Package: scar (via r-universe)

August 21, 2024

Version 0.2-2

Date 2022-05-25

Title Shape-Constrained Additive Regression: a Maximum Likelihood Approach

Author Yining Chen and Richard Samworth

Maintainer Yining Chen <y.chen101@lse.ac.uk>

Depends R (>= 3.0.0)

Suggests gam, mgcv, scam

Description Computes the maximum likelihood estimator of the generalised additive and index regression with shape constraints. Each additive component function is assumed to obey one of the nine possible shape restrictions: linear, increasing, decreasing, convex, convex increasing, convex decreasing, concave, concave increasing, or concave decreasing. For details, see Chen and Samworth (2016) <doi:10.1111/rssb.12137>.

License GPL (>= 2)

NeedsCompilation yes

Date/Publication 2022-05-25 22:50:02 UTC

Repository https://ychen101.r-universe.dev

RemoteUrl https://github.com/cran/scar

RemoteRef HEAD

RemoteSha dffb2dd867390cbe9f3935a77d475653578c1ef9

Contents

`scar-package` *Shape-Constrained Additive (index) Regression: a maximum likelihood approach*

Description

scar computes the maximum likelihood estimator of the generalised additive and index regression with shape constraints. Each component of the additive function of the predictors is assumed to belong to one of the nine possible shape restrictions: linear, increasing, decreasing, convex, convex and increasing, convex and decreasing, concave, concave and increasing, or concave and decreasing. For the generalised additive regression, the problem is transformed into a convex optimisation problem and the active set algorithm is used to find the optimum. We emphasise that unlike most of the other nonparametric methods, this approach is free of tuning parameters.

Furthermore, we can extend our findings to the generalised additive index regression, where a stochastic search algorithm is proposed to solve the problem.

Details

Package:	scar
Type:	Package
Version:	0.2-2
Date:	2022-05-25
License:	GPL(>=2)

This package contains a selection of functions for maximum likelihood estimation of the generalised additive (and additive index) regression under shape constraints:

`scar` computes the maximum likelihood estimator (specified via its value at the observed covariates). Output is a list of class `scar` which is used as input to various auxiliary functions.

`plot.scar` produces plots of the maximum likelihood estimator produced by `scar` on the scale of the additive predictors.

`predict.scar` obtains predictions either on the scale of the additive predictors or on the scale of the response variable from a fitted `scar` object.

`scair` tries to find the maximum likelihood estimator (specified via its value at the observed indices). Output is a list of class `scair` which is used as input to various auxiliary functions.

`plot.scair` produces plots of the maximum likelihood estimator produced by `scair` on the scale of the additive index predictors.

predict.scair obtains predictions either on the scale of the additive index predictors or on the scale of the response variable from a fitted scair object.

The methods proposed here were applied to the following datasets: PhDPublications, decathlon.

Note

The authors gratefully acknowledge the assistance of Ming Yuan for his insights into this problem.

Thanks also go to Mary Meyer for kindly providing the authors with her manuscript prior to its publication.

Author(s)

Yining Chen (maintainer) <y.chen101@lse.ac.uk>

Richard Samworth <r.samworth@statslab.cam.ac.uk>

References

Chen, Y. and Samworth, R. J. (2016). Generalized additive and index models with shape constraints. Journal of the Royal Statistical Society: Series B, 78, 729-754.

Groeneboom, P., Jongbloed, G. and Wellner, J.A. (2008). The support reduction algorithm for computing non-parametric function estimates in mixture models. Scandinavian Journal of Statistics, 35, 385-399.

Hastie, T. and Tibshirani, R. (1990) Generalized Additive Models. Chapman and Hall, London.

Meyer, M. C. (2013) Semi-parametric additive constrained regression. Journal of nonparametric statistics, 25, 715-743.

Nocedal, J., and Wright, S. J. (2006) Numerical Optimization, 2nd edition. Springer, New York.

Robertson, T., Wright, F. T. and Dykstra, R. L. (1988). Order Restricted Statistical Inference. Wiley, New York.

Venables, W. N. and Ripley, B. D. (2002) Modern Applied Statistics with S. Springer, New York.

Wood, S. N. (2004) Stable and efficient multiple smoothing parameter estimation for generalized additive models. Journal of American Statistical Association, 99, 673-686.

