
lines(fitted(g1)~x, col="purple", lwd="3")
points(pg1~I(-1:12), col="darkgreen", pch = 21, bg="green" )
#-----
#-----
# now negative binomial data
eta1 <- ifelse(x<=knot,5+0.5*x,5+0.5*x+(x-knot))
set.seed(143)
y <- rNBI(201, mu=exp(eta1/7), sigma=.1)
da <- data.frame(y=x)
```

```

#rm(y,x)
plot(y~x, data=da)
# fitting the model in gamlss using profile deviance
n1 <- quote(gamlss(y ~ x+I((x>this)*(x-this)),family=NBI,data=da))
prof.term(n1, min=1, max=9, step=.1, criterion="GD")
# now fit the model using fk
g1 <- gamlss(y~fk(x, degree=1, start=c(4)), data=da, family=NBI)
# get the breakpoint
knots(g1$mu.coefSmo[[1]])
# summary of the gamlss object FreeBreakPointsReg object
summary(g1)
# summary of the fitted
summary(g1$mu.coefSmo[[1]])
lines(fitted(g1)~x, data=da)
#-----
# the aids data
# using fk()
data(aids)
a1<-gamlss(y~fk(x, degree=1, start=25)+qrt, data=aids, family=NBI)
knots(a1$mu.coefSmo[[1]])
# using profile deviance
aids.1 <- quote(gamlss(y ~ x+I((x>this)*(x-this))+qrt,family=NBI,data=aids))
prof.term(aids.1, min=16, max=21, step=.1, criterion="GD")
# using non-linear estimation
library(gamlss.nl)
naids <- nl.obj(~p1*(x-p2)*I((x>p2)), start=c(0.2, 20), data=aids)
mod1 <- gamlss(y~nl(naids)+x+qrt, data=aids , family=NBI)
mod1$mu.coefSmo
GAIC(mod1,a1) # almost identical
# plotting the fit
with(aids, plot(x,y,pch=21,bg=c("red","green3","blue","yellow")[unclass(qrt)]))
lines(fitted(a1)~aids$x)
#-----

```

ga

A interface function to use Simon Wood's gam() function within GAMLSS

Description

The `ga()` function is a additive function to be used for GAMLSS models. It is an interface for the `gam()` function of package `mgcv` of Simon Wood. The function `ga()` allows the user to use all of the available smoothers of `gam()` within `gamlss`. The great advantage of course come from fitting models outside the exponential family.

Usage

```
ga(formula, ...)
```

Arguments

formula A formula containing $s()$ and te functions i.e. $\sim s(x1) + te(x2, x3)$.
 ... arguments used by the `gam()` function.

Details

Note that `ga` itself does no smoothing; it simply sets things up for the function `gamlss()` which in turn uses the function `additive.fit()` for backfitting which in turn uses `gamlss.ga()`

Note that, in our (limited) experience, for normal errors or exponential family, the fitted models using `gam()` and `ga()` within `gamlss()` are identical or at least very similar. This is particularly true if the default values for `gam()` are used.

Value

the fitted values of the smoother is returned, endowed with a number of attributes. The smoother fitted values are used in the construction of the overall fitted values of the particular distribution parameter. The attributes can be use to obtain information about the individual fit. In particular the `coefSmo` within the parameters of the fitted model contains the final additive fit.

Warning

The function is experimental so please report any peculiar behaviour to the authors

Author(s)

Mikis Stasinopoulos

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>.

Wood S.N. (2006) *Generalized Additive Models: An Introduction with R*. Chapman and Hall/CRC Press.

Examples

```
library(gamlss)
library(mgcv)
data(rent)
#-----
# normal errors one x-variable
ga1 <- gam(R~s(F1, bs="ps", k=20), data=rent, method="ML")
```

