Package 'Iso'

July 21, 2025

Version 0.0-21

Date 2023-10-02

Title Functions to Perform Isotonic Regression

Author Rolf Turner <rolfturner@posteo.net>

Maintainer Rolf Turner <rolfturner@posteo.net>

Depends R (>= 1.7.0)

Description Linear order and unimodal order (univariate)
isotonic regression; bivariate isotonic regression
with linear order on both variables.

LazyData true

License GPL (>= 2)

NeedsCompilation yes

Repository CRAN

Date/Publication 2023-10-02 06:50:13 UTC

Contents

`biviso` *Bivariate isotonic regression.*

Description

Bivariate isotonic regression with respect to simple (increasing) linear ordering on both variables.

Usage

```
biviso(y, w = NULL, eps = NULL, eps2 = 1e-9, ncycle = 50000,
       fatal = TRUE, warn = TRUE)
```

Arguments

y	The matrix of observations to be isotonized. It must of course have at least two rows and at least two columns.
w	A matrix of weights, greater than or equal to zero, of the same dimension as y. If left NULL then w is created as a matrix all of whose entries are equal to 1.
eps	Convergence criterion. The algorithm is deemed to have converged if each entry of the output matrix, after the completion of the current iteration, does not differ by more than eps from the corresponding entry of the matrix after the completion of the previous iteration. If this argument is not supplied it defaults to sqrt(.Machine$double.eps).
eps2	Criterion used to determine whether isotonicity is "violated", whence whether (further) application of the "pool adjacent violators" procedure is required.
ncycle	The maximum number of cycles of the iteration procedure. Must be at least 2 (otherwise an error is given). If the procedure has not converged after ncycle iterations then an error is given. (See below.)
fatal	Logical scalar. Should the function stop if the subroutine returns an error code other than 0 or 4? If fatal is FALSE then output is returned by the function even if there was a "serious" fault. One can set fatal=FALSE to inspect the values of the objective matrix at various interim stages prior to convergence. See **Examples**.
warn	Logical scalar. Should a warning be produced if the subroutine returns a value of ifault equal to 4 (or to any other non-zero value when fatal has been set to FALSE)?

Details

See the paper by Bril et al., (*References*) and the references cited therein for details.

Value

A matrix of the same dimensions as y containing the corresponding isotonic values. It has an attribute icycle equal to the number of cycles required to achieve convergence of the algorithm.

Error Messages

The subroutine comprising Algorithm AS 206 produces an error code ifault with values from 0 to 6 The meaning of these codes is as follows:

- 0: No error.
- 1: Convergence was not attained in ncycle cycles.
- 2: At least one entry of w was negative.

- 3: Either `nrow(y)` or `ncol(y)` was less than 2.
- 4: A near-zero weight less than `delta=0.00001` was replaced by `delta`.
- 5: Convergence was not attained *and* a non-zero weight was replaced by `delta`.
- 6: All entries of `w` were less than `delta`.

If `ifault==4` a warning is given. All of the other non-zero values of `ifault` result in an error being given.

WARNING

This function appears not to achieve exact isotonicity, at least not quite. For instance one can do:

```
set.seed(42)
u  <- matrix(runif(400),20,20)
iu <- biviso(u)
any(apply(iu,2,is.unsorted))
```

and get `TRUE`. It turns out that columns 13, 14, and 16 of `iu` have exceptions to isotonicity. E.g. six of the values of `diff(iu[,13])` are less than zero. However only one of these is less than `sqrt(.Machine$double.eps)`, and then only "marginally" smaller.

So some of these negative values are "numerically different" from zero, but not by much. The largest in magnitude in this example, from column 16, is `-2.217624e-08` — which is probably not of "practical importance".

Note also that this example occurs in a very artificial context in which there is no actual isotonic structure underlying the data.

Author(s)

Rolf Turner `<rolfturner@posteo.net>`

References

Bril, Gordon; Dykstra, Richard; Pillers Carolyn, and Robertson, Tim ; Isotonic regression in two independent variables; Algorithm AS 206; JRSSC (Applied Statistics), vol. 33, no. 3, pp. 352-357, 1984.

