Package: assist (via r-universe)

September 15, 2024

Title A Suite of R Functions Implementing Spline Smoothing Techniques

Version 3.1.9

Description Fit various smoothing spline models. Includes an ssr() function for smoothing spline regression, an nnr() function for nonparametric nonlinear regression, an snr() function for semiparametric nonlinear regression, an slm() function for semiparametric linear mixed-effects models, and an snm() function for semiparametric nonlinear mixed-effects models. See Wang (2011) <doi:10.1201/b10954> for an overview.

Depends R (>= 3.0.2), nlme, lattice

License GPL-2

LazyData true

URL https://yuedong.faculty.pstat.ucsb.edu/software.html

NeedsCompilation yes

Author Yuedong Wang [aut, cre], Chunlei Ke [aut], Cleve Moler [cph]

Maintainer Yuedong Wang <yuedong@ucsb.edu>

Repository CRAN

Date/Publication 2023-08-22 07:00:02 UTC

Contents

`acid` *Lake Acidity Study*

Description

The `acid` data frame has 112 rows and 4 columns of data derived based on the Eastern Lakes Survey of 1984 implemented by the Environmental Protection Agency of the USA.

Usage

```
data(acid)
```

Format

The data frame contains the following columns:

ph a numeric vector of surface pH values.

t1 a numeric vector of calcium concentrations in log10 milligrams per liter.

x1, x2 numeric vectors of the lakes' geographic locations.

Details

112 lakes are extracted in the southern Blue Ridge mountains area. The surface pH values were recorded together with the calcium concentration and geographic locations.

Source

Douglas, A. and Delampady, M. (1990), Eastern Lake Survey Phase I: Documentation for the Data Base and the Derived Data sets. Tech Report 160 (SIMS), Dept. Statistics, University of British Columbia.

References

Gu, C. and Wahba, G. (1993) Smoothing Spline ANOVA with component-wise Bayesian confidence intervals. Journal of Computational and Graphic Statistics 55, 353-368.

`alogit` *Calculate the Inverse Logit Transformation*

Description

Perform an inverse logit calculation

Usage

```
alogit(x)
```

Arguments

`x`	a numeric value

Value

Returned is $e^x/(1 + e^x)$.

Author(s)

Chunlei ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

`anova.ssr` *Testing a Non-parametric Function Fitted via Smoothing Splines*

Description

For smoothing spline models with a single smoothing parameter, test the hypothesis that the unknown funciton lies in the null space using the local most powerful (LMP) test and a GCV or GML test.

Usage

```
## S3 method for class 'ssr'
anova(object, simu.size=100, ...)
```

Arguments

`object`	an object of class "ssr" fitted with a single smooting parameter.
`simu.size`	an optional integer giving the number of simulations to calcualte p-values based on simulation. Default is 100.
`...`	other available arguments, currently unused.

Details

For Gaussian data with one smoothing parameter, test the hypothesis that the function is in the null space H_0, i.e. the parametric part of the fitted model is sufficient. Available are the LMP and GCV or GML methods. However, the p-values cannot be calculated analytically since the null distributions for these testing statistics under H_0 are unknown. Monte Carlo simulation is used to approximate the p-values for the LMP, and GCV (if spar="v") or GML (if spar="m") methods. Due to computation burden, the smoothing parameters are fixed at their estimate in the currect calculation.

When spar="m", an approximate p-value based on a mixture of two Chi-square distributions is also provided for the GML test, which tends to be conservationve (Pinherio and Bates, 2002).

Methods further developed in Liu and Wang (2004) and Liu, Meiring and Wang (2004) will be implemented in the future.

Value

a list including test values.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

References

Cox, D. and Koh, E. (1989). A smoothing spline based test of model adequency in polynomial regression. Ann. Ins. Stat. Math. 41, 383-400.

Cox, D., Koh, E., Wahba, G. and Yandell, B.S. (1988). Testing the parameteric null model hypothesis in semi-parametric partial and generalized spline models. Ann. Statist. 16, 113-119.

Wahba, G. (1990). Spline Models for Observational Data. SIAM, Vol. 59.

Pinherio, J. C. and Bates, D. M. (2000) Mixed-effects Models in S and S-Plus. Springer.

Liu, A. and Wang, Y. (2004) Hypothesis Testing in Smoothing Spline Models. Journal of Statistical Computation and Simulation, to appear.

Liu, A., Meiring, W. and Wang, Y. (2004), Testing Generalized Linear Models Using Smoothing Spline Methods. Statistica Sinica, to appear,

See Also

`ssr`, `print.anova.ssr`

Examples

```
## Not run:
data(acid)

# fit a partial thin-plate spline
temp <- ssr(ph~t1+x1+x2, rk=tp(t1), data=acid, spar="m")
anova(temp, 500)

## End(Not run)
```

`Arosa` *Monthly Mean Ozone Thickness in Arosa of Switzerland*

Description

The `Arosa` data frame has 518 rows and 3 columns of data for monthly mean ozone thickness.

Usage

```
data(Arosa)
```

Format

The data frame contains the following columns:

year a vector of integers from 1 to 46 indicating the years when the measures were taken from 1926.

month a vector of integers from 1 to 12 represeting the months in a year.

thick a numeric vetor of mean ozone thickness (Dobson units).

Details

Monthly mean ozone thickness in Arosa, Switzerland was recorded from 1926-1971.

Source

Andrew,D. F. and Herzberg, A. M. (1985). Data: a collection of problems from many fields for the students and research workers. Springer: Berlin: New York.

bdiag — *Construct a Block Diagonal Matrix*

Description

Return a block diagonal matrix formed from the input list of matrices

Usage

```
bdiag(x)
```

Arguments

`x`	a list of matrices

Value

Returned is a matrix of the form diag(x1, ..., xn) where n is the length of the list.

bond — *Treasury and GE bonds*

Description

The `bond` data frame has 1234 rows and 5 columns of data derived from 144 General Electronic Company bonds and 78 Treasury bonds.

Usage

```
data(bond)
```

Format

The data frame contains the following columns:

name a vector of index for individual bond

price a numeric vector of current price

time a numeric vector of future time points at which the payments are made

payment a numeric vector of future payments

type a vector of character strings identifying the groups, "govt" or "ge", which the individual bonds belong to.

Source

Bloomberg

references

Chunlei Ke and Yuedong Wang (2004), Nonlinear Nonparametric Regression Models. Journal of the American Statistical Association 99, 1166-1175.

---

canadaTemp *Monthly Mean Temperatures*

Description

The `canadaTemp` data frame has 420 rows and 3 columns of data for monthly mean temperatures in Canada

Usage

```
data(canadaTemp)
```

Format

The data frame contains the following columns:

temp a numeric vector of mean temperatures at some stations in Canada.

month a vector of integers from 1 to 12 represeting the months in a year.

station a vetor of integers from 1 to 35 indicating the sations where the temperatures were recorded.

Source

The data set was downloaded from `https://www.psych.mcgill.ca/misc/fda/downloads/FDAfuns/R/data/`.

References

Ramsay, J. O and Silverman, B. W. (1997). Functional Data Analysis. New York:Springer.

Ke, C. and Wang, Y. (2001). Semi-parametric Nonlinear Mixed Effects Models and Their Applications. JASA 96:1272-1298.

chickenpox *Chickenpox in New York City*

Description

The `chickenpox` data frame has 498 rows and 3 columns of data recording the number of Chickenpox occurrences in New York City.

Usage

```
data(chickenpox)
```

Format

The data frame contains the following columns:

count the number of monthly reported Chickenpox cases.

month a vector of integers from 1 to 12 representing the month for the reported cases. year a numeric vector representing the year when the cases were reported.

Details

This data frame contains monthly number of reported cases of chickenpox in New York City from 1931 to the first six months of 1972.

Source

The data were downloaded from https://robjhyndman.com/tsdl/.

chol.new *A Modified Cholesky Decomposition*

Description

Returned a matrix forming Cholesky Decomposition

Usage

```
chol.new(Q)
```

Arguments

`Q`	a symmetric matrix, maybe non-positive.

Details

This is used internally as an extension of `chol` that works on a positive matrix.

Value

A mtrix M suth that $XX^T = Q$.

See Also

`chol`

`climate` *Winter Average Temperatures*

Description

The data frame `climate`, obrained from the Carbon Dioxide Information and Analysis Center, has 690 rows and 5 columns of data representing station winter temperature measurements.

Usage

```
data(climate)
```

Format

The data frame contains the following columns:

temp a numeric vector of temperatures in celsius.

lat, long numeric vectors identifying the lattitudes and longitudes of the stations in.

lat.degree, long.degree numeric vectors identifying the lattitudes and longitudes of the stations in degree.

Details

The station winter average temperatures were the averages of the December, January and Febuary monthly average temperatures obtained from the Jones/Wigley data files obtainable from the CDIAC at Oak Ridge National Laboratory in the files ndp020r1/jonesnh.data.Z and ndp020r1/jonessh.dat.Z in the pbu directory at 128.219.24.36.

Source

Jones, P., Wigley, T. and Briffa, K.. lobal and hemisphere temperature anaomalies-land and marine instrumental records. In T. Boden, D. Kaiser, R. Sepanski, and F. Stoss, editors, Trends ’93: A Compendium of Data on Global Change, ORNL/CDIAC-65, pages 603-608, Oak Ridge, TN 1994. CDIAC, Oak Ridge National Laboratory.

dcrdr — *Interface to Fortran Subroutine dcrdr*

Description

Calculate some matrix operations needed to construct Bayesian confidence intervals

Usage

```
dcrdr(rkpk.obj, r)
```

Arguments

`rkpk.obj`	an object returned from calling dsidr
`r`	a matrix to evaluate reproducing kernels on grid points

Value

See the document for the corresponding Fortran subroutine.

deviance.ssr — *Model Deviance*

Description

Extract deviance from a fitted ssr object

Usage

```
## S3 method for class 'ssr'
deviance(object,residuals=FALSE, ...)
```

Arguments

`object`	a fitted `ssr` object
.	
`residuals`	a logical value. If 'TRUE', deviance residuals are returned. If 'FALSE', the sum of deviance residuals squares is returned. Default is FALSE.
`...`	other arguments, currently unused.