See Also

scar, scair, scam, glm

Examples

```
## See examples provided in functions scar and scair
```

`decathlon` *Men's decathlon athletes in 2012*

Description

This dataset consists of men's decathlon athletes who scored at least 6500 points in at least one athletic competition in 2012 and scored points in every event there. To avoid data dependency, we include only one performance from each athlete, namely their 2012 personal best (over the whole decathlon).

Usage

```
data("decathlon")
```

Format

A data frame containing 614 observations on 10 variables. Different rows represent results of different athletes.

X100m Points scored in 100 metres.

LJ Points scored in long jump.

SP Points scored in shot put.

HJ Points scored in high jump.

X400m Points scored in 400 metres.

X110H Points scored in 110 metres hurdles.

DT Points scored in discus throw.

PV Points scored in polt vault.

JT Points scored in javelin throw.

X1500m Points scored in 1500 metres.

The point system (on how to convert results to scores) can be found at

http://en.wikipedia.org/wiki/Decathlon

Source

Compiled from the dataset `decathlon_raw`.

Examples

```
data(decathlon)
dat = as.matrix(decathlon[,c("X100m","SP","DT")]); y = decathlon$JT

## We consider the problem of predicting a decathlete's javelin performance
## from their performances in the other decathlon disciplines.
## As an illustration, we only select 100m, shot put and discus throw
```

```
## also, we reduce the number of iterations here to shorten the run time
set.seed(1)
object = scair(dat,y,shape=c("in","in"),iter=40, allnonneg=TRUE)
object$index

## Here we can treat the obtained index matrix as a warm start and perform
## further optimization using e.g. the default R optimisation routine.
## The result is actually only slightly different from the warm start.
## The optimisation itself takes around 5-10 seconds.
## Code provided but commented out here because CRAN team requires all the
## examples running here to be completed in a few seconds.
# fn<-function(w){
#  wnew = matrix(0,nrow=3,ncol=2)
#  wnew[1,1] = w[1]
#  wnew[2,1] = w[2]
#  wnew[3,1] = 1 - abs(w[1]) - abs(w[2])
#  wnew[1,2] = w[3]
#  wnew[2,2] = w[4]
#  wnew[3,2] = 1 - abs(w[3]) - abs(w[4])
#  if ((wnew[3,1] < 0)|| (wnew[3,2] < 0)) {dev = Inf}
#  else if ((w[1] < 0) || (w[2] < 0) || (w[3] < 0) || (w[4] < 0)) {dev = Inf}
#  else {
#    datnew = dat %*% wnew
#    dev = scar(datnew,y,shape=c("in","in"))$deviance
#  }
#  return (dev)
# }
#
# index123 = optim(c(object$index[1:2,1],object$index[1:2,2]),fn)$par
# newindex = matrix(0,nrow=3,ncol=2)
# newindex[1:2,1] = index123[1:2]; newindex[1:2,2] = index123[3:4]
# newindex[3,1] = 1 - sum(abs(index123[1:2]))
# newindex[3,2] = 1 - sum(abs(index123[3:4]))
# newindex
```

`decathlon_raw` *Men's decathlon in 2012*

Description

This dataset consists of men's decathlon results of at least 6500 points in 2012. Only those with athletes scored points in every event are included. Most men's decathlons are divided into a two-day competition. First day events include 100 metres, long jump, shot put, high jump and 400 metres. Second day events include 110 metres hurdles, discus throw, pole vault, javelin throw and 1500 metres.

Usage

```
data("decathlon_raw")
```

Format

A data frame containing 1075 observations on 15 variables.

Total Total (decathlon) score.

Name First name of the athlete.

Surname Surname of the athlete.

X100m Result of 100 metres, in seconds.

LJ Result of long jump, in metres.

SP Result of shot put, in metres.

HJ Result of high jump, in metres.

X400m Result of 400 metres, in seconds.

X110H Result of 110 metres hurdles, in seconds.

DT Result of discus throw, in metres.

PV Result of polt vault, in metres.

JT Result of javelin throw, in metres.

X1500m Result of 1500 metres, in seconds.

First.day Total points scored in the first day.

Second.day Total points scored in the second day.