See Also

`pava()` `pava.sa()` `ufit()`

Examples

```
x <- 1:20
y <- 1:10
xy <- outer(x,y,function(a,b){a+b+0.5*a*b}) + rnorm(200)
ixy <- biviso(xy)

set.seed(42)
u <- matrix(runif(400),20,20)
```

```
v <- biviso(u)
progress <- list()
for(n in 1:9) progress[[n]] <- biviso(u,ncycle=50*n,fatal=FALSE,warn=FALSE)
```

---

pava *Linear order isotonic regression.*

Description

The "pool adjacent violators algorithm" (PAVA) is applied to calculate the isotonic regression of a set of data, with respect to the usual increasing (or decreasing) linear ordering on the indices.

Usage

```
pava(y, w, decreasing=FALSE, long.out=FALSE, stepfun=FALSE)
pava.sa(y, w, decreasing=FALSE, long.out=FALSE, stepfun=FALSE)
```

Arguments

y	Vector of data whose isotonic regression is to be calculated.
w	Optional vector of weights to be used for calculating a weighted isotonic regression; if w is not given, all weights are taken to equal 1.
decreasing	Logical scalar; should the isotonic regression be calculated with respect to *decreasing* (rather than increasing) order?
long.out	Logical argument controlling the nature of the value returned.
stepfun	Logical scalar; if TRUE a step function representation of the isotonic regression is returned.

Details

The function pava() uses dynamically loading of a fortran subroutine "pava" to effect the computations. The function pava.sa() ("sa" for "stand-alone") does all of the computations in raw R. Thus pava.sa() could be considerably slower for large data sets.

The x values for the step function returned by these functions (if stepfun is TRUE) are thought of as being 1, 2, ..., n=length(y). The knots of the step function are the x values (indices) *following* changes in the y values (i.e. the starting indices of the level sets, except for the first level set). The y value corresponding to the first level set is the "left hand" value of y or yleft. The step function is formed using the default arguments of stepfun(). In particular it is *right* continuous.

Value

If long.out is TRUE then the result returned consists of a list whose components are:

y	the fitted values
w	the final weights

tr	a set of indices made up of the smallest index in each level set, which thus "keeps track" of the level sets.
h	a step function which represents the results of the isotonic regression. This component is present *only if* stepfun is TRUE.

If long.out is FALSE and stepfun is TRUE then only the step function is returned.

If long.out and stepfun are both FALSE then only the vector of fitted values is returned.

Author(s)

Rolf Turner <rolfturner@posteo.net>

References

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

See Also

ufit() stepfun() biviso()

Examples

```
# Increasing order:
y <- (1:20) + rnorm(20)
ystar <- pava(y)
plot(y)
lines(ystar,type='s')
# Decreasing order:
z <- NULL
for(i in 4:8) {
z <- c(z,rep(8-i+1,i)+0.05*(0:(i-1)))
}
zstar <- pava(z,decreasing=TRUE)
plot(z)
lines(zstar,type='s')
# Using the stepfunction:
zstar <- pava(z,decreasing=TRUE,stepfun=TRUE)
plot(z)
plot(zstar,add=TRUE,verticals=FALSE,pch=20,col.points="red")
```

---

ufit *Unimodal isotonic regression.*

Description

A "divide and conquer" algorithm is applied to calculate the isotonic regression of a set of data, for a unimodal order. If the mode of the unimodal order is not specified, then the optimal (in terms of minimizing the error sum of squares) unimodal fit is calculated.

Usage

```
ufit(y, lmode=NULL, imode=NULL, x=NULL, w=NULL, lc=TRUE, rc=TRUE,
        type=c("raw","stepfun","both"))
```

Arguments

y	Vector of data whose isotonic regression is to be calculated.
lmode	Numeric scalar specifiing the location of the mode. It must be one of the values of x (see below) otherwise an error is thrown.
imode	Integer scalar specifying the index, amongst the values of x (see below) of the location of the mode. It must be one of the indices from 1 to n, where n is the length of y, otherwise an error is thrown. It is an error to specify both lmode and imode. Note that if neither lmode nor imode is specified then the function performs an exhaustive search among all possible mode locations for the optimal (in terms of minimising the error sum of squares) location.
x	A somewhat notional vector of x values corresponding to the data vector y; the value of the mode must be given, or will be determined in terms of these x values. Conceptually the model is y = m(x) + E, where m() is a unimodal function with mode at lmode, and where E is random "error". If x is not specified, it defaults to an equi-spaced sequence of length n on [0,1] (where n is the length of y).
w	Optional vector of weights to be used for calculating a weighted isotonic regression; if w is not specified then all weights are taken to equal 1.
lc	Logical scalar; should the isotonization be left continuous? If lc==FALSE then the value of the isotonization just before the mode is set to NA, which causes line plots to have a jump discontinuity at (just to the left of) the mode. The default is lc=TRUE.
rc	Logical scalar; should the isotonization be right continuous? If rc==FALSE then the value of the isotonization just after the mode is set to NA, which causes line plots to have a jump discontinuity at (just to the right of) the mode. The default is rc=TRUE.
type	String specifying the type of the output; see **Value**. May be abbreviated.