Details

This is a method for the function `deviance` for objects inheriting from class `ssr`.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

See Also

ssr

---

dmudr *Interface of dmudr subroutine in RKPACK*

Description

To calculate a spline estimate with multiple smoothing parameters

Usage

```
dmudr(y, q, s, weight = NULL, vmu = "v", theta = NULL, varht = NULL,
    tol = 0, init = 0, prec = 1e-06, maxit = 30)
```

Arguments

y	a numerical vector representing the response.
q	a list, or an array, of square matrices of the same order as the length of y, which are the reproducing kernels evaluated at the design points.
s	the design matrix of the null space H_0 of size (length-of-y,$dim(H_0)$), with elements equal to the bases of H_0 evaluated at design points.
weight	a weight matrix for penalized weighted least-square: $(y - f)'W(y - f) + n\lambda J(f)$. Default is NULL for iid random errors.
vmu	a character string specifying a method for choosing the smoothing parameter. "v", "m" and "u" represent GCV, GML and UBR respectively. "u$\sim$", only used for non-Gaussian family, specifies UBR with estimated variance. Default is "v".
theta	If 'init=1', theta includes intial values for smoothing parameters. Default is NULL.
varht	needed only when vmu="u", which gives the fixed variance in calculation of the UBR function. Default is NULL.
tol	the tolerance for truncation in the tridiagonalization. Default is 0.0.
init	an integer of 0 or 1 indicating if initial values are provided for theta. If init=1, initial values are provided using theta. Default is 0.
prec	precision requested for the minimum score value, where precision is the weaker of the absolute and relative precisions. Default is $1e - 06$.
maxit	maximum number of iterations allowed. Default is 30.

Value

`info`	an integer that provides error message. info=-1 indicates dimension error, info=-2 indicates $F_2^T Q_*^\theta F_2! >= 0$, info=-3 indicates tuning parameters are out of scope, info=-4 indicates fails to converge within maxite steps, info=-5 indicates fails to find a reasonable descent direction, info>0 indicates the matrix S is rank deficient with $info = rank(S) + 1$.
`fit`	fitted values.
`c`	estimates of c.
`d`	estimates of d.
`resi`	vector of residuals.
`varht`	estimate of variance.
`theta`	estimates of parameters $log10(\theta)$.
`nlaht`	the estimate of $log10(nobs * \lambda)$.
`score`	the minimum GCV/GML/UBR score at the estimated smoothing parameters.
`df`	equavilent degree of freedom.
`nobs`	length(y), number of observations.
`nnull`	dim(H_0), number of bases.
`nq`	length(rk), number of reproducing kernels.
`s, q, y`	changed from the inputs.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

References

Gu, C. (1989). RKPACK and its applications: Fitting smoothing spline models. Proceedings of the Statistical Computing Section, ASA, 42-51.

Wahba, G. (1990). Spline Models for Observational Data. SIAM, Vol. 59

See Also

`dsidr`, `gdsidr`, `gdmudr`, `ssr`

dog *Coronary Ainus Potassium Concentrations*

Description

The dog data frame has 252 rows and 4 columns of data considered by Grizzle and Alen (1969)

Usage

```
data(dog)
```

Format

The data frame contains the following columns:

y a numeric vector of meansurements of coronary sinus postassium concentrations.

group a vector of group index for the four groups of dogs.

dog a vector of integers identifying dogs.

time a numeric vector of time points measurements were made.

Details

The data are coronary sinus potassium concentrations measured on each of 36 dogs. These 36 dogs were divided into 4 treatment groups, and the measurements on each dog were taken every two minutes from 1 to 13 minutes after occlusion.

Source

Grizzle, J. E. and Allen, D. M. (1969). Analysis of growth and dose response curves, Biometrics 25: 357-381.

dsidr *Interface of dsidr subroutines in RKPACK*

Description

To calculate a spline estimate with a single smoothing parameter

Usage

```
dsidr(y, q, s=NULL, weight=NULL, vmu="v", varht=NULL,
limnla=c(-10, 3), job=-1, tol=0)
```

Arguments

`y` a numerical vector representing the response.

`q` a square matrix of the same order as the length of y, with elements equal to the reproducing kernel evaluated at the design points.

`s` the design matrix of the null space H_0 of size (length(y),dim(H_0)), with elements equal to the bases of H_0 evaluated at design points. Default is NULL, representing an empty NULL space.

`weight` A weight matrix for penalized weighted least-square: $(y - f)'W(y - f) + n\lambda J(f)$. Default is NULL for iid random errors.

`vmu` a character string specifying a method for choosing the smoothing parameter. "v", "m" and "u" represent GCV, GML and UBR respectively. "u∼", only used for non-Gaussian family, specifies UBR with estimated variance. Default is "v".

`varht` needed only when vmu="u", which gives the fixed variance in calculation of the UBR function. Default is NULL.

`limnla` a vector of length 2, specifying a search range for the n times smoothing parameter on $log10$ scale. Default is $(-10, 3)$.

`job` an integer representing the optimization method used to find the smoothing parameter. The options are job=-1: golden-section search on (limnla(1), limnla(2)); job=0: golden-section search with interval specified automatically; job >0: regular grid search on $[limnla(1), limnla(2)]$ with the number of grids = job + 1. Default is -1.

`tol` tolerance for truncation used in 'dsidr'. Default is 0.0, which sets to square of machine precision.

Value

`info` an integer that provides error message. info=0 indicates normal termination, info=-1 indicates dimension error, info=-2 indicates $F_2^T Q F_2! >= 0$, info=-3 indicates vmu is out of scope, and info>0 indicates the matrix S is rank deficient with info=rank(S)+1.

`fit` fitted values.

`c` estimates of c.

`d` estimates of d.

`resi` vector of residuals.

`varht` estimate of variance.

`nlaht` the estimate of log10(nobs*lambda).

`limnla` searching range for nlaht.

`score` the minimum GCV/GML/UBR score at the estimated smoothing parameter. When job>0, it gives a vector of GCV/GML/UBR functions evaluated at regular grid points.

`df` equavilent degree of freedom.

`nobs` length(y), number of observations.

nnull	dim(H_0), number of bases.
s, qraux, jpvt	QR decomposition of S=FR, as from Linpack 'dqrdc'.
q	first dim(H_0) columns gives $F^T QF_1$, and its bottom-right corner gives tridiagonalization of $F_2^T QF_2$.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

References

Gu, C. (1989). RKPACK and its applications: Fitting smoothing spline models. Proceedings of the Statistical Computing Section, ASA, 42-51.

Wahba, G. (1990). Spline Models for Observational Data. SIAM, Vol. 59.

See Also

dmudr, gdsidr, gdmudr, ssr

---

dsms *Interface to Fortran Subroutine dsms*

Description

Calculate a matrix operation needed to construct Bayesian confidence intervals

Usage

```
dsms(rkpk.obj)
```

Arguments

rkpk.obj	an object returned from calling dsidr

Value

a matrix. See the corresponding Fortran subroutine.

gdmudr *Interface of dbmdr, dbimdr, dgmdr, dpmdr in GRKPACK.*

Description

To calculate a spline estimate with multiple smoothing parameters for non-Gaussian data

Usage

```
gdmudr(y, q, s, family, vmu = "v", varht = NULL,
    init = 0, theta = NULL, tol1 = 0, tol2 = 0, prec1 = 1e-06,
    maxit1 = 30, prec2 = 1e-06, maxit2 = 30)
```

Arguments

`y`	a numerical vector representing the response, or a matrix of two columns for binomial data with the first column as the largest possible counts and the second column as the counts actually obsered.
`q`	a list, or an array, of square matrices of the same order as the length of y, which are the reproducing kernels evaluated at the design points.
`s`	the design matrix of the null space H_0 of size (length-of-y,$dim(H_0)$), with elements equal to the bases of H_0 evaluated at design points.
`family`	a string specifying the family of distribution. Families supported are "binary", "binomial", "poisson" and "gamma" for Bernoulli, binomial, poisson, and gamma distributions respectively. Canonical links are used except for Gamma family where log link is used.
`vmu`	a character string specifying a method for choosing the smoothing parameter. "v", "m" and "u" represent GCV, GML and UBR respectively. "u$\sim$", only used for non-Gaussian family, specifies UBR with estimated variance. Default is "v".
`varht`	needed only when vmu="u", which gives the fixed variance in calculation of the UBR function. Default is 1.0.
`init`	an integer of 0 or 1 indicating if initial values are provided for theta. If init=1, initial values are provided using theta. Default is 0.
`theta`	If 'init=1', theta includes intial values for smoothing parameters. Default is NULL.
`tol1`	the tolerance for elements of w's. Default is 0.0 which sets to square of machine precision.
`tol2`	tolerance for truncation used in 'dsidr'. Default is 0.0 which sets to square of machine precision.
`prec1`	precision requested for the minimum score value, where precision is the weaker of the absolute and relative precisions. Default is 1e-06.
`maxit1`	maximum number of iterations allowed for DMUDR subroutine. Default is 30.
`prec2`	precision requested for stopping the iteration. Default is $1e-06$.
`maxit2`	maximum number of iterations allowed for the iteration in GRKPACK. Default is 30.

Value

info	an integer that provides error message. info=-1 indicates dimension error, info=-2 idicates $F_2^T Q_*^{theta} F_2! >= 0$, info=-3 indicates tuning parameters are out of scope, info=-4 indicates dmudr fails to converge within maxit1 steps, info=-5 indicates dmudr fails to find a reasonable descent direction, info=-6 indicates GRKPACK fails to converge within maxit2 steps, info=-7 indicates there are some w's equals to zero, info>0 indicates the matrix S is rank deficient with $info = rank(S) + 1$.
fit	estimate of the function at design points.
c	estimates of c.
d	estimates of d.
resi	vector of working residuals.
varht	estimate of dispersion parameter.
theta	estimates of parameters $log10(theta)$.
nlaht	the estimate of $log10(nobs * lambda)$.
score	the minimum GCV/GML/UBR score at the estimated smoothing parameters.
df	equavilent degree of freedom.
nobs	length-of-y, number of observations.
nnull	$dim(H_0)$, number of bases.
nq	length(rk), number of reproducing kernels.
s, q, y, init, maxit2	changed from the inputs.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

References

Wahba, G. (1990). Spline Models for Observational Data. SIAM, Vol. 59.

Wang, Y. (1997). GRKPACK: Fitting Smoothing Spline ANOVA Models for Exponential Families. Communications in Statistics: Simulation and Computation, 24: 1037-1059.