```

gn1 <- gamlss(R~pb(Fl), data=rent) # additive
gn2 <- gamlss(R~ga(~s(Fl)), data=rent) # additive
AIC(ga1,gn1,gn2, k=0)
#-----
# normal error additive in Fl and A
# normal error additive in Fl and A
ga2 <- gam(R~s(Fl)+s(A), data=rent)
gn0 <- gamlss(R~pb(Fl)+pb(A), data=rent) # additive
gn1 <- gamlss(R~ga(~s(Fl)+s(A)), data=rent) # additive
gn2 <- gamlss(R~ga(~s(Fl))+ga(~s(A)), data=rent)
# similar fitting
AIC(ga2,gn0,gn1, gn2, k=0)
#-----
# gamma errors one x-var
ga1 <- gam(R~s(Fl), data=rent, family=Gamma)
gg1 <- gamlss(R~pb(Fl), data=rent, family=GA)
gg2 <- gamlss(R~ga(~s(Fl)), data=rent, family=GA)
AIC(ga1, gg1, gg2, k=0)
# different degrees of freedom for mu in ga1
#-----
# gamma error two variables s() function
g22 <-gam(R~s(Fl,A), data=rent, family=Gamma)
gm22 <- gamlss(R~ga(~s(Fl,A)), data=rent, family=GA)
AIC(g22,gm22)
# predict
newrent <- data.frame(expand.grid(Fl=seq(30,120,5), A=seq(1890,1990,5 )))
newrent$pred2 <- predict(gm22, newdata=newrent, type="response")
newrent$pred1 <- predict(g22, newdata=newrent, type="response")
library(lattice)
wf1<-wireframe(pred1~Fl*A, newrent, aspect=c(1,0.5), drape=TRUE, colorkey=(list(space="right", height=0.6)), mai
wf2<-wireframe(pred2~Fl*A, newrent, aspect=c(1,0.5), drape=TRUE, colorkey=(list(space="right", height=0.6)), mai
print(wf1, split=c(1,1,2,1), more=TRUE)
print(wf2, split=c(2,1,2,1))
#-----
#gamma error two variables te() function
g221 <-gam(R~te(Fl,A), data=rent, family=Gamma)
gm221 <- gamlss(R~ga(~te(Fl,A)), data=rent, family=GA)
AIC(g221,gm221)
# predict
newrent <- data.frame(expand.grid(Fl=seq(30,120,5), A=seq(1890,1990,5 )))
newrent$pred2 <- predict(gm221, newdata=newrent, type="response")
newrent$pred1 <- predict(g221, newdata=newrent, type="response")
library(lattice)
wf1<-wireframe(pred1~Fl*A, newrent, aspect=c(1,0.5), drape=TRUE, colorkey=(list(space="right", height=0.6)), mai
wf2<-wireframe(pred2~Fl*A, newrent, aspect=c(1,0.5), drape=TRUE, colorkey=(list(space="right", height=0.6)), mai
print(wf1, split=c(1,1,2,1), more=TRUE)
print(wf2, split=c(2,1,2,1))
#-----

```

Description

This is support for the functions `fk()`. It is not intended to be called directly by users. The function `gamlss.fk` is calling on the R function `curfit.free.knot()` of Sundar Dorai-Raj

Usage

```
gamlss.fk(x, y, w, xeval = NULL, ...)
```

Arguments

<code>x</code>	the design matrix
<code>y</code>	the response variable
<code>w</code>	prior weights
<code>xeval</code>	used in prediction
<code>...</code>	for extra arguments

Author(s)

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

References

Dierckx, P. (1991) *Curve and Surface Fitting with Splines*, Oxford Science Publications

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

[fk](#)

gamlss.ga

Support for Function ga()

Description

This is support for the smoother function `ga()` an interface for Simon Wood's `gam()` function. It is not intended to be called directly by users.

Usage

