Package ‘MonoPoly’

February 19, 2015

Type Package
Title Functions to fit monotone polynomials
Version 0.2-8
Date 2013-11-03
Description Package contains functions to fit monotone polynomials to data
License GPL (>= 2)
Depends R (>= 2.15.0), quadprog
LazyData yes
Encoding latin1
Author Berwin A. Turlach [aut, cre], Kevin Murray [ctb]
Maintainer Berwin A. Turlach <Berwin.Turlach@gmail.com>
NeedsCompilation yes
Repository CRAN
Date/Publication 2013-11-03 12:44:22

R topics documented:

c coef.monpol ............................................................... 2
evalPol ................................................................. 3
fitted.monpol ............................................................. 4
hawkins ................................................................. 4
model.matrix.monpol ...................................................... 5
monpol ................................................................. 6
monpol.control ......................................................... 7
monpol.fit ............................................................ 8
predict.monpol ........................................................... 10
print.monpol ............................................................ 11
residuals.monpol ......................................................... 11
w0 ................................................................. 12
w2 ................................................................. 13
coef.monpol

Index 14

---

**coef.monpol**  *Extract Model Coefficients*

---

### Description

coef method for ‘monpol’ objects.

### Usage

```r
## S3 method for class 'monpol'
coef(object, scale = c("original", "fitted"), type = c("beta", "monpar"), ...)
```

### Arguments

- `object`: A ‘monpol’ object.
- `scale`: Extract coefficients on the original scale of the data or on the scale used during fitting.
- `type`: Extract coefficients in the ‘beta’ parameterisation of the polynomial or for the monotone parameterisation used in the algorithm.
- `...`: Additional optionals arguments. At present no optional arguments are used.

### Details

This is the `coef` method for objects inheriting from class "monpol".

### Value

Coefficients extracted from the model object `object`.

### Author(s)

Berwin A Turlach
**evalPol**

**Evaluating Polynomials**

**Description**
Function to evaluate polynomials in a numerical robust way using the Horner scheme.

**Usage**
`evalPol(x, beta)`

**Arguments**
x
- numerical values at which to evaluate polynomials, can be provided in a vector, matrix, array or data frame

beta
- numerical vector containing the coefficient of the polynomial

**Value**
The result of evaluating the polynomial at the values in `x`, returned in the same dimension as `x` has.

**Author(s)**
Berwin A Turlach

**Examples**
```r
## The function is currently defined as
function (x, beta)
{
    res <- 0
    for(bi in rev(beta))
        res <- res*x + bi
    res
}

beta <- c(1,2,1)
x <- 0:10
evalPol(x, beta)
str(evalPol(x, beta))

x <- cbind(0:10, 10:0)
evalPol(x, beta)
str(evalPol(x, beta))

x <- data.frame(x=0:10, y=10:0)
evalPol(x, beta)
str(evalPol(x, beta))
```
fitted.monpol  

Extract Model Fitted Values

Description

fitted method for ‘monpol’ objects.

Usage

```r
## S3 method for class 'monpol'
fitted(object, scale = c("original", "fitted"), ...)
```

Arguments

- `object`: A ‘monpol’ object.
- `scale`: Extract fitted values on the original scale of the data or on the scale used during fitting.
- `...`: Additional optional arguments. At present no optional arguments are used.

Details

This is the `fitted` method for objects inheriting from class "monpol".

Value

Fitted values extracted from the model object `object`.

Author(s)

Berwin A Turlach

Description

This data gives x and y variables for the data published in Hawkins’ 1994 article. This data was originally simulated from a standard cubic polynomial with equally spaced x values between -1 and 1.

Format

A data frame with 50 simulated observations on the following 2 variables.

- `y`  a numeric vector
- `x`  a numeric vector
References


Examples

data(hawkins)

---

**model.matrix.monpol**  
*Construct Design Matrices*

**Description**

`model.matrix` creates a design (or model) matrix for ‘monpol’ objects.

**Usage**

```r
## S3 method for class 'monpol'
model.matrix(object, scale = c("original", "fitted"), ...)
```

**Arguments**

- `object`  
  A ‘monpol’ object.
- `scale`  
  Create design matrix on the original scale of the data or on the scale used during fitting.
- `...`  
  Additional optional arguments. At present no optional arguments are used.

**Details**

This is the `model.matrix` method for objects inheriting from class "monpol".

**Value**

Design matrix created from the model object object.