See Also

dsidr, dmudr, gdsidr, ssr

gdsidr *Interface of dbsdr, dbisdr, dgsdr, dpsdr in GRKPACK.*

Description

To calculate a spline estimate with single smoothing parameter for non-Gaussian data.

Usage

```
gdsidr(y, q, s, family, vmu="v", varht=NULL, limnla=c(-10, 3),
maxit=30, job=-1, tol1=0, tol2=0, prec=1e-06)
```

Arguments

`y`	a numerical vector representing the response, or a matrix of two columns for binomial data with the first column as the largest possible counts and the second column as the counts actually obsered.
`q`	a square matrix of the same order as the length of y, with elements equal to the reproducing kernel evaluated at the design points.
`s`	the design matrix of the null space H_0 of size (length-of-y,dim(H_0)), with elements equal to the bases of H_0 evaluated at design points.
`family`	a string specifying the family of distribution. Families supported are "binary", "binomial", "poisson" and "gamma" for Bernoulli, binomial, poisson, and gamma distributions respectively. Canonical links are used except for Gamma family where a log link is used.
`vmu`	a character string specifying a method for choosing the smoothing parameter. "v", "m" and "u" represent GCV, GML and UBR respectively. "u$\sim$", only used for non-Gaussian family, specifies UBR with estimated variance. Default is "v".
`varht`	needed only when vmu="u", which gives the fixed variance in calculation of the UBR function. Default is 1.0.
`limnla`	a vector of length 2, specifying a search range for the n times smoothing parameter on log10 scale. Default is (-10, 3).
`maxit`	maximum number of iterations allowed for the iteration in GRKPACK.
`job`	an integer representing the optimization method used to find the smoothing parameter. The options are job=-1: golden-section search on (limnla(1), limnla(2)); job=0: golden-section search with interval specified automatically; job >0: regular grid search on [limnla(1), limnla(2)] with the number of grids = job + 1. Default is -1.
`tol1`	the tolerance for elements of w's. Default is 0.0 which sets to square of machine precision.
`tol2`	tolerance for truncation used in 'dsidr'. Default is 0.0 which sets to square of machine precision.
`prec`	precision requested for stopping the iteration. Default is $1e-06$.

Value

`info`	an integer that provides error message. info=0 indicates normal termination, info=-1 indicates dimension error, info=-2 indicates $F_2^T Q F_2 ! >= 0$, info=-3 indicates vmu is out of scope, info=-4 indicates the algorithm fails to converge at the maxiter steps, info=-5 indicates there are some w's equals to zero, and info>0 indicates the matrix S is rank deficient with info=rank(S)+1.
`fit`	estimate of the function at design points.
`c`	estimates of c.
`d`	estimates of d.
`resi`	vector of working residuals.
`varht`	estimate of dispersion parameter.
`nlaht`	the estimate of $log10(nobs * lambda)$.
`limnla`	searching range for nlaht.
`score`	the minimum GCV/GML/UBR score at the estimated smoothing parameter. When job>0, it gives a vector of GCV/GML/UBR functions evaluated at regular grid points.
`df`	equavilent degree of freedom.
`nobs`	length-of-y, number of observations.
`nnull`	$dim(H_0)$, number of bases.
`s`, `qraux`, `jpvt`	QR decomposition of S=FR, as from Linpack 'dqrdc'.
`q`	first $dim(H_0)$ columns gives $F^T Q F_1$, and its bottom-right corner gives tridiagonalization of $F_2^T Q F_2$.

Author(s)

Chunlei Ke `<chunlei_ke@yahoo.com>` and Yuedong Wang `<yuedong@pstat.ucsb.edu>`

References

Wahba, G. (1990). Spline Models for Observational Data. SIAM, Vol. 59.

Wang, Y. (1997). GRKPACK: Fitting Smoothing Spline ANOVA Models for Exponential Families. Communications in Statistics: Simulation and Computation, 24: 1037-1059.

See Also

`dsidr`, `dmudr`, `gdmudr`, `ssr`

hat.ssr *Extract the Hat Matrix from a ssr Object*

Description

Calculate the hat matrix for a `ssr` object.

Usage

```
hat.ssr(ssr.obj)
```

Arguments

`ssr.obj`	a fitted ssr object.

Details

The hat matrix may be used for diagnosis. Note that the full name hat.ssr shoud be used since the function hat already exist.

Value

returned is the hat (influence, smoother) matrix.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

References

Eubank, R. L. (1984). The Hat Matrix for Smoothing Splines. Statistics and Probability Letters, 2:9-14.

Eubank, R. L. (1985). Diagnostics for Smoothing Splines. Journal of the Royal Statistical Society B. 47: 332-341.

Wahba, G. (1990). Spline Models for Observational Data. SIAM, Vol. 59.

See Also

`ssr`

Examples

```
## Not run: library(MASS)
## Not run: fit1<- ssr(accel~times, data=mcycle, scale=T, rk=cubic(times))
## Not run: h <- hat.ssr(fit1)
```

horm.cort — *Hormone Measurements of Cortisol*

Description

The `horm.cort` data frame has 425 rows and 4 columns of data representing measurement of cortisol on 36 individuals.

Usage

```
data(horm.cort)
```

Format

The data frame contains the following columns:

ID a vector of index indicating individuals on whom measures were made.

time a numeric vector of time points of every 2 hours in 24 hours. The time is scaled into [0, 1].

type a vector of character strings identifying the groups, "normal", "depressed", or "cushing", which the individuals belong to.

conc cortisol concentration measurements in $log10$ scale.

Details

Blood samples were collected every 2 hours for 24 hours from three group of healthy normal volunteers and volunteers with depresession and suchsing syndrome. They were analyzed for parameters that measure hormones of the hypothalamic-pituitary axix. Human circadian thythm is one of the research objective. In this data set, only measurements of cortisol concetration were included.

Source

This data set was extracted from a stress study conducted in the medical center of the University of Michigan.

References

Wang, Y. and Brown, M. B. (1996). A Flexible Model for Human Circadian Rhythms. Biometrics 52, 588-596.

Yuedong Wang, Chunlei Ke and Morton B. Brown (2003), Shape Invariant Modelling of Circadian Rhythms with Random Effects and Smoothing Spline ANOVA Decompositions. Biometrics, 59:804-812.

`ident` *Scaling a Vector*

Description

Perform standarization of vector relative to another.

Usage

```
ident(x, y = x)
```

Arguments

`x`	a numeric vector, matrix or data frame
`y`	an optional numeric vector, matrix or data frame. Default is `x`.

Details

Scale `y` based on `x` component by component. For example, if both are a matrix, $y[, i] = (y[,] - min(x[, i]))/(max(x[, i]) - min(x[, i]))$.

Value

a scaled `y`.

`inc` *Fit a Monotone Curve Using a Cubic Spline*

Description

Return a spline fit of a increasing curve.

Usage

```
inc(y, x, spar = "v", limnla = c(-6, 0), grid = x, prec = 1e-06, maxit = 50, verbose = F)
```

Arguments

`y`	a vecetor, used as the response data
`x`	a vector, used as the covariate. Assume an increasing relationshop of `y` on `x`
`spar`	a character string specifying a method for choosing the smoothing parameter. "v", "m" and "u" represent GCV, GML and UBR respectively. Default is "v" for GCV

`limnla`	a vector of length one or two, specifying a search range for log10(n*lambda), where lambda is the smoothing parameter and n is the sample size. If it is a single value, the smoothing parameter will be fixed at this value.
`grid`	a vector of `x` used to assess convergence. Default is `x`
`prec`	a numeric value used to assess convergence. Default is 1e-6
`maxit`	an integer represeenting the maximum iterations. Default is 50.
`verbose`	an optional logical value. If 'TRUE', detailed iteration results are displayed. Default is "FALSE"

Details

This function is to fit a increasing fucntion to the data. The monotone function is expressed as integral of an unknown function that a cubic spline is used to estimate.

Value

a split fit together with the convergence information

Author(s)

Yuedong Wang <yuedong@pstat.ucsb.edu> and Chunlei Ke <chunlei_ke@yahoo.com>

See Also

`ssr`

`intervals.nnr` *Calculate Predictions and Approximate Posterior Standard Deviations for Spline Estimates From a nnr Object*

Description

Approximate posterior standard deviations are calculated for the spline estimate of nonparametric functions from a `nnr` object, based on which approximate Bayesian confidence intervals may be constructed.

Usage

```
## S3 method for class 'nnr'
intervals(object,level=0.95, newdata=NULL, terms, pstd=TRUE, ...)
```

Arguments

- `object` — an object inheriting from class `nnr`, representing a nonlinear nonparametric regression model fit.
- `newdata` — a data frame on which the fitted spline estimates are to be evaluated. Only those predictors, referred in `func` of `nnr` fitting, have to be present. The variable names of the data frame should correspond to the function(s)' arguments appearing in the opion func= of nnr. Default is NULL, where predictions are made at the same values used to fit the object.
- `terms` — an optional named list of vectors or matrices containing 0's and 1's collecting one or several combinations of the components of spline estimates in the fitted snr object. The length and names of the list shall match those of the unknown functions appearing in the 'snr' fit object. For the case of a single function, a vector of 0's and 1's can also be accepted. A value "1" at a particular position means that the component at that position is collected. Default is a vector of 1's, representing the overall fits of all unknown functions.
- `pstd` — an optional logic value. If TRUE (the default), the posterior standard deviations are calculated. Orelse, only the predictions are calculated. Computation required for posterior standard deviations could be intensive.
- `level` — a numeric value set as 0.95.
- `...` — other arguments, currently unused.

Details

The standard deviation returned is based on approximate Bayesian confidence intervals as formulated in Ke and Wang (2002).

Value

an object of class `bCI` is returned, which is a list of length 2. Its first element is a matrix which contains predictions for combinations specified by `terms`, and second element is a matrix which contains corresponding posterior standard deviations.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

References

Ke, C. and Wang, Y. (2002). Nonlinear Nonparametric Regression Models. Submitted.

