Package ‘rgcvpack’

August 28, 2007

Version 0.1-2

Date 2007/08/28

Title R Interface for GCVPACK Fortran Package

Author Xianhong Xie <xie@stat.wisc.edu>

Maintainer Xianhong Xie <xie@stat.wisc.edu>

Description Thin plate spline fitting and prediction

Depends R (>= 2.1.0)

License GPL version 2 or later

URL http://www.stat.wisc.edu/~xie

R topics documented:

`fitTps` *Fitting Thin Plate Smoothing Spline*

Description

Fit thin plate splines of any order with user specified knots

Usage

```
fitTps(x, y, m = 2, knots = NULL, scale.type = "range", method = "v",
    lambda = NULL, cost = 1, nstep.cv = 80, verbose = FALSE, tau = 0)
```

Arguments

`x`	the design data points
`y`	the observation vector
`m`	the order of the spline
`knots`	the placement the thin plate spline basis
`scale.type`	"range" (default), the x and knots will be rescaled with respect to x; "none", nothing is done on x and knots
`method`	"v", GCV is used for choosing lambda; "d", user specified lambda
`lambda`	only used when method="d"
`cost`	the fudge factor for inflating the model degrees of freedom, default to be 1
`nstep.cv`	the number of initial steps for GCV grid search
`verbose`	whether some computational details should be outputed
`tau`	the truncation ratio used in SVD when knots is specified by the user, some possible values are 1, 10, 100, ...

Details

The minimization problem for this function is

$$\sum_{i=1}^{n}(y_i - f(x_i))^2 + \lambda * J_m(f),$$

where $J_m(.)$ is the m-the order thin plate spline penalty functional.

If scale.type="range", each column of x is rescaled to [0 1] in the following way x' = (x - min(x))/range(x), and the knots is rescaled w.r.t. min(x) and range(x) in the same way.

When the cost argument is used, the GCV score is computed as

$$\text{GCV}(\lambda) = \frac{n * \text{RSS}(\lambda)}{(n - cost * \text{tr}(A))^2}.$$

Value

A Tps object of the following components

`x`	same as input
`y`	same as input
`m`	same as input
`knots`	same as input
`scale.type`	same as input
`method`	same as input
`lambda`	same as input
`cost`	same as input
`nstep.cv`	same as input
`tau`	same as input
`df`	model degrees of freedom
`gcv`	gcv score of the model adjusted for the fudge factor

`xs`	scaled design points
`ks`	scaled knots design
`c`	coefficient c
`d`	coefficient d
`yhat`	predicted values at the data points
`svals`	singular values of the matrix decomposition
`gcv.grid`	gcv grid table, number of rows=nstep.cv
`call`	the call to this function

Note

This function uses GCVPACK fortran code with some addition and modification by the author.

Author(s)

Xianhong Xie

References

D. Bates, M. Lindstrom, G. Wahba, B. Yandell (1987), GCVPACK – routines for generalized cross-validation. Commun. Statist.-Simula., 16(1), 263-297.

See Also

`predict.Tps`

Examples

```
#define the test function
f <- function(x, y) { .75*exp(-((9*x-2)^2 + (9*y-2)^2)/4) +
                      .75*exp(-((9*x+1)^2/49 + (9*y+1)^2/10)) +
                      .50*exp(-((9*x-7)^2 + (9*y-3)^2)/4) -
                      .20*exp(-((9*x-4)^2 + (9*y-7)^2)) }

#generate a data set with the test function
set.seed(200)
N <- 13; xr <- (2*(1:N) - 1)/(2*N); yr <- xr
zr <- outer(xr, yr, f); zrmax <- max(abs(zr))

noise <- rnorm(N^2, 0, 0.07*zrmax)
zr <- zr + noise #this is the noisy data we will use

#convert the data into column form
xc <- rep(xr, N)
yc <- rep(yr, rep(N,N))
zc <- as.vector(zr)

#fit the thin plate spline with all the data points as knots
tpsfit1 <- fitTps(cbind(xc,yc), zc, m=2, scale.type="none")
persp(xr, yr, matrix(predict(tpsfit1),N,N), theta=130, phi=20,
      expand=0.45, xlab="x1", ylab="x2", zlab="y", xlim=c(0,1),
      ylim=c(0,1),zlim=range(zc), ticktype="detailed", scale=FALSE,
      main="GCV Smooth I")
```

```
#fit the thin plate spline with subset of data points as knots
grid.list  <- list(xc=seq(2/13,11/13,len=10),
                   yc=seq(2/13,11/13,len=10))
knots.grid <- expand.grid(grid.list)

tpsfit2 <- fitTps(cbind(xc,yc), zc, m=2, knots=knots.grid)
persp(xr, yr, matrix(predict(tpsfit2),N,N), theta=130, phi=20,
      expand=0.45, xlab="x1", ylab="x2", zlab="y", xlim=c(0,1),
      ylim=c(0,1),zlim=range(zc), ticktype="detailed", scale=FALSE,
      main="GCV Smooth II")
```

`predict.Tps` *Predicting Thin Plate Smoothing Spline*

Description

Predict the thin plate spline fitting at given new data points.

Usage

```
predict.Tps(object, newdata = NULL, ...)
```

Arguments

`object`	a Tps object returned by fitTps
`newdata`	the new data to be predicted at
`...`	currently not used

Value

A vector with the length = the number of rows in newdata.

Note

This function uses GCVPACK fortran code with some addition and modification by the author.

Author(s)

Xianhong Xie

See Also

`fitTps`

Examples

```
#the same test function as in fitTps
f <- function(x, y) { .75*exp(-((9*x-2)^2 + (9*y-2)^2)/4) +
                      .75*exp(-((9*x+1)^2/49 + (9*y+1)^2/10)) +
                      .50*exp(-((9*x-7)^2 + (9*y-3)^2)/4) -
                      .20*exp(-((9*x-4)^2 + (9*y-7)^2)) }

#generate a data set with the test function
set.seed(200)
N <- 13; xr <- (2*(1:N) - 1)/(2*N); yr <- xr
zr <- outer(xr, yr, f); zrmax <- max(abs(zr))

noise <- rnorm(N^2, 0, 0.07*zrmax)
zr <- zr + noise #this is the noisy data we will use

#convert the data into column form
xc <- rep(xr, N)
yc <- rep(yr, rep(N,N))
zc <- as.vector(zr)

#fit the thin plate spline with all the data points as knots
tpsfit1 <- fitTps(cbind(xc,yc), zc, m=2, scale.type="none")

#predict the thin plate spline on a finer grid (50x50)
xf <- seq(1/26, 25/26, length=50); yf <- xf
zf <- predict(tpsfit1, expand.grid(xc=xf,yc=yf))

#plot the predicted result
persp(xf, yf, matrix(zf,50,50), theta=130, phi=20, expand=0.45,
      xlab="x1", ylab="x2", zlab="y", xlim=c(0,1), ylim=c(0,1),
      zlim=range(zc), ticktype="detailed", scale=FALSE,
      main="GCV Smoothing")
```

Index