Details

This function dynamically loads fortran subroutines "pava", "ufit" and "unimode" to do the actual work.

Value

If type=="raw" then the value is a list with components:

x	The argument x if this is specified, otherwise the default value.
y	The fitted values.

mode — The value of the location of the mode as determined by lmode or imode if one of these was specified. Otherwise it is the value of the location of the mode which was found to minimize the error sum of squares.

mse — The mean squared error.

If type=="both" then a component h which is the step function representation of the isotonic regression is added to the foregoing list.

If type=="stepfun" then only the step function representation h is returned.

Author(s)

Rolf Turner <rolfturner@posteo.net>

References

Mureika, R. A., Turner, T. R. and Wollan, P. C. (1992). An algorithm for unimodal isotonic regression, with application to locating a maximum. University of New Brunswick Department of Mathematics and Statistics Technical Report Number 92 – 4.

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

Shi, Ning-Zhong. (1988) A test of homogeneity for umbrella alternatives and tables of the level probabilities. Commun. Statist. — Theory Meth. vol. 17, pp. 657 – 670.

Turner, T. R., and Wollan, P. C. (1997) Locating a maximum using isotonic regression. Computational Statistics and Data Analysis vol. 25, pp. 305 – 320.

See Also

pava() biviso()

Examples

```
y <- c(0,1,2,3,3,2)
f1 <- ufit(y,lmode=0.4) # The third entry of the default
                        # value of x = c(0.0,0.2,0.4,0.6,0.8,1.0).
f2 <- ufit(y,imode=3)   # Identical to f1.
f3 <- ufit(y,lmode=3,x=1:6)   # Effectively the same as f1 and f2.
                              # But is different  in appearance.
f4 <- ufit(y,imode=3,x=1:6)   # Identical to f3.

## Not run:
    ufit(y,lmode=3)     # Throws an error.
    ufit(y,imode=7)     # Throws an error.

## End(Not run)

x <- c(0.00,0.34,0.67,1.00,1.34,1.67,2.00,2.50,3.00,3.50,4.00,4.50,
       5.00,5.50,6.00,8.00,12.00,16.00,24.00)
y <- c(0.0,61.9,183.3,173.7,250.6,238.1,292.6,293.8,268.0,285.9,258.8,
       297.4,217.3,226.4,170.1,74.2,59.8,4.1,6.1)
z <- ufit(y,x=x,type="b")
```

```
plot(x,y)
lines(z,col="red")
plot(z$h,do.points=FALSE,col.hor="blue",col.vert="blue",add=TRUE)
abline(v=z$mode,col="green",lty=2)
```

---

vigour *vigour*

Description

Growth vigour of stands of spruce trees in New Brunswick, Canada.

Usage

```
data("vigour")
```

Format

A data frame with 23 observations (rows). The first column is the year of observation (1965 to 1987 inclusive). The other five columns are observations on the vigour of growth of the given stand in each of the years.

Details

The stands each had different initial tree densities. It was expected that vigour would initially increase (as the trees increased in size) and then level off and start to decrease as the growing trees encroached upon each others' space and competed more strongly for resources such as moisture, nutrients, and light. It was further expected that the position of the mode of the vigour observations would depend upon the initial densities.

Source

These data were collected and generously made available by Kirk Schmidt who was at the time of collecting the data a graduate student in the Department of Forest Engineering at the University of New Brunswick, Fredericton, New Brunswick, Canada. The data were collected as part of his research for his Master's degree (supervised by Professor Ted Needham) at the University of New Brunswick. See Schmidt (1993).

References

K. D. Schmidt (1993). *Development of a precommercial thinning guide for black spruce*. Thesis (M.Sc.F.), University of New Brunswick, Faculty of Forestry.

Examples

```
matplot(vigour[,1],vigour[,2:6],
        main="Growth vigour of stands of New Brunswick spruce",
        xlab="year",ylab="vigour",type="b")
```

Index