See Also

`nnr`, `plot.bCI`

Examples

```
## Not run:
## fit a generalized varying coefficient models
data(Arosa)
Arosa$csmonth <- (Arosa$month-0.5)/12
Arosa$csyear <- (Arosa$year-1)/45
ozone.fit <- nnr(thick~f1(csyear)+exp(f2(csyear))*f3(csmonth),
        func=list(f1(x)~list(~I(x-.5),cubic(x)), f2(x)~list(~I(x-.5)-1,cubic(x)),
        f3(x)~list(~sin(2*pi*x)+cos(2*pi*x)-1,lspline(x,type="sine0"))),
     data=Arosa[Arosa$year%%2==1,], spar="m", start=list(f1=mean(thick),f2=0,f3=sin(csmonth)),
 control=list(backfit=1))

x <- seq(0,1,len=50)
u <- seq(0,1,len=50)

## calculate Bayesian confidence limits for all components of all functions
p.ozone.fit <- intervals(ozone.fit, newdata=list(csyear=x,csmonth=u),
                 terms=list(f1=matrix(c(1,1,1,1,1,0,0,0,1),nrow=3,byrow=TRUE),
                     f2=matrix(c(1,1,1,0,0,1),nrow=3,byrow=TRUE),
                            f3=matrix(c(1,1,1,1,1,0,0,0,1),nrow=3,byrow=TRUE)))
plot(p.ozone.fit, x.val=x)

## End(Not run)
```

`intervals.slm` *Calculate Predictions and Posterior Standard Deviations of Spline Estimates From a slm Object*

Description

Provide a way to calculate approximate posterior standard deviations and fitted values at any specified values for any combinations of elements of the spline estimate of nonparametric functions from a `slm` object, based on which approximate Bayesian confidence intervals may be constructed.

Usage

```
## S3 method for class 'slm'
intervals(object, level=0.95, newdata=NULL, terms, pstd=TRUE, ...)
```

Arguments

`object`	an object inheriting from class "slm", representing a semi-parametric nonlinear regression model fit.
`level`	set as 0.95, unused currently
`newdata`	an optional data frame on which the fitted spline estimate is to be evaluated.

`terms` an optional vector of 0's and 1's collecting a combination of components, or a matrix of 0's and 1's collecting several combinations of components, in a fitted ssr object. All components include bases on the right side of ~ in the formula and reproducing kernels in the rk list. Note that the first component is usually a constant function if it is not specifically excluded in the formula. A value "1" at a particular position means that the component at that position is collected. Default is a vector of 1's, representing the overall fit.

`pstd` an optional logic value. If TRUE (the default), the posterior standard deviations are calculated. Orelse, only the predictions are calculated. Computation required for posterior standard deviations could be intensive.

`...` other arguments, currently unused.

Details

The standard deviation returned is based on approximate Bayesian confidence intervals as formulated in Wang (1998).

Value

an object of class `bCI` is returned, which is a list of length 2. Its first element is a matrix which contains predictions for combinations specified by `terms`, and second element is a matrix which contains corresponding posterior standard deviations.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

References

Wang, Y. (1998). Mixed-effects smoothing spline ANOVA. Journal of the Royal Statistical Society, Series B 60, 159-174.

See Also

`slm`, `plot.bCI`, `predict.ssr`

Examples

```
## Not run:
data(dog)
# fit a SLM model with random effects for dogs
dog.fit<-slm(y~group*time, rk=list(cubic(time), shrink1(group),
    rk.prod(kron(time-0.5),shrink1(group)),rk.prod(cubic(time),
    shrink1(group))), random=list(dog=~1), data=dog)

intervals(dog.fit)

## End(Not run)
```

intervals.snm *Calculate Predictions and Approximate Posterior Standard Deviations for Spline Estimate From a snm Object*

Description

Provide a way to calculate approximate posterior standard deviations and fitted values at any specified values for any combinations of elements of the spline estimate of nonparametric functions from a snm object, based on which approximate Bayesian confidence intervals may be constructed.

Usage

```
## S3 method for class 'snm'
intervals(object,level=0.95,newdata=NULL, terms, pstd=TRUE, ...)
```

Arguments

`object`	an object inheriting from class snm, representing a semi-parametric nonlinear mixed effects model fit.
`newdata`	a data frame on which the fitted spline estimates are to be evaluated. Only those predictors, referred in 'func' of 'snm' fitting, have to be present. The variable names of the data frame should correspond to the function(s)' arguments appearing in the opion func= of snm. Default is NULL, where predictions are made at the same values used to fit the object.
`terms`	an optional vector of 0's and 1's collecting a combination of components, or a matrix of 0's and 1's collecting several combinations of components of spline estimates in a fitted snm object. Note that in the cases of multiple functions, the order of all componets is collection of base functions for all functions followed by RK's. A value "1" at a particular position means that the component at that position is collected. Default is a vector of 1's, representing the overall fit.
`pstd`	an optional logic value. If TRUE (the default), approximate posterior standard deviations are calculated. Orelse, only the predictions are calculated. Computation required for posterior standard deviations could be intensive.
`level`	a numeric value set as 0.95.
`...`	other arguments, currently unused.

Details

The standard deviation returned is based on approximate Bayesian confidence intervals as formulated in Ke and Wang (2001).

Value

an object of class `bCI` is returned, which is a list of length 2. Its first element is a matrix which contains predictions for combinations specified by "terms", and second element is a matrix which contains corresponding posterior standard deviations.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>.

References

Ke, C. and Wang, Y. (2001). Semi-parametric Nonlinear Mixed Effects Models and Their Applications. JASA 96:1272-1298.

See Also

snm, plot.bCI, predict.ssr

Examples

```
## Not run:
data(horm.cort)

## extract normal dubjects
cort.nor<- horm.cort[horm.cort$type=="normal",]

## fit a self-modelling model with random effects
cort.fit<- snm(conc~b1+exp(b2)*f(time-alogit(b3)),
  func=f(u)~list(periodic(u)), fixed=list(b1~1),
  random=pdDiag(b1+b2+b3~1), data=cort.nor,
  groups= ~ID,start=mean(cort.nor$conc))

## note the variable name of newdata
intervals(cort.fit, newdata=data.frame(u=seq(0,1,len=50)))

## End(Not run)
```

`intervals.snr` *Calculate Predictions and Approximate Posterior Standard Deviations for Spline Estimates From a snr Object*

Description

Approximate posterior standard deviations are calculated for the spline estimate of nonparametric functions from a snr object, based on which approximate Bayesian confidence intervals may be constructed.

Usage

```
## S3 method for class 'snr'
intervals(object, level=0.95,newdata=NULL, terms=list(), pstd=TRUE,  ...)
```

Arguments

- `object` an object inheriting from class 'snr', representing a semi-parametric nonlinear regression model fit.
- `level` set as 0.95, unused currently
- `newdata` a data frame on which the fitted spline estimates are to be evaluated. Only those predictors, referred in 'func' of 'snr' fitting, have to be present. The variable names of the data frame should correspond to the function(s)' arguments appearing in the opion func= of snr. Default is NULL, where predictions are made at the same values used to fit the object.
- `terms` an optional named list of vectors or matrices containing 0's and 1's collecting one or several combinations of the components of spline estimates in the fitted snr object. The length and names of the list shall match those of the unknown functions appearing in the 'snr' fit object. For the case of a single function, a vector of 0's and 1's can also be accepted. A value "1" at a particular position means that the component at that position is collected. Default is a vector of 1's, representing the overall fits of all unknown functions.
- `pstd` an optional logic value. If TRUE (the default), the posterior standard deviations are calculated. Orelse, only the predictions are calculated. Computation required for posterior standard deviations could be intensive.
- `...` other arguments, currently unused.

Details

The standard deviation returned is based on approximate Bayesian confidence intervals as formulated in Ke (2000).

Value

a named list of objects of class "bCI" is returned, each component of which is a list of length 2. Within each component, the first element is a matrix which contains predictions for combinations specified by "terms", and the second element is a matrix which contains corresponding posterior standard deviations.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

References

Ke, C. (2000). Semi-parametric Nonlinear Regression and Mixed Effects Models. PhD thesis, University of California, Santa Barbara.

See Also

`snr`, `plot.bCI`, `predict.ssr`

Examples

```
## Not run:
data(CO2)
options(contrasts=rep("contr.treatment", 2))

## get start values
co2.fit1 <- nlme(uptake~exp(a1)*(1-exp(-exp(a2)*(conc-a3))),
                 fixed=list(a1+a2~Type*Treatment,a3~1),
                 random=a1~1, groups=~Plant,
                 start=c(log(30),0,0,0,log(0.01),0,0,0,50),
                 data=CO2)

M <- model.matrix(~Type*Treatment, data=CO2)[,-1]

## fit a SNR model
co2.fit2 <- snr(uptake~exp(a1)*f(exp(a2)*(conc-a3)),
                func=f(u)~list(~I(1-exp(-u))-1,lspline(u, type="exp")),
                params=list(a1~M-1, a3~1, a2~Type*Treatment),
                start=list(params=co2.fit1$coe$fixed[c(2:4,9,5:8)]), data=CO2)

p.co2.fit2<- intervals(co2.fit2, newdata=data.frame(u=seq(0,10,len=50)))

## End(Not run)
```

---

kron *Calculate reproducing kernels for one-dimensional space*

Description

Return a matrix evaluating reproducing kernels for the one-dimensional space usually spanned by a vector

Usage

```
kron(x,y=x)
```

Arguments

x	a vector or a list of numerical values which spans the one-dimensional space.
y	a vector or a list of numerical values. Default is x.

Value

a matrix with the numbers of row and column equal to the length of x and y respectively. The [i, j] element is the reproducing kernel evaluated at the ith element of x and jth element of y.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

See Also

kronecker,ssr

Examples

```
## Not run:
x<-runif(10)
kron(x)

## End(Not run)
```

---

lspline — *Calculate Reproducing Kernels for Some L-splines*

Description

Return a matrix evaluating reproducing kernels for some L-splines at observed points.

Usage

```
lspline(x,y=x, type="exp", ...)
```

Arguments

x	a numeric vector on which reproducing kerenls are evaluated.
y	an optional vector, specifying the second argument of reproducing kernels. Default is x.
type	a string indicating the type of L-splines. Available options are "exp", "logit","sine", "sine1", and "linSinCos". Default is "exp".
...	other arguments needed.

Details

Denote L as the differential oprator, H_0 as the null (kernel) space. The available kernels correspond to the following L:

- exp: $L = rD + D^2$, $H_0 = span\{1, exp(-rx)\}$. $r > 0$, default to be 1;
- logit: $L = D - 1/(1 + e^t)$, $H_0 = span\{e^t/(1 + e^t)\}$;
- sine0: $L = D^2 + (2\pi)^2$, $H_0 = span\{sin(2\pi x), cos(2\pi x)\}$;
- sine1: $L = D(D^2 + (2\pi)^2)$, $H_0 = span\{1, sin(2\pi x), cos(2\pi x)\}$;
- linSinCos: $L = D^4 + D^2$, $H_0 = spac\{1, x, sin(x), cos(x)\}$.