```
gamlss.ga(x, y, w, xeval = NULL, ...)
```

Arguments

x	the explanatory variables
y	iterative y variable
w	iterative weights
xeval	if xeval=TRUE then predicion is used
...	for extra arguments

Author(s)

Mikis Stasinopoulos <d.stasinopoulos@londonmet.ac.uk>, Bob Rigby <r.rigby@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. 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>.
- Wood S.N. (2006) *Generalized Additive Models: An Introduction with R*. Chapman and Hall/CRC Press.

gamlss.nn

Support for Function nn()

Description

This is support for the smoother function ga() an inteface for Simon Wood's gam() function. It is not intended to be called directly by users.

Usage

```
gamlss.nn(x, y, w, xeval = NULL, ...)
```

Arguments

x	the explanatory variables
y	iterative y variable
w	iterative weights
xeval	if xeval=TRUE then predicion is used
...	for extra arguments

Author(s)

Mikis Stasinopoulos <d.stasinopoulos@londonmet.ac.uk>, Bob Rigby <r.rigby@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.
- Ripley, B. D. (1996) Pattern Recognition and Neural Networks. Cambridge.
- 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>.
- Venables, W. N. and Ripley, B. D. (2002) Modern Applied Statistics with S. Fourth edition. Springer.

See Also

[ga](#)

 nn

A interface function to use nnet() function within GAMLSS

Description

The nn() function is a additive function to be used for GAMLSS models. It is an interface for the nnet() function of package nnet of Brian Ripley. The function nn() allows the user to use neural networks within gamlss. The great advantage of course comes from the fact GAMLSS models provide a variety of distributions and diagnostics.

Usage

```
nn(formula, control = nn.control(...), ...)
nn.control(size = 3, linout = TRUE, entropy = FALSE, softmax = FALSE,
           censored = FALSE, skip = FALSE, rang = 0.7, decay = 0,
           maxit = 100, Hess = FALSE, trace = FALSE,
           MaxNWts = 1000, abstol = 1e-04, reltol = 1e-08)
```

Arguments

formula	A formula containing the expolanatory variables i.e. $\sim x_1+x_2+x_3$.
control	control to pass the arguments for the nnet() function
...	for extra arguments
size	number of units in the hidden layer. Can be zero if there are skip-layer units

linout	switch for linear output units. Default is TRUE, identily link
entropy	switch for entropy (= maximum conditional likelihood) fitting. Default by least-squares.
softmax	switch for softmax (log-linear model) and maximum conditional likelihood fitting. linout, entropy, softmax and censored are mutually exclusive.
censored	A variant on softmax, in which non-zero targets mean possible classes. Thus for softmax a row of (0, 1, 1) means one example each of classes 2 and 3, but for censored it means one example whose class is only known to be 2 or 3.
skip	switch to add skip-layer connections from input to output
rang	Initial random weights on $[-rang, rang]$. Value about 0.5 unless the inputs are large, in which case it should be chosen so that $rang * \max(x)$ is about 1
decay	parameter for weight decay. Default 0.
maxit	parameter for weight decay. Default 0.
Hess	If true, the Hessian of the measure of fit at the best set of weights found is returned as component Hessian.
trace	switch for tracing optimization. Default FALSE
MaxNWts	The maximum allowable number of weights. There is no intrinsic limit in the code, but increasing MaxNWts will probably allow fits that are very slow and time-consuming.
abstol	Stop if the fit criterion falls below abstol, indicating an essentially perfect fit.
reltol	Stop if the optimizer is unable to reduce the fit criterion by a factor of at least 1 - reltol.

Details

Note that, neural networks are over parameterized models and therefor notorious for multiple maximum. There is no guarantee that two identical fits will produce identical results.

Value

Note that nn itself does no smoothing; it simply sets things up for the function `gamlss()` which in turn uses the function `additive.fit()` for backfitting which in turn uses `gamlss.nn()`

Warning

You may have to fit the model several time to unsure that you obtain a reasonable minimum

Author(s)

Mikis Stasinopoulos <d.stasinopoulos@londonmet.ac.uk>, Bob Rigby <r.rigby@londonmet.ac.uk> based on work of Venables & Ripley wich also based on work by Kurt Hornik and Albrecht Gebhardt.

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.
- Ripley, B. D. (1996) Pattern Recognition and Neural Networks. Cambridge.
- 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>.
- Venables, W. N. and Ripley, B. D. (2002) Modern Applied Statistics with S. Fourth edition. Springer.

Examples