Source

Data compiled from the following sources:

www.decathlon2000.com/

www.iaaf.org/

www.ceskydesetiboj.wz.cz/statistika.html

See Also

`decathlon`

PhDPublications *Doctoral Publications*

Description

Data on the scientific productivity of PhD students in biochemistry.

Usage

```
data("PhDPublications")
```

Format

A data frame containing 915 observations on 6 variables:

articles Number of articles published during last 3 years of PhD.

gender Factor indicating gender.

married Factor indicating the marital status.

kids Number of children less than 6 years old.

prestige Prestige of the graduate program.

mentor Number of articles published by student's mentor.

Source

Imported directly from CRAN package AER (Kleiber and Zeileis, 2013) for the sake of convenience.

References

Chen, Y. and Samworth, R. J. (2016). Generalized additive and index models with shape constraints. Journal of the Royal Statistical Society: Series B, 78, 729-754.

Kleiber, C. and Zeileis, A. (2013). *AER: applied econometrics with R*. R package version 1.2-0, http://cran.r-project.org/web/packages/AER/

Long, J.S. (1990). *Regression Models for Categorical and Limited Dependent Variables*. Thousand Oaks: Sage Publications.

Long, J.S. (1997). The origin of sex differences in science. *Social Forces*, 68, 1297–1315.

Examples

```
## Some data pre-processing
data(PhDPublications)
dat = matrix(0,ncol=4,nrow=nrow(PhDPublications))
dat[,1] = as.numeric(PhDPublications$gender == "female")
dat[,2] = as.numeric(PhDPublications$married == "yes")
dat[,3] = as.numeric(PhDPublications$kids)
dat[,4] = as.numeric(PhDPublications$mentor)
y = PhDPublications$articles

## Run scar on the dataset
object = scar(dat,y,shape=c("l","l","de","ccv"),family=poisson())

## Check the effects of mentor
plot(object,which=c(4))
```

`plot.scair` *Plot additive index components of a "scair" object*

Description

This function takes a fitted `scair` object produced by `scair` and plots the component functions of the additive indices that make it up, on the scale of the additive index predictors.

Usage

```
## S3 method for class 'scair'
plot(x, whichplot = 1:m, addp = TRUE, ...)
```

Arguments

`x`	A fitted `scair` object produced by `scair`.
`whichplot`	A numeric vector indicates the required plots.
`addp`	A `logical` scalar that indicates whether the data points should be plotted as circles on the fitted curves of each component.
`...`	Other arguments passed to `plot`.

Details

A plot is produced for each additive index component function in the fitted `scair` object. Each function is plotted in the range of the observed indices, with the identifiability condition satisfied (see Details of `scar` and `scair`).

Author(s)

Yining Chen and Richard Samworth

See Also

`scair`, `predict.scair`, `plot.scar`

Examples

```
## See examples for the function scair
```

`plot.scar` *Plot components of a "scar" object*

Description

This function takes a fitted `scar` object produced by `scar` and plots the component functions that make it up, on the scale of the additive predictors.

Usage

```
## S3 method for class 'scar'
plot(x, whichplot = 1:d, addp = TRUE, ...)
```

Arguments

`x`	A fitted `scar` object produced by `scar`.
`whichplot`	A numeric vector indicates the required plots.
`addp`	A `logical` scalar that indicates whether the data points should be plotted as circles on the fitted curves of each component.
`...`	Other arguments passed to `plot`.

Details

A plot is produced for each additive component in the fitted `scar` object. Each function is plotted in the range of the observed covariates, with the identifiability condition satisfied (see Details of `scar`).

It can be invoked by calling `"plot(x)"` for an object `x` of the `scar` class, or directly by calling `"plot.scar(x)"` regardless of the class of the object.

Author(s)

Yining Chen and Richard Samworth

See Also

`scar`, `predict.scar`, `plot.scair`

Examples

```
## See examples for the function scar
```

predict.scair *Predict method for* scair *fits*

Description

This function obtains predictions from a fitted scair object.