Value

a matrix with the numbers of row and column equal to the lengths of x and y respectively. The [i, j] element is the reproducing kernel evaluated at (x[i], y[j]).

Author(s)

Chunlei Ke `<chunlei_ke@yahoo.com>` and Yuedong Wang `<yuedong@pstat.ucsb.edu>`

References

Wahba, G. (1990). Spline Models for Observational Data. SIAM, Vol. 59.

Heckman, N and Ramsay, J. O. (2000). Penalised regression with model-based penalties. To appear in Canadian Journal of Statisitcs.

See Also

`ssr`

Examples

```
## Not run:
x<- seq(0,1, len=20)
lspline(x, type="exp", r=1.5)

## End(Not run)
```

`nnr` *Nonlinear Non-parametric Regression*

Description

Fit a nonlinear nonparametric regression models with spline smoothing based on extended Gauss-Newton/Newton-Raphson and backfitting.

Usage

```
nnr(formula, func, spar="v", data=list(),
    start=list(),verbose=FALSE,  control=list())
```

Arguments

`formula`	a model formula, with the response on the left of a ~ operator and on the right an expression representing the mean function with a nonparametric function appearing with a symbol, e.g. f.

`func`	a required formula specifying the spline components necessary to estimate the non-parametric function. On the left of a ~ operator is the unknow function symbol as well as its arguments, while the right side is a list of two components, an optional `nb` and a required `rk`. `nb` and `rk` are similar to `formula` and `rk` in `ssr`. A missing `nb` denotes an empty null space.
`spar`	a character string specifying a method for choosing the smoothing parameter. "v", "m" and "u" represent GCV, GML and UBR respectively. Default is "v" for GCV.
`data`	an optional data frame.
`start`	a list of vectors or expressions which input inital values for the unknown functions. If expressions, the argument(s) inside should be the same as in `func`. The length of `start` should be the same as the number of unknown functions. If named, the names of the list should match those in "func". If not named, the order of the list is taken as that appearing in "func".
`verbose`	an optional logical numerical value. If `TRUE`, information on the evolution of the iterative algorithm is printed. Default is `FALSE`.
`control`	an optional list of control values to be used. See nnr.control for details.

Details

A nonlinear nonparametric model is fitted using the algorithms developed in Ke and Wang (2002).

Value

an object of class `nnr` is returned, containing fitted values, fitted function values as well as other information used to assess the estimate.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>.

References

Ke, C. and Wang, Y. (2002). Nonlinear Nonparametric Regression Models. Submitted.

See Also

`nnr.control`, `ssr`, `print.nnr`, `summary.nnr`, `intervals.nnr`

Examples

```
## Not run:
x<- 1:100/100
y<- exp(sin(2*pi*x))+0.3*rnorm(x)
fit<- nnr(y~exp(f(x)), func=list(f(u)~list(~u, cubic(u))), start=list(0))

## fit a generalized varying coefficient models
data(Arosa)
Arosa$csmonth <- (Arosa$month-0.5)/12
```

```
Arosa$csyear <- (Arosa$year-1)/45
ozone.vc.fit <- nnr(thick~f1(csyear)+exp(f2(csyear))*f3(csmonth),
        func=list(f1(x)~list(~I(x-.5),cubic(x)), f2(x)~list(~I(x-.5)-1,cubic(x)),
        f3(x)~list(~sin(2*pi*x)+cos(2*pi*x)-1,lspline(x,type="sine0"))),
     data=Arosa[Arosa$year%%2==1,], spar="m", start=list(f1=mean(thick),f2=0,f3=sin(csmonth)),
        control=list(backfit=1))

## End(Not run)
```

`nnr.control` *Set Control Parameters for nnr*

Description

Control parameters supplied in the function call replace the defaults to be used in calling `nnr`.

Usage

```
nnr.control(job = -1, tol = 0, max.iter = 50, init = 0, limnla = c(-10,
    0), varht = NULL, theta = NULL, prec = 1e-06, maxit = 30,
    method = "NR", increment = 1e-04, backfit = 5, converg = "coef",
    toler = 0.001)
```

Arguments

`job`	an integer representing the optimization method used to find the smoothing parameter. The options are job=-1: golden-section search on (limnla(1), limnla(2)); job=0: golden-section search with interval specified automatically; job >0: regular grid search on [limnla(1), limnla(2)] with the number of grids = job + 1. Default is -1.
`tol`	tolerance for truncation used in 'dsidr'. Default is 0.0, which sets to square of machine precision.
`max.iter`	maximum number of iterations allowed for the Gauss-Newton/Newton-Raphson iteration.
`init`	an integer of 0 or 1 indicating if initial values are provided for theta. If init=1, initial values are provided using theta. Default is 0.
`limnla`	a vector of length 2, specifying a search range for the n times smoothing parameter on log10 scale. Default is (-10, 0).
`varht`	needed only when vmu="u", which gives the fixed variance in calculation of the UBR function. Default is NULL.
`theta`	If 'init=1', theta includes intial values for smoothing parameters. Default is NULL.
`prec`	precision requested for the minimum score value, where precision is the weaker of the absolute and relative precisions. Default is 1e-06.
`maxit`	maximum number of iterations allowed. Default is 30.

method	a character string specifying a method for iterations, "GN" for Gauss-Newton and "NR" for Newton-Raphson. Default is "GN".
increment	specifies a small value as increment to calcuate derivatives. Default is 1e-04.
backfit	an integer representing the number of backfitting iterations for multiple functions. Default is 5.
converg	an optional character, with possible values "coef" and "ortho", specifying the convergence criterion to be used. "coef" uses the change of estimate of parameters and functions to assess convergence, and "ortho" uses a criterion similar to the relative offset used in nls. Default is "coef".
toler	tolerance for convergence of the algorithm. Default is 0.001.

Value

returned is a list includes all re-seted control parameters.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

See Also

nnr, dsidr,dmudr

Examples

```
## Not run:
## use Newton-Raphson
nnr.control(method="NR")

## End(Not run)
```

---

paramecium *Growth of paramecium caudatum population*

Description

The 'paramecium' data frame has 25 rows and 2 columns of data from an experiment that grow paramecium caudatum

Usage

```
data(paramecium)
```

Format

The data frame contains the following columns:

day a numeric vector of days since the start of the experiment

density a numeric vector of mean number of individuals in 0.5 ml of medium of four different cultures started simultaneously

Source

Gause, G.F. (1934). The Struggle for Existence. Baltimore, MD: Williams & Wilkins.

references

Neal, D. (2004). Introduction to Population Biology. Cambridge University Press.

`periodic` *Calculate Reproducing Kernels for Periodic Polynomial Splines with Period 1*

Description

Return a matrix evaluating reproducing kernels for periodic polynomial splines at observed points.

Usage

```
periodic(s, t=s, order=2)
```

Arguments

`s`	a numeric vector.
`t`	an optional vector. Default is the same as s.
`order`	an optional integer sepcifying the order of the polynomial spline. Default is 2 for the periodic cubic spline.

Details

The general formula of the reproducing kernel is sum of an infinite series, which is approximated by taking the first 50 terms. For the case of order=2, the close form is available and used.

Value

a matrix with the numbers of row and column equal to the lengths of s and t respectively. The [i, j] element is the reproducing kernel evaluated at (s[i], t[j]).

References

Wahba, G. (1990). Spline Models for Observational Data. SIAM, Vol. 59.

Gu, C. (2001). Smoothing Spline ANOVA Modes. Chapman and Hall.

See Also

cubic, lspline

Examples

```
## Not run:
x<- seq(0, 1, len=100)
periodic(x, order=3)

## End(Not run)
```

---

plot.bCI *Bayesian Confidence Interval Plot of a Smoothing Spline Fit*

Description

Create trellis plots of a nonparametric function fit together with its (approximate) 95% Bayesian confidence intervals from a ssr/slm/snr/snm object.

Usage

```
## S3 method for class 'bCI'
plot(x, x.val=NULL, type.name=NULL, ...)
```

Arguments

`x`	an object of class "bCI" containing point evaluation of the unknown function and/or corresponding posterior standard devaitions.
`x.val`	an optional vector representing values of argument based on which the function is to evaluate.
`type.name`	an optional character vector specifying the names of fits.
`...`	options suitable for xyplot.

Details

This function is to visualize a spline fit by use of trellis graphic facility with Bayesian confidence intervals superposed. Multi-panel plots, based on xyplot, are suitable for SS ANOVA decomposition of a spline estimate.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

See Also

predict.ssr, intervals.slm, intervals.snr, intervals.snm

Examples

```
## Not run:
x<- seq(0, 1, len=100)
y<- 2*sin(2*pi*x)+rnorm(x)*0.5

fit<- ssr(y~x, cubic(x))
p.fit<- predict(fit)
plot(p.fit)
plot(p.fit,type.name="fit")

## End(Not run)
```

---

plot.ssr — *Generate Diagnostic Plots for a ssr Object*

Description

Creates a set of plots suitable for assessing a fitted smoothing spline model of class `ssr`.

Usage

```
## S3 method for class 'ssr'
plot(x, ask=FALSE, ...)
```

Arguments

`x`	a `ssr` object.
`ask`	if TRUE, plot.ssr operates in interactive mode.
`...`	Other options used for plot, currently inactive.

Details

This function is a method for the generic function plot for class `ssr`. It can be invoked by calling plot for an object of the appropriate class, or directly by calling plot.ssr regardless of the class of the object.

An appropriate x-y plot is produced to display diagnostic plots. These can be one or all of the following choices:

- Estimate of function with CIs
- Residuals against Fitted values
- Response against Fitted values
- Normal QQplot of Residuals

The first plot of estimate of function with CIs is only useful for univariate smoothing spline fits.