**Author(s)**

Berwin A Turlach
monpol

Monotone Polynomials

Description

Determine the least-squares estimates of the parameters of a monotone polynomial

Usage

monpol(formula, data, subset, weights, na.action,
              degree=3, K, start, trace = FALSE, plot.it = FALSE,
              control = monpol.control(),
              algorithm = c("Full", "Hawkins", "BCD", "CD1", "CD2"),
              ptype = c("Elphinstone", "EHH", "Penttila"),
              ctype = c("cge0", "c2"),
              model=FALSE, x=FALSE, y=FALSE)

Arguments

formula an object of class "formula" (or one that can be coerced to that class): a symbolic description of the model to be fitted.
data an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which monpol is called.
subset an optional vector specifying a subset of observations to be used in the fitting process.
weights an optional vector of weights to be used in the fitting process. Should be NULL or a numeric vector.
na.action a function which indicates what should happen when the data contain NAs. The default is set by the na.action setting of options, and is na.fail if that is unset. The ‘factory-fresh’ default is na.omit. Another possible value is NULL, no action. Value na.exclude can be useful.
degree a polynomial with highest power equal to degree will be fitted to the data.
K a polynomial with highest power $2K + 1$ will be fitted to the data.
start optional starting value for the iterative fitting.
trace print out information about the progress of the interactive fitting at the start and then every trace iterations.
plot.it plot the data and initial fit, then plot current fit every plot.it iterations.
control settings that control the iterative fit; see monpol.control for details.
algorithm algorithm to be used. It is recommended to use either ‘Full’ or ‘Hawkins’; see paper in ‘References’ for details.
ptype parameterisation to be used; see paper in ‘References’ for details.
ctype
model, x, y

parameterisation to be used; see paper in ‘References’ for details.
logicals. If TRUE the corresponding components of the fit (the model frame, the
model matrix, the response, the QR decomposition) are returned.

Details

A monpol object is a type of fitted model object. it has methods for the generic function coef, fitted, formula, loglik, model.matrix, predict, print, residuals.

Value

monpol returns an object of class "monpol"

Author(s)

Berwin A Turlach

References


Examples

monpol(y~x, w0)

monpol.control

Control the Iterations in monpol

Description

Allow the user to set some characteristics of the monpol monotone polynomial fitting algorithm.

Usage

monpol.control(maxiter = 1000, tol = 1e-05,
               toli=1e-10, tol2=1e-07, tolqr=1e-07)

Arguments

maxiter A positive integer specifying the maximum number of iterations allowed, used
         in all algorithms.
tol A positive numeric value specifying an absolute tolerance for determining whether
     entries in the gradient are zero for algorithms ‘Full’, ‘BCD’, ‘CD1’ and ‘CD2’.
toli A positive numeric value, used in algorithm ‘Hawkins’. Any number not smaller
      than −tol1 is deemed to be non-negative.
tol2 A positive numeric value, used in algorithm ‘Hawkins’. Any number whose absolute value is smaller than tol2 is taken to be zero.

tolqr A positive numeric value, used in algorithm ‘Hawkins’ as tolerance for the QR factorisation of the design matrix.

Value
A list with exactly five components:

maxiter
tol
tol1
tol2
tolqr

with meanings as explained under ‘Arguments’.

Author(s)
Berwin A Turlach

See Also
monpol, monpol.fit, qr

Examples

monpol.control(maxiter = 2000)
monpol.control(tolqr = 1e-10)

monpol.fit Monotone Polynomials

Description
This is the basic computing engine called by monpol used to fit monotonic polynomials. These should usually not be used directly unless by experienced users.

Usage

monpol.fit(x, y, w, K=1, start, trace = FALSE, plot.it = FALSE, control = monpol.control(),
algorithm = c("Full", "Hawkins", "BCD", "CD1", "CD2"),
ptype = c("Elphinstone", "EHH", "Penttila"),
ctype = c("cge0", "c2"))
Arguments

x  vector containing the observed values for the regressor variable.
y  vector containing the observed values for the response variable; should be of same length as x.
w  optional vector of weights; should be of the same length as x if specified.
K  a polynomial with highest power $2K + 1$ will be fitted to the data.
start  optional starting value for the iterative fitting.
trace  print out information about the progress of the iterative fitting at the start and then every trace iterations.
plot.it  plot the data and initial fit, then plot current fit every plot.it iterations.
control  settings that control the iterative fit; see monpol.control for details.
algorithm  algorithm to be used; see monpol for details.
ptype  parameterisation to be used; see monpol for details.
cctype  parameterisation to be used; see monpol for details.

Value

a list with components

par  the fitted parameters.
grad  the gradient of the objective function at the fitted parameters.
beta  the coefficients of the fitted polynomial in the ‘beta’ parameterisation; on the fitted scale.
RSS  the value of the objective function; on the fitted scale.
niter  number of iterations.
converged  indicates whether algorithm has converged.
ptype  input parameter ptype.
cctype  input parameter cctype.
beta.raw  the coefficients of the fitted polynomial in the ‘beta’ parameterisation; on the original scale.
fitted.values  the fitted values; on the fitted scale.
residuals  the residuals; on the fitted scale.
K  input parameter K.
minx  the minimum value in the vector x.
sclx  the difference between the maximum and minimum values in the vector x.
miny  the minimum value in the vector y.
scly  the difference between the maximum and minimum values in the vector y.
algorithm  input parameter algorithm.