Usage

```
## S3 method for class 'scair'
predict(object, newdata, type = c("link", "response"), rule=1, ...)
```

Arguments

object	A fitted scair object produced by scair.
newdata	An optional numeric matrix of d columns, with each row specifying a location at which prediction is required. This argument can be missing, in which case predictions are made at the same values of the covariates used to compute the object.
type	Type of predictions, with choices "link" (the default), or "response". The default produces predictions on the scale of the index predictors. If "response" is selected, the predictions are on the scale of the response (i.e. mean of the exponential family), and are monotone transformations of the index predictors using the inverse link function.
rule	An integer describing how to handle the new data outside the range of the observed indices (computed via linear combination of observed covariates). If rule=1, then we use linear interpolation to get the value of each fitted component function outside the range of observed covariates. Otherwise if rule=2, then the value at the closest data extreme is used. Note that if there is convex/concave component, the choice of the first rule can lead to somewhat unsatifactary performance on/outside the "boundary" of the data (when comparing to the second rule).
...	Further arguments passed to or from other methods.

Value

A numeric vector of predictions.

Author(s)

Yining Chen and Richard Samworth

See Also

scair, plot.scair, predict.scar

Examples

```
## See examples for the function scair
```

`predict.scar` *Predict method for* `scar` *fits*

Description

This function obtains predictions from a fitted `scar` object.

Usage

```
## S3 method for class 'scar'
predict(object, newdata, type = c("link", "response"), rule=1, ...)
```

Arguments

`object`	A fitted `scar` object produced by `scar`.
`newdata`	An optional numeric `matrix` of d columns, with each row specifying a location at which prediction is required. This argument can be missing, in which case predictions are made at the same values of the covariates used to compute the object.
`type`	Type of predictions, with choices "link" (the default), or "response". The default produces predictions on the scale of the additive predictors. If "response" is selected, the predictions are on the scale of the response (i.e. mean of the exponential family), and are monotone transformations of the additive predictors using the inverse link function.
`rule`	An integer describing how to handle the new data outside the range of the observed covariates. If `rule=1`, then we use linear interpolation to get the value of each fitted component function outside the range of observed covariates. Otherwise if `rule=2`, then the value at the closest data extreme is used. Note that if there is convex/concave component, the choice of the first rule can lead to somewhat unsatifactary performance on/outside the boundary of the data (when comparing to the second rule).
`...`	Further arguments passed to or from other methods.

Value

A numeric vector of predictions.

Author(s)

Yining Chen and Richard Samworth

See Also

`scar`, `plot.scar`

Examples

```
## See examples for the function scar
```

`scair` *Maximizing the likelihood of the generalised additive index model with shape constraints*

Description

This function searches for a maximum likelihood estimator (mle) of the generalised additive index regression with shape constraints. A stochastic search strategy is used here.

Each index is a linear combination of some (or all) the covariates. Each additive component function of these index predictors is assumed to belong to one of the nine possible shape restrictions.

The output is an object of class `scair` which contains all the information needed to plot the estimator using the `plot` method, or to evaluate it using the `predict` method.

Usage

```
scair(x,y,shape=rep("l",1), family=gaussian(), weights=rep(1,length(y)),
  epsilon=1e-8, delta=0.1, indexgen=c("unif", "norm"), iter = 200,
  allnonneg = FALSE)
```

Arguments

`x`	Observed covariates in R^d, in the form of an $n \times d$ numeric `matrix`.
`y`	Observed responses, in the form of a numeric `vector` of length n.
`shape`	A vector that specifies the shape restrictions for additive component function of each index (also called the ridge function), in the form of a string vector of length m. Here for the sake of identifiability, we require the number of indices $m \leq d$. The shape constraints we considered (with their coresponding abbreviations used in `shape`) are listed below: `l`: linear `in`: monotonically increasing `de`: monotonically decreasing `cvx`: convex `cvxin`: convex and increasing `cvxde`: convex and decreasing `ccv`: concave `ccvin`: concave and increasing `ccvde`: concave and decreasing
`family`	A description of the error distribution and link function to be used in the model. This can be a character string naming a family function, a family function or the result of a call to a family function. Currently only the following five common exponential families are allowed: Gaussian, Binomial, Poisson, and Gamma. By default the canonical link function is used.

`weights` An optional vector of prior weights to be used when maximising the likelihood. It is a numeric vector of length n. By default equal weights are used.

`epsilon` Positive convergence tolerance epsilon when performing the iteratively reweighted least squares (IRLS) method at each iteration of the active set algorithm in `scar`. See `scar` for more details.