When ask=TRUE, rather than produce each plot sequentially, plot.ssr displays a menu listing all the plots that can be produced. If the menu is not desired but a pause between plots is still wanted one must set par(ask=TRUE) before invoking this command with argument ask=FALSE.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

See Also

plot, ssr, predict.ssr

Examples

```
## Not run: library(MASS)
## Not run: fit1<- ssr(accel~times, data=mcycle, scale=TRUE, rk=cubic(times))
## Not run: plot(fit1,ask=TRUE)
```

Polynomial *Calculate Reproducing Kernels for Polynomial Splines on [0, 1]*

Description

Return a matrix evaluating reproducing kernels for polynomial splines at observed points.

Usage

```
linear(s, t=s)
cubic(s, t=s)
quintic(s, t=s)
septic(s, t=s)
```

Arguments

s	a vector of values in [0, 1], at which the kernels are evaluated.
t	an optional vector in [0, 1]. Default is the same as s.

Details

The reproducing kernels implemented in these functions are based on Bernoulli functions with domain [0, 1].

Value

a matrix with the numbers of row and column equal to the lengths of s and t respectively. The [i, j] element is the reproducing kernel of linear, cubic, quintic, or septic spline evaluated at (s[i], t[j]).

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@ucsb.edu>

References

Wahba, G. (1990). Spline Models for Observational Data. SIAM, Vol. 59.

See Also

ssr, linear2, cubic2, quintic2, septic2

Examples

```
## Not run:
x<-seq(0, 1, len=10)
cubic(x)

## End(Not run)
```

Polynomial2 *Calculate Reproducing Kernels for Polynomial Splines on [0, T]*

Description

Return a matrix evaluating reproducing kernels for polynomial splines at observed points.

Usage

```
linear2(s, t=s)
cubic2(s, t=s)
quintic2(s, t=s)
septic2(s, t=s)
```

Arguments

s	a vector of non-negative values, at which the kernels are evaluated.
t	an optional non-negative vector. Default is the same as s.

Details

The reproducing kernels implemented in these functions are based on Green functions. The domain is [0, T], where T is a given positive number.

Value

a matrix with the numbers of row and column equal to the length of s and t respectively. The [i, j] element is the reproducing kernel of linear, cubic, quintic, or septic spline evaluated at (s[i], t[j]).

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

References

Wahba, G. (1990). Spline Models for Observational Data. SIAM, Vol. 59.

See Also

`ssr`, `linear`, `cubic`, `quintic`, `septic`

Examples

```
## Not run:
x<- seq(0, 5, len=10)
linear2(x)

## End(Not run)
```

`predict.slm` *Predict Method for Semiparametric Linear Mixed Effects Model Fits*

Description

Predicted Values on different levels of random effects with the spline fit as part of fixed effects

Usage

```
## S3 method for class 'slm'
predict(object, newdata=NULL, ...)
```

Arguments

`object`	an object inheriting from class `slm`, representing a semi-parametric linear mixed effects model fit.
`newdata`	a data frame containing the values at which predictions are required. Only those predictors, referred to in the right side of the formula in the object, need to be present by name in newdata. Default is NULL, where predictions are made at the same values used to compute the object.
`...`	other arguments, but currently unused.

Value

returned is a data.frame with columns given by the predictions at different levels and the grouping factors. Note that the smooth part of the spline fit is regarded as fixed.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>.

References

Wang, Y. (1998) Mixed Effects Smoothing Spline ANOVA. JRSS, Series B, 60:159–174.

Pinherio, J. C. and Bates, D. M. (2000) Mixed-effects Models in S and S-Plus. Springer.

See Also

slm

Examples

```
## Not run:
data(dog)

dog.fit<-slm(y~group*time, rk=list(cubic(time), shrink1(group),
    rk.prod(kron(time-0.5),shrink1(group)),rk.prod(cubic(time),
    shrink1(group))), random=list(dog=~1), data=dog)

predict(dog.fit)

## End(Not run)
```

---

predict.snm — *Predictions from a Semiparametric Nonlinear Mixed Effects Model Fit*

Description

The predictions are obtained on a semiparametric nonlinear mixed effects model object by replacing the unknown functuons and the unknown parameters with their estimates. Of note, only a population level of predictions is available.

Usage

```
## S3 method for class 'snm'
predict(object, newdata, ...)
```

Arguments

object	a fitted snm object.
newdata	a data frame containing the values at which predictions are required. Default are data used to fit the object.
...	other arguments, but currently unused.

Details

This function is a method for the generic function predict for class snm.

Value

a vector of prediction values, obtained by evaluating the model in the frame `newdata`

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

References

Wahba, G. (1990). Spline Models for Observational Data. SIAM, Vol. 59.

Ke, C. and Wang, Y. (2001). Semi-parametric Nonlinear Mixed Effects Models and Their Applications. JASA.

See Also

`snm`, `predict`

`predict.snr` *Predict Method from a Semiparametric Nonlinear Regression Model Fit*

Description

The predictions on a semiparametric nonlinear regression model object are obtained by substituting the unknwon functions together with unknown parameters with their estimates and evaluating the regression functional based on provided or default covariate values.

Usage

```
## S3 method for class 'snr'
predict(object, newdata, ...)
```

Arguments

`object`	a fitted `snr` object.
`newdata`	a data frame containing the values at which predictions are required. Default are NULL, where data used to produce the fit are to be taken.
`...`	other arguments, but currently unused.

Details

This function is a method for the generic function predict for class `snr`

Value

a vector of prediction values, obtained by evaluating the model in the frame `newdata`.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

References

Wahba, G. (1990). Spline Models for Observational Data. SIAM, Vol. 59.

Ke, C. (2000). Semi-parametric Nonlinear Regression and Mixed Effects Models. PhD thesis, University of California, Santa Barbara.

See Also

snr

---

predict.ssr *Calculate Predictions and Posterior Standard Deviations for a ssr Object*

Description

Provide a way to calculate predictions at any specified values for any combinations of elements in the fitted model. Posterior standard deviations may be used to construct Bayesian confidence intervals.

Usage

```
## S3 method for class 'ssr'
predict(object, newdata=NULL, terms, pstd=TRUE, ...)
```

Arguments

`object`	a fitted `ssr` object.
`newdata`	an optional data frame containing the values at which predictions are required. Default is NULL, where predictions are made at the same values used to compute the object. Note that if scale=T, the newdata is on the original scale before transformation.
`terms`	an optional vector of 0's and 1's collecting a combination of components, or a matrix of 0's and 1's collecting several combinations of components, in a fitted ssr object. All components include bases on the right side of ~ in the formula and reproducing kernels in the rk list. Note that the first component is usually a constant function if it is not specifically excluded in the formula. A value "1" at a particular position means that the component at that position is collected. Default is a vector of 1's, representing the overall fit.
`pstd`	an optional logic value. If TRUE (the default), the posterior standard deviations are calculated. Otherwise, only the predictions are calculated. Computation required for posterior standard deviations could be intensive.
`...`	other arguments, but currently unused.

Details

This function is a method for the generic function predict for class `ssr`. It can be used to construct Bayesian confidence intervals for any combinations of components in the fitted model.

Value

an object of class `bCI` is returned, which is a list of length 2. Its first element is a matrix which contains predictions for combinations specified by `terms`, and second element is a matrix which contains corresponding posterior standard deviations.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>.

References

Wahba, G. (1990). Spline Models for Observational Data. SIAM, Vol. 59.

See Also

`ssr`, `plot.bCI`

Examples

```
## Not run:
data(acid)

# tp.pseudo calculates the pseudo kernel
acid.fit<- ssr( ph ~ t1 + x1 + x2, rk = list(tp.pseudo(t1),
       tp.pseudo(list(x1, x2))), spar = "m", data=acid)

# extract the main effect of t1
grid <- seq(min(acid$t1),max(acid$t1),length=100)
p <- predict(acid.fit,data.frame(t1=grid,x1=0,x2=0),
     terms=c(0,1,0,0,1,0))

# extract the main effect of (x1,x2)
grid <- expand.grid(x1=seq(min(acid$x1),max(acid$x1),length=20),
     x2=seq(min(acid$x2),max(acid$x2),length=20))
p <- predict(acid.fit,data.frame(t1=0,x1=grid$x1,x2=grid$x2),
     terms=c(0,0,1,1,0,1),pstd=FALSE)

## End(Not run)
```

print.anova.ssr — *Print an anova.ssr Object*

Description

Calculate and output p-values for tests available.

Usage

```
## S3 method for class 'anova.ssr'
print(x, ...)
```

Arguments

x	an object inheriting from class anova.ssr, generally obtained by applying the anova.ssr method to an ssr object.
...	other available arguments, currently unused.

Author(s)

Chunlei Ke `<chunlei_ke@yahoo.com>` and Yuedong Wang `<yuedong@pstat.ucsb.edu>`

See Also

`anova.ssr`, `ssr`

print.nnr — *Print Values*

Description

Print the arguments of a 'nnr' object.

Usage

```
## S3 method for class 'nnr'
print(x, ...)
```

Arguments

x	a `nnr` object
...	unused argument

Details

This is a method for the function `print` for objects inheriting from class `nnr`.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

See Also

nnr

---

print.slm *Print Values*

Description

Print the arguments of a `slm` object.

Usage

```
## S3 method for class 'slm'
print(x, ...)
```

Arguments

`x`	a `slm` object
`...`	unused argument

Details

This is a method for the function `print` for objects inheriting from class `slm`.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

See Also

slm

`print.snm` *Print Values*

Description

Print the arguments of a 'snm' object.

Usage

```
## S3 method for class 'snm'
print(x, ...)
```

Arguments

`x`	a `snm` object
`...`	unused argument

Details

This is a method for the function `print` for objects inheriting from class 'snm'.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

See Also

`slm`, `print`

`print.snr` *Print Values*

Description

Print the arguments of a `snr` object.

Usage

```
## S3 method for class 'snr'
print(x, ...)
```

Arguments

`x`	a `snr` object
`...`	unused argument

Details

This is a method for the function `print` for objects inheriting from class `snr`.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

See Also

snr

---

print.ssr *Print Values*

Description

Print the arguments of a `ssr` object.