```
library(nnet)
data(rock)
area1<- with(rock,area/10000)
peri1<- with (rock,peri/10000)
rock1<- with(rock, data.frame(perm, area=area1, peri=peri1, shape))
# fit nnet
r1 <- nnet(log(perm)~area+peri+shape, rock1, size=3, decay=1e-3, linout=TRUE, skip=TRUE, max=1000, Hess=TRUE)
summary(r1)
# get gamlss
library(gamlss)
cc <- nn.control(size=3, decay=1e-3, linout=TRUE, skip=TRUE, max=1000, Hess=TRUE)
g1 <- gamlss(log(perm)~nn(~area+peri+shape,size=3, control=cc), data=rock1)
summary(g1$mu.coefSmo[[1]])
# predict
Xp <- expand.grid(area=seq(0.1,1.2,0.05), peri=seq(0,0.5, 0.02), shape=0.2)
rocknew <- cbind(Xp, fit=predict(r1, newdata=Xp))
library(lattice)
wf1<-wireframe(fit~area+peri, rocknew, screen=list(z=160, x=-60), aspect=c(1, 0.5), drape=TRUE, main="nnet()")
rocknew1 <- cbind(Xp, fit=predict(g1, newdata=Xp))
wf2<-wireframe(fit~area+peri, rocknew1, screen=list(z=160, x=-60), aspect=c(1, 0.5), drape=TRUE, main="nn()")
print(wf1, split=c(1,1,2,1), more=TRUE)
print(wf2, split=c(2,1,2,1))
#-----
data(rent)
mr1 <- gamlss(R~nn(~F1+A, size=5, decay=0.001), data=rent, family=GA)
library(gamlss.add)
mg1<-gamlss(R~ga(~s(F1,A)), data=rent, family=GA)
AIC(mr1,mg1)
newrent <- newrent1 <-newrent2 <- data.frame(expand.grid(F1=seq(30,120,5), A=seq(1890,1990,5 )))
newrent1$fit <- predict(mr1, newdata=newrent, type="response") ##nn
newrent2$fit <- predict(mg1, newdata=newrent, type="response")# gam
library(lattice)
wf1<-wireframe(fit~F1+A, newrent1, aspect=c(1,0.5), drape=TRUE, colorkey=(list(space="right", height=0.6)), main="nn")
wf2<-wireframe(fit~F1+A, newrent2, aspect=c(1,0.5), drape=TRUE, colorkey=(list(space="right", height=0.6)), main="gam")
print(wf1, split=c(1,1,2,1), more=TRUE)
print(wf2, split=c(2,1,2,1))
```

```
#-----  
data(db)  
mdb1 <- gamlss(head~nn(~age,size=20, decay=0.001), data=db)  
plot(head~age, data=db)  
points(fitted(mdb1)~db$age, col="red")  
  
# do not run  
#mdb2 <- gamlss(head~nn(~age,size=20, decay=0.001), data=db, family=BCT)  
#plot(head~age, data=db)  
#points(fitted(mdb2)~db$age, col="red")
```

Index

*Topic **regression**

fk, [2](#)

ga, [4](#)

gamlss.fk, [7](#)

gamlss.ga, [7](#)

gamlss.nn, [8](#)

nn, [9](#)

fk, [2](#), [7](#)

ga, [4](#), [9](#)

gamlss.fk, [3](#), [6](#)

gamlss.ga, [7](#)

gamlss.nn, [8](#)

nn, [9](#)