`delta` A tuning parameter used to avoid the perfect fit phenomenon, and to ensure identifiability. It represents the lower bound of the minimum eigenvalue of all possible $A^T A$ subject to identiability conditions, where A is an index matrix. It should be smaller than 1. This parameter is NOT needed when $d = 1$, or the prediction function is convex or concave, or all the entries of the index matrix are non-negative if all ridge functions are increasing or decreasing.

`indexgen` It determines how the index matrices are generated in the stochastic search. If its value is "`unif`", then entries of the index matrices are drawn from uniform distribution; otherwise, if its value is "`norm`", entries are drawn from normal.

`iter` Number of iterations of the stochastic search.

`allnonneg` A boolean variable that specifies whether all the entries of the index matrices are non-negative. If it is true, then `delta` is no longer needed in case the ridge functions are either all increasing, or all decreasing.

Details

For $i = 1, \ldots, n$, let X_i be the d-dimensional covariates, Y_i be the corresponding one-dimensional response and w_i be its weight. The generalised additive index model can be written as

$$g(\mu) = f(x),$$

where $x = (x_1, \ldots, x_d)^T$, g is a known link function, A is an $d \times m$ index matrix, and f is an additive function. Our task is to estimate both the index matrix and the additive function.

Assume the canonical link function is used here, then the maximum likelihood estimator of the generalised additive index model based on observations $(X_1, Y_1), \ldots, (X_n, Y_n)$ is the function that maximises

$$\frac{1}{n} \sum_{i=1}^{n} w_i \{Y_i f(A^T X_i) - B(f(A^T X_i))\}$$

subject to the restrictions that for every $j = 1, \ldots, m$, the j-th additive component of f satisfies the constraint indicated by the j-th element of `shape`. Here $B(.)$ is the log-partition function of the specified exponential family distribution, and w_i are the weights. For i.i.d. data, w_i should be 1 for each i.

For any given A, the optimization problem can solved using the active set algorithm implemented in `scar`. Therefore, this problem can be reduced to a finite-dimensional optimisation problem. Here we apply a simple stochastic search strategy is proposed, though other methods, such as downhill simplex, is also possible (and sometimes offers competitive performance). All the implementaton details can be found in *Chen and Samworth (2016)*, where theoretical justification of our estimator (i.e. uniform consistency) is also given.

For the identifiability of additive index models, we refer to *Yuan (2011)*.

Value

An object of class `scair`, with the following components:

`x`	Covariates copied from input.
`y`	Response copied from input.
`shape`	Shape vector copied from input.
`weights`	Vector of weights copied from input.
`family`	The exponential family copied from input.
`componentfit`	Value of the fitted component function at each observed index (computed using the estimated index matrix), in the form of an $n \times m$ numeric `matrix`, where the element at the i-th row and the j-th column is the value of f_j at the j-th coordinate of $A^T X_i$, with the identifiability condition satisfied (see details of `scar`).
`constant`	The estimated value of the constant c in the additive function f (see details of `scar`)).
`deviance`	Up to a constant, minus twice the maximised log-likelihood. Where applicable, the constant is chosen to make the saturated model to have zero deviance. See also `glm`.
`nulldeviance`	The deviance for the null model.
`delta`	A parameter copied from input.
`iter`	Total number of iterations of the stochastic search algorithm
`allnonneg`	specifies whether all entris of the index matrix is non-negative, copied from input.

Author(s)

Yining Chen and Richard Samworth

References

Chen, Y. and Samworth, R. J. (2016). Generalized additive and index models with shape constraints. Journal of the Royal Statistical Society: Series B, 78, 729-754.

Yuan, M. (2011). On the identifiability of additive index models. Statistica Sinica, 21, 1901-1911.