Usage

```
## S3 method for class 'ssr'
print(x, ...)
```

Arguments

x	a `ssr` object
...	unused argument

Details

This is a method for the function `print` for objects inheriting from class `ssr`.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

See Also

ssr

print.summary.nnr *Print Vales*

Description

Print the arguments of a `summary.nnr` object

Usage

```
## S3 method for class 'summary.nnr'
print(x, ...)
```

Arguments

`x`	an object of class `summary.nnr`
`...`	unused argument

Details

This is a method for the function `print` for objects inheriting from class `summary.nnr`.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

See Also

`nnr`, `summary.nnr`

print.summary.slm *Print Values*

Description

Print the arguments of a `summary.slm` object

Usage

```
## S3 method for class 'summary.slm'
print(x, ...)
```

Arguments

`x`	an object of class `summary.slm`
`...`	unused argument

Details

This is a method for the function `print` for objects inheriting from class `summary.slm`.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

See Also

`slm`, `summary.slm`

---

print.summary.snm *Print Values*

Description

Print the arguments of a `summary.snm` object

Usage

```
## S3 method for class 'summary.snm'
print(x, ...)
```

Arguments

`x`	an object of class `summary.snm`
`...`	unused argument

Details

This is a method for the function `print` for objects inheriting from class `summary.snm`.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

See Also

`snm`, `summary.snm`

print.summary.snr *Print Values*

Description

Print the arguments of a summary.snr object

Usage

```
## S3 method for class 'summary.snr'
print(x, ...)
```

Arguments

x	an object of class summary.snr
...	unused argument

Details

This is a method for the function print for objects inheriting from class summary.snr.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

See Also

snr, summary.snr

print.summary.ssr *Print Values*

Description

Print the arguments of a summary.ssr object

Usage

```
## S3 method for class 'summary.ssr'
print(x, ...)
```

Arguments

x	an object of class summary.ssr
...	unused argument.

Details

This is a method for the function `print` for objects inheriting from class `summary.ssr`.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

See Also

ssr, summary.ssr

---

rk.prod — *Calculate product of reproducing kernels*

Description

Return a matix as product of reproducing kernels

Usage

```
rk.prod(x, ...)
```

Arguments

`x`	a matrix evaluating a reproducing kernel, or a vector.
`...`	optional lists of matrices evaluating reproducing kernels or vectors. All matrics must have the same dimensions. All vectors must have the same length. The length of each vector must equal to the column and row numbers of each matrix.

Details

The product of reproducing kernels is agian a reproducing kernel. In SS ANOVA, product of reproduing kernels is often used to model interaction spline terms.

Value

a matrix as the product of reproducing kernels. If one argument is a vector, a `kron` kernel is constructed first.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

References

Gu, C. and Wahba, G. (1993a). Smoothing Spline ANOVA with component-wise Bayesian confidence intervals. Journal of Computational and Graphical Statistics 55, 353–368.

Gu, C. and Wahba, G. (1993b). Semiparametric analysis of variance with tensor product thin plate splines. JRSS B 55, 353–368.

See Also

kron, ssr

Examples

```
## Not run:
x1<- 1:10/10
x2<- runif(10)
rk.prod(cubic(x1), periodic(x2))

## End(Not run)
```

`seizure` *IEEG segments from a seizure patient*

Description

The 'seizure' data frame has 60,000 rows and 3 columns of data from an IEEG time series

Usage

```
data(seizure)
```

Details

The baseline segment contains 5-minute IEEG signal extracted at least four hours before the seizure's onset. The preseizure segment contains 5-minute IEEG signal right before a seizure's clinical onset. The sampling rate of the IEEG signal is 200 observations per second. Therefore there are 60,000 time points in each segment.

Format

The data frame contains the following columns:

t a numeric vector of the observation number

base a numeric vector of the baseline segment

preseizure a numeric vector of the segment right before a seizure

Source

D'Alessandro, M., Vachtsevanos, G., Esteller, R., Echauz, J. and Litt, B. (2001). A Generic Approach to Selecting the Optimal Feature for Epileptic Seizure Prediction. IEEE International Meeting of the Engineering in Medicine and Biology Society.

references

Qin, L. and Wang, Y. (2008), Nonparametric Spectral Analysis With Applications to Seizure Characterization Using EEG Time Series. Annals of Applied Statistics 2, 1432-1451.

Shrinkage *Calculate reproducing kernels for Stein shrinkage estimate*

Description

Return a matrix evaluating reproducing kernels for the discrete shrinkage towards zero or the mean estimate

Usage

```
shrink0(x, y=x)
shrink1(x, y=x)
```

Arguments

x	a vector of numerical values or factor indicating different levels.
y	a vector of numerical values or factor indicating different levels. Default is x.

Value

a matrix with the numbers of row and column equal to the length of x and y respectively. The $[i, j]$ element is the reproducing kernel evaluated at the ith element of x and jth element of y.

`shink0` shrinks towards zero, and `shrink1` shinks towards the mean.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

See Also

shrink0, ssr

Examples

```
## Not run:
x<-rep(1:10,2)
shrink1(x)

## End(Not run)
```

sine4p — *Calculate Reproducing Kernels for Periodic L-Splines with Period 1/2*

Description

Return a matrix evaluating reproducing kernels for periodic L-splines at observed points.

Usage

```
sine4p(s, t=s)
```

Arguments

`s`	a numeric vector.
`t`	an optional vector. Default is the same as s.

Details

The general formula of the reproducing kernel is provided in Gu (2001). The close form is not available, so an approximate based on the first 50 terms of the series is used.

Value

a matrix with the numbers of row and column equal to the lengths of s and t respectively. The [i, j] element is the reproducing kernel evaluated at (s[i], t[j]).

References

Wahba, G. (1990). Spline Models for Observational Data. SIAM, Vol. 59.

Gu, C. (2001). Smoothing Spline ANOVA Modes. Chapman and Hall.

See Also

`cubic`, `lspline`

Examples

```
## Not run:
x<- seq(0, 1, len=100)
sine4p(x)

## End(Not run)
```

slm *Fit a Semi-parametric Linear Mixed Effects Model*

Description

Returns an object of class `slm` that represents a semi-parametric linear mixed effects model fit.

Usage

```
slm(formula, rk, data=list(), random, weights=NULL,
correlation=NULL, control=list(apVar=FALSE))
```

Arguments

`formula`	a formula object, with the response on the left of a $\sim$ operator, and the bases of the null space H_0 of the non-parametric function and other terms, separated by + operators, on the right.
`rk`	a list of expressions that specify the reproducing kernels of the spline function(s), $R^1, \ldots, R^p$ for spaces $H_1, \ldots, H_p$. See the help file of ssr for more details.
`data`	An optional data frame containing the variables appearing in `formula`, `random`, `rk`, `correlation`, `weights`. By default, the variables are taken from the environment from which `slm` is called.
`random`	A named list of formulae, lists of formulae, or pdMat objects, which defines nested random effects structures. See help file of lme for more details.
`weights`	An optional `varFun` object or one-sided formula describing the within-group heteroscedasticity stucture. See the help file of `lme` for more details.
`correlation`	An optional `corStruct` object specifying the within-group correlation structure. See `lme` for more details.
`control`	an optional list of any applicable control parameters from `lme`.

Details

This generic function fits a semi-parametric linear mixed effects model (or non-parametric mixed effects models) as described in Wang (1998), but allowing for general random and correlation structures. Because the connection to a linear mixed effects model is adopted, only GML is available to choose smoothing parameters.

Value

An object of class `slm` is returned. Generic functions such as print, summary, predict and intervals have methods to show the results of the fit.

Note: output from earlier versions of `slm` shows incorrect smoothing spline parameters for SSANOVA, which is corrected in this version.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>.

References

Wang, Y. (1998) Mixed Effects Smoothing Spline ANOVA. JRSS, Series B, 60:159–174.

Pinherio, J. C. and Bates, D. M. (2000) Mixed-effects Models in S and S-Plus. Springer.

See Also

ssr, predict.slm, intervals.slm, print.slm,summary.slm

Examples

```
## Not run:
## SS ANOVA is used to model "time" and "group"
## with random intercept for "dog".
data(dog)

dog.fit<- slm(y~group*time, rk=list(cubic(time), shrink1(group),
     rk.prod(kron(time-0.5),shrink1(group)),rk.prod(cubic(time),
     shrink1(group))), random=list(dog=~1), data=dog)

## End(Not run)
```

---

snm *Fit a Semi-parametric Nonlinear Mixed-effects Model*

Description

This generic function fits a semi-paramteric nonlinear mixed-effects model in the formulation described in Ke and Wang (2001). Current version only allows linear dependence on non-parametric functions.

Usage

```
snm(formula, func, data=list(), fixed, random=fixed,
groups, start, spar="v", verbose=FALSE, method="REML", control=NULL,
correlation=NULL, weights=NULL)
```

Arguments

formula	a formula object, with the response on the left of a ~ operator, and an expression of variables, parameters and non-parametric functions on the right.

func
: a list of spline formulae each specifying the spline components necessary to estimate each non-parametric function. On the left of a ~ operator of each component is the unknow function symbol(s) as well as its arguments, while the right side is a list of two components `nb`, an optional one-side formula for representing the null space's bases, and a required `rk` structure. `nb` and `rk` are similar to `formula` and `rk` in ssr. A missing `nb` denotes an empty null space.

fixed
: a two-sided formula specifying models for the fixed effects. The syntax of `fixed` in `nlme` is adopted.

start
: a numeric vector, the same length as the number of fixed effects, supplying starting values for the fixed effects.

spar
: a character string specifying a method for choosing the smoothing parameter. "v", "m" and "u" represent GCV, GML and UBR respectively. Default is "v" for GCV.

data
: An optional data frame containing the variables appearing in `formula` , `random`, `rk`, `correlation`, `weights`. By default, the variables are taken from the environment from which `snm` is called.

random
: an optional random effects structure specifying models for the random effects. The same syntax of `random` in `nlme` is assumed.

groups
: an optional one-sided formula of the form ~g1 (single level) or ~g1/.../gQ (multiple levels of nesting), specifying the partitions of the data over which the random effects vary. g1,...,gQ must evaluate to factors in data. See nlme for details.

verbose
: an optional logical numerical value. If `TRUE`, information on the evolution of the iterative algorithm is printed. Default is `FALSE`.

method
: a character string. If 'REML' the model is fit by maximizing the restricted log-likelihood. If 'ML' the log-likelihood is maximized. Default is 'REML.

control
: a list of parameters to control the performance of the algorithm.

correlation
: an optional `corStruct` object describing the within-group correlation structure. See the documentation of corClasses for a description of the available corStruct classes. Default is NULL, corresponding to no within-in group correlations.

weights
: an optional `varFunc` object or one-sided formula describing the within-group heteroscedasticity structure. If given as a formula, it is used as the argument to `varFixed`, corresponding to fixed variance weights. See the documentation on varClasses for a description of the available varFunc classes. Defaults to NULL, corresponding to homoscesdatic within-group errors.