Author(s)

Berwin A Turlach
References


See Also

monpol which you should use for fitting monotonic polynomials unless you know better.

predict.monpol

Predicting from Monotone Polynomial Fits

Description

predict.monpol produces predicted values, obtained by evaluating the monotone polynomial in the frame newdata.

Usage

## S3 method for class 'monpol'
predict(object, newdata, scale = c("original", "fitted"), ...)

Arguments

object A 'monpol' object.
newdata A named list or data frame in which to look for variables with which to predict. If newdata is missing the fitted values at the original data points are returned.
scale Predict values on the original scale of the data or on the scale used during fitting. Data in newdata is assumed to be on the indicated scale.
... Additional optionals arguments. At present no optional arguments are used.

Details

This is the predict method for objects inheriting from class "monpol".

Value

predict.monpol produces a vector of predictions.

Author(s)

Berwin A Turlach
**print.monpol**

**Printing Monotone Polynomials**

### Description

print method for 'monpol' objects.

### Usage

```r
## S3 method for class 'monpol'
print(x, digits = max(3,getOption("digits") - 3), ...)
```

### Arguments

- `x`: A 'monpol' object.
- `digits`: minimal number of significant digits, see `print.default`.
- `...`: Additional optional arguments. At present only those additaion arguments for `coef.monpol` are used.

### Details

This is the `print` method for objects inheriting from class "monpol".

### Value

`x` returned invisibly.

### Author(s)

Berwin A Turlach

---

**residuals.monpol**

**Extract Model Residuals**

### Description

residuals method for 'monpol' objects.

### Usage

```r
## S3 method for class 'monpol'
residuals(object, scale = c("original", "fitted"), ...)
```
Arguments

object  A 'monpol' object.
scale   Extract residuals on the original scale of the data or on the scale used during fitting.
...     Additional optional arguments. At present no optional arguments are used.

Details

This is the residuals method for objects inheriting from class "monpol".

Value

Residuals extracted from the model object object.

Author(s)

Berwin A Turlach

---

Simulated w0 data used in Murray et al. (2013)

Description

This data set gives simulated data from the function

\[ y = 0.1x^3 + e \]

for \( e \sim N(0, 0.01^2) \) and \( x \) evenly spaced between -1 and 1.

Format

A data frame with 21 observations on the following 2 variables.

- y  a numeric vector
- x  a numeric vector

Source


Examples

- `str(w0)`
- `plot(y~x, w0)`
- `monpol(y~x, w0)`
Simulated w2 data used in Murray et al. (2013)

Description

Simulated data from the function

\[ y_{ij} = 4\pi - x_i + \cos(x_i - \frac{\pi}{2}) + e_{ij} \]

for \( x_i = 0, 1, \ldots, 12 \); \( n_i = 5 \) for \( i = 0 \) and \( n_i = 3 \) otherwise; \( e_{ij} \sim N(0, 0.5^2) \)

Format

A data frame with 41 observations on the following 2 variables.

- \( y \) a numeric vector
- \( x \) a numeric vector

Source


Examples

```r
str(w2)
plot(y~x, w2)
monpol(y~x, w2)
monpol(y~x, w2, K=2)
```
Index

*Topic datasets
  hawkins, 4
  w0, 12
  w2, 13
*Topic models
  coef.monpol, 2
  fitted.monpol, 4
  model.matrix.monpol, 5
  monpol, 6
  monpol.control, 7
  monpol.fit, 8
  predict.monpol, 10
  residuals.monpol, 11
*Topic nonlinear
  monpol, 6
  monpol.fit, 8
  predict.monpol, 10
*Topic print
  print.monpol, 11
*Topic regression
  coef.monpol, 2
  evalPol, 3
  fitted.monpol, 4
  monpol, 6
  monpol.control, 7
  monpol.fit, 8
  predict.monpol, 10
  residuals.monpol, 11
*Topic utilities
  evalPol, 3
  fitted, 4, 7
  fitted.monpol, 4
  fitted.values.monpol(fitted.monpol), 4
  formula, 6, 7
  hawkins, 4
  logLik, 7
  model.matrix, 5, 7
  model.matrix.monpol, 5
  monpol, 6, 8–10
  monpol.control, 6, 7, 9
  monpol.fit, 8, 8
  na.exclude, 6
  na.fail, 6
  na.omit, 6
  options, 6
  predict, 7, 10
  predict.monpol, 10
  print, 7, 11
  print.default, 11
  print.monpol, 11
  qr, 8
  resid.monpol(residuals.monpol), 11
  residuals, 7, 12
  residuals.monpol, 11
  w0, 12
  w2, 13

as.data.frame, 6

class, 7
coeff, 2, 7
coeff.monpol, 2, 11

evalPol, 3

14