See Also

`plot.scair`, `predict.scair`, `scar`, `decathlon`

Examples

```
## An example in the Gaussian additive index regression setting:
## Define the additive function f on the scale of the predictors
f<-function(x){
  return((0.5*x[,1]+0.25*x[,2]-0.25*x[,3])^2)
}
```

```
## Simulate the covariates and the responses
## covariates are drawn uniformly from [-1,1]^3
set.seed(10)
d = 3
n = 500
x = matrix(runif(n*d)*2-1,nrow=n,ncol=d)
y = f(x) + rnorm(n,sd=0.5)

## Single index model so no delta is required here
shape=c("cvx")
object = scair(x,y,shape=shape, family=gaussian(),iter = 100)

## Estimated index matrix
object$index

## Root squared error for the estimated index
sqrt(sum((object$index - c(0.5,0.25,-0.25))^2))

## Plot the estimatied additive function for the single index
plot(object)

## Evaluate the estimated prediction function at 10^4 random points
## drawing from the interior of the support
testx = matrix((runif(10000*d)*1.96-0.98),ncol=d)
testf = predict(object,testx)

## and calculate the (estimated) absolute prediction error
mean(abs(testf-f(testx)))

## Here we can treat the obtained index matrix as a warm start and perform
## further optimization (on the second and third entry of the index)
## using e.g. the default R optimisation routine.
fn<-function(w){
    dev = Inf
    if (abs(w[1])+abs(w[2])>1) return(dev)
    else {
      wnew = matrix(c(1-abs(w[1])-abs(w[2]),w[1],w[2]),ncol=1)
      dev = scar(x %*% wnew, y, shape = "cvx")$deviance
      return (dev)
    }
}
index23 = optim(object$index[2:3],fn)$par
newindex = matrix(c(1-sum(abs(index23)),index23),ncol=1); newindex

## Root squared error for the new estimated index
sqrt(sum((newindex - c(0.5,0.25,-0.25))^2))

## A further example is provided in decathlon dataset
```

scar *Compute the maximum likelihood estimator of the generalised additive regression with shape constraints*

Description

This function uses the active set algorithm to compute the maximum likelihood estimator (mle) of the generalised additive regression with shape constraints. Each component function of the additive predictors is assumed to belong to one of the nine possible shape restrictions. The estimator's value at the data points is unique.

The output is an object of class `scar` which contains all the information needed to plot the estimator using the `plot` method, or to evaluate it using the `predict` method.

Usage

```
scar(x, y, shape = rep("l", d), family = gaussian(),
  weights = rep(1, length(y)), epsilon = 1e-08)
```

Arguments

`x`	Observed covariates in R^d, in the form of an $n \times d$ numeric `matrix`.
`y`	Observed responses, in the form of a numeric `vector` of length n.
`shape`	A vector that specifies the shape restrictions for each component function, in the form of a string vector of length d. The string allowed and its corresponding shape constraint is listed as follows (see Details): `l`: linear `in`: monotonically increasing `de`: monotonically decreasing `cvx`: convex `cvxin`: convex and increasing `cvxde`: convex and decreasing `ccv`: concave `ccvin`: concave and increasing `ccvde`: concave and decreasing
`family`	A description of the error distribution and link function to be used in the model. This can be a character string naming a family function, a family function or the result of a call to a family function. Currently only the following five common exponential families are allowed: Gaussian, Binomial, Poisson, and Gamma. By default the canonical link function is used.
`weights`	An optional vector of prior weights to be used when maximising the likelihood. It is a numeric vector of length n. By default equal weights are used.
`epsilon`	Positive convergence tolerance epsilon when performing the iteratively reweighted least squares (IRLS) method at each iteration of the active set algorithm. See `glm.control` for more details.

Details

For $i = 1, \ldots, n$, let X_i be the d-dimensional covariates, Y_i be the corresponding one-dimensional response and w_i be its weight. The generalised additive model can be written as

$$g(\mu) = f(x),$$

where $x = (x_1, \ldots, x_d)^T$, g is a known link function and f is an additive function (to be estimated).

Assume the canonical link function is used here, then the maximum likelihood estimator of the generalised additive model based on observations $(X_1, Y_1), \ldots, (X_n, Y_n)$ is the function that maximises

$$\frac{1}{n} \sum_{i=1}^{n} w_i \{Y_i f(X_i) - B(f(X_i))\}$$

subject to the restrictions that for every $j = 1, \ldots, d$, the j-th additive component of f satisfies the constraint indicated by the j-th element of `shape`. Here $B(.)$ is the log-partition function of the specified exponential family distribution, and w_i are the weights. For i.i.d. data, w_i should be 1 for each i.

To make each component of f identifiable, we write

$$f(x) = \sum_{j=1}^{d} f_j(x_j) + c$$

and let $f_j(0) = 0$ for every $j = 1, \ldots, d$. In case zero is outside the range of the j-th observed covariate, for the sake of convenience, we set f_j to be zero at the sample mean of the j-th predictor.

This problem can then be re-written as a concave optimisation problem, and our function uses the active set algorithm to find out the maximum likelihood estimator. A general introduction can be found in *Nocedal and Wright (2006)*. A detailed description of our algorithm can be found in *Chen and Samworth (2016)*. See also *Groeneboom, Jongbloed and Wellner (2008)* for some theoretical supports.

Value

An object of class `scar`, with the following components:

`x`	Covariates copied from input.
`y`	Response copied from input.
`shape`	Shape vector copied from input.
`weights`	The vector of weights copied from input.
`family`	The exponential family copied from input.
`componentfit`	Value of the fitted component function at each observed covariate, in the form of an $n \times d$ numeric `matrix`, where the element at the i-th row and the j-th column is the value of f_j at the j-th coordinate of X_i, with the identifiability condition satisfied (see Details)
`constant`	The estimated value of the constant c in the additive function f (see Details).
`deviance`	Up to a constant, minus twice the maximised log-likelihood. Where applicable, the constant is chosen to make the saturated model to have zero deviance. See also `glm`.

nulldeviance	The deviance for the null model.
iter	Total number of iterations of the active set algorithm

.

Note

We acknowledge that glm.fit from the R package **stats** is called to perform the method of iterated reweighted least squares (IRLS) in our routine. It is possible to speed up the implementation considerably by simply suppressing all the run-time checks there.

If all the component functions are linear, then it is prefered to call directly the function glm.

For the one-dimensional covariate, see the pool adjacent violators algorithm (PAVA) of *Robertson, Wright and Dykstra (1998)* and the support reduction method of *Groeneboom, Jongbloed and Wellner (2008)*.

A different approach to tackle this problem is to use splines. See the R package scam. We stress here that our approach is free of tuning parameters while scam is not, which can be viewed as a major difference.

To estimate the generalised additive regression function without any shape restrictions, see *Wood (2004)* and *Hastie and Tibshirani (1990)*. Their corresponding R implementations are mgcv and gam.

Author(s)

Yining Chen and Richard Samworth

References

Chen, Y. and Samworth, R. J. (2016). Generalized additive and index models with shape constraints. Journal of the Royal Statistical Society: Series B, 78, 729-754.

Groeneboom, P., Jongbloed, G. and Wellner, J.A. (2008). The support reduction algorithm for computing non-parametric function estimates in mixture models. Scandinavian Journal of Statistics, 35, 385-399.

Hastie, T. and Tibshirani, R. (1990). Generalized Additive Models. Chapman and Hall, London.

Meyer, M. C. (2013). Semi-parametric additive constrained regression. Journal of nonparametric statistics, 25, 715-743.

Nocedal, J., and Wright, S. J. (2006). Numerical Optimization, 2nd edition. Springer, New York.

Robertson, T., Wright, F. T. and Dykstra, R. L. (1988). Order Restricted Statistical Inference. Wiley, New York.

Venables, W. N. and Ripley, B. D. (2002). Modern Applied Statistics with S. Springer, New York.

Wood, S.N. (2004). Stable and efficient multiple smoothing parameter estimation for generalized additive models. Journal of American Statistical Association, 99, 673-686.

See Also

plot.scar, predict.scar, scair, scam, mgcv, gam, glm

Examples

```
## An example in the Poission additive regression setting:
## Define the additive function f on the scale of the predictors
f<-function(x){
  return(1*abs(x[,1]) + 2*(x[,2])^2 + 3*(abs(x[,3]))^3)
}

## Simulate the covariates and the responses
## covariates are drawn uniformly from [-1,1]^3
set.seed(0)
d = 3
n = 500
x = matrix(runif(n*d)*2-1,nrow=n,ncol=d)
rpoisson <- function(m){rpois(1,exp(m))}
y = sapply(f(x),rpoisson)

## All the components are convex so one can use scar
shape=c("cvx","cvx","cvx")
object = scar(x,y,shape=shape, family=poisson())

## Plot each component of the estimatied additive function
plot(object)

## Evaluate the estimated additive function at 10^4 random points
## drawing from the interior of the support
testx = matrix((runif(10000*d)*1.96-0.98),ncol=d)
testf = predict(object,testx)

## and calculate the (estimated) absolute prediction error
mean(abs(testf-f(testx)))
```

Index