Value

an object of class `snm` is returned, representing a semi-parametric nonlinear mixed effects model fit. Generic functions such as print, summary, predict and intervals have methods to show the results of the fit.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>.

References

Ke, C. and Wang, Y. (2001). Semi-parametric Nonlinear Mixed Effects Models and Their Applications. JASA 96:1272-1298.

Pinheiro, J.C. and Bates, D. M. (2000). Mixed-Effects Models in S and S-PLUS. Springer.

See Also

`predict.snm`, `intervals.snm`, `snm.control`, `print.snm`,`summary.snm`

Examples

```
## Not run:
data(CO2)

options(contrasts=rep("contr.treatment", 2))
co2.fit1 <- nlme(uptake~exp(a1)*(1-exp(-exp(a2)*(conc-a3))),
                 fixed=list(a1+a2~Type*Treatment,a3~1),
                 random=a1~1, groups=~Plant,
                 start=c(log(30),0,0,0,log(0.01),0,0,0,50),
                 data=CO2)

M <- model.matrix(~Type*Treatment, data=CO2)[,-1]
co2.fit2 <- snm(uptake~exp(a1)*f(exp(a2)*(conc-a3)),
                func=f(u)~list(~I(1-exp(-u))-1,lspline(u, type="exp")),
                fixed=list(a1~M-1,a3~1,a2~Type*Treatment),
                random=list(a1~1), group=~Plant, verbose=TRUE,
                start=co2.fit1$coe$fixed[c(2:4,9,5:8)], data=CO2)

## End(Not run)
```

---

snm.control *Set Control Parameters for snm*

Description

Control parameters supplied in the function call replace the defaults to be used in calling `snm`.

Usage

```
snm.control(rkpk.control, nlme.control, prec.out=0.0005,
  maxit.out=30, converg="COEF", incDelta)
```

Arguments

`rkpk.control`	a optional list of control parameters for dsidr or dmudr to estimate the unknown functions.
`nlme.control`	a list of control parameters for the nonlinear regression step, the same as nlmeControl. Default is `list(returnObject = T, maxIter = 5)`.

`prec.out`	tolerance for convergence criterion. Default is 0.0005.
`maxit.out`	maximum number of iterations for the algorithm. Default is 30.
`converg`	an optional character, with possible values "COEF" and "PRSS", specifying the convergence criterion to be used. "COEF" uses the change of estimate of parameters and functions to assess convergence, and "PRSS" uses penalized residual sums of squares. Default is "COEF".
`incDelta`	specifies a small value as increment to calcuate derivatives. Default is 0.001.

Value

Returned is a list includes all re-seted control parameters.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

See Also

snm, dsidr, dmudr

Examples

```
## Not run:
## set maximum iteration to be 50
snm.control(maxit.out=50)

## End(Not run)
```

---

snr — *Fit A Semi-parametric Nonlinear Regression Model*

Description

This generic function fits a Semi-parametric Nonlinear Regression Model as formulated in Ke (2000).

Usage

```
snr(formula, func, params, data, start,
    spar = "v", verbose = FALSE, control = list(), correlation = NULL,
    weights = NULL)
```

Arguments

- `formula` — a model formula, with the response on the left of a ~ operator and on the right an expression representing the mean function with at least one unknown function appearing with a symbol, e.g. f. If "data" is present, all names except the nonparametric function(s) used in the formula should be defined as parameters or variables in the data frame.
- `func` — a list of spline formulae each specifying the spline components necessary to estimate each non-parametric function. On the left of a ~ operator of each component is the unknow function symbol(s) as well as its arguments, while the right side is a list of two components `nb`, an optional one-side formula for representing the null space's bases, and a required `rk` structure. `nb` and `rk` are similar to `formula` and `rk` in ssr. A missing `nb` denotes an empty null space.
- `params` — a two-sided formula specifying models for the parameters. The syntax of `params` in `gnls` is adopted. See `gnls` for details.
- `data` — an optional data frame containing the variables named in model, params, correlation and weights. By default the variables are taken from the environment from which snr is called.
- `start` — a numeric list with two components: "params=", a vector of the size of the length of the unknown parameters, providing inital values for the paramters, and "f=" a list of vectors or expressions which input inital values for the unknown functions. If the unknown functions appear linear in the model, the intial values then are not necessary.
- `spar` — a character string specifying a method for choosing the smoothing parameter. "v", "m" and "u" represent GCV, GML and UBR respectively. Default is "v" for GCV.
- `verbose` — an optional logical numerical value. If `TRUE`, information on the evolution of the iterative algorithm is printed. Default is `TRUE`.
- `control` — an optional list of control parameters. See `snr.control` for details.
- `correlation` — an optional `corStruct` as in gnls. Default is NULL, corresponding to uncorrelation.
- `weights` — an optional `varFunc` structure as in `gnls`. Default is NULL, representing equal variances.

Details

A semi-parametric regression model is generalization of self-modeling regression, nonlinear regression and smoothing spline models, including as special cases (nonlinear) partial spline models, varying coefficients models, PP regression and some other popular models. 'snr' is implemented with an alternate iterative procedures with smoothing splines to estimate the unknown functions and general nonlinear regression to estimate parameters.

Value

An object of class `snr` is returned, representing a semi-parametric nonlinear regression fit. Generic functions such as print, summary, intervals and predict have methods to show the results of the fit.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>.

References

Ke, C. (2000). Semi-parametric Nonlinear Regression and Mixed Effects Models. PhD thesis, University of California, Santa Barbara.

Pinheiro, J.C. and Bates, D. M. (2000). Mixed-Effects Models in S and S-PLUS. Springer.

Wahba, G. (1990). Spline Models for Observational Data. SIAM, Vol. 59.

See Also

intervals.snr, predict.snr, snr.control

Examples

```
## Not run:
data(CO2)
options(contrasts=rep("contr.treatment", 2))
co2.fit1 <- nlme(uptake~exp(a1)*(1-exp(-exp(a2)*(conc-a3))),
                 fixed=list(a1+a2~Type*Treatment,a3~1),
                 random=a1~1, groups=~Plant,
                 start=c(log(30),0,0,0,log(0.01),0,0,0,50),
                 data=CO2)

M <- model.matrix(~Type*Treatment, data=CO2)[,-1]

## fit a SNR model
co2.fit2 <- snr(uptake~exp(a1)*f(exp(a2)*(conc-a3)),
                func=f(u)~list(~I(1-exp(-u))-1,lspline(u, type="exp")),
                params=list(a1~M-1, a3~1, a2~Type*Treatment),
                start=list(params=co2.fit1$coe$fixed[c(2:4,9,5:8)]), data=CO2)

## End(Not run)
```

---

snr.control *Set Control Parameters for snr*

Description

Control parameters supplied in the function call replace the defaults to be used in calling snr.

Usage

```
snr.control(rkpk.control = list(job = -1, tol = 0, init = 0, limnla = c(-10,
    0), varht = NULL, theta = NULL, prec = 1e-06, maxit = 30),
    nls.control = list(returnObject = TRUE, maxIter = 5), incDelta = 0.001,
    prec.out = 0.001, maxit.out = 30, converg = "COEF", method = "GN",
    backfit = 5)
```

Arguments

rkpk.control
: a optional list of control parameters for dsidr or dmudr to estimate the unknown functions. Default is "list(job = -1, tol = 0, init = 0, limnla = c(-10, 0), varht = NULL, theta = NULL, prec = 1e-06, maxit = 30)".

nls.control
: a list of control parameters for the nonlinear regression step, the same as gnlsControl. Default is "list(returnObject = TRUE, maxIter = 5).

incDelta
: the incremental value to be used to calculate derivatives for the unknown functions. Default is 0.001

prec.out
: tolerance for convergence criterion. Default is 0.0001.

maxit.out
: maximum number of iterations for the algorithm. Default is 30.

converg
: an optional character, with possible values COEF and PRSS, specifying the convergence criterion to be used. COEF uses the change of estimate of parameters and functions to assess convergence, and PRSS uses penalized residual sums of squares. Default is COEF.

method
: an optional string of value either GN for Gauss-Newton or NR for Newton-Raphson iteration methods to estimate the unknown functions. Default is GN.

backfit
: an integer to set the number of backfitting iterations inside the loop. Default is 5

.

Value

returned is a list includes all re-seted control parameters.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>.

See Also

snr, dsidr, dmudr

Examples

```
## use Newton-Raphson iteration and only a single backfitting
## Not run:
snr.control(method="NR", backfit=1)

## End(Not run)
```

`sphere` *Calculate Pseudo Reproducing Kernels for Spherical Splines*

Description

Return a matrix evaluating reproducing kernels for splines on a sphere.

Usage

```
sphere(x, y=x, order=2)
```

Arguments

`x`	a matrix of two columns or a list of two components, representing observed latitude and longitude respectively.
`y`	a matrix of two columns or a list of two components, representing latitude and longitude respectively. Default is the same as x.
`order`	an optional integer sepcifying the order of the spherical spline. Available are 2, 3, 4, 5 and 6, with a default 2.

Details

The kernel for sperical splines is a series inconvenient to compute. This pseudo kernel is based on a topological equivalence as described in Wahba (1981), for which cases the closed form can be derived.

Value

a matrix with the numbers of row and column equal to the lengths of x and y respectively. The [i, j] element is the reproducing kernel evaluated at $(x[i,], y[j,])$ (or $((x[[1]][i], x[[2]][i]), (y[[1]][j], y[[2]][j]))$ for lists).

Author(s)

Chunlei Ke `<chunlei_ke@yahoo.com>` and Yuedong Wang `<yuedong@pstat.ucsb.edu>`

References

Wahba, G. (1981). Spline Interprolation and Smoothing on the Sphere. SIAM J. Sci. Stat.Comput., Vol. 2, No. 1, March 1981.

Wahba, G. (1990). Spline Models for Observational Data. SIAM, Vol. 59.

See Also

`periodic`

Examples

```
## Not run:
x<- seq(0, 2*pi, len=10)
y<- seq(-pi/2, pi/2, len=10)
s.ker<- sphere(cbind(x, y), order=3)

## End(Not run)
```

---

ssr *Fit a General Smoothing Spline Regression Model*

Description

Returns an object of class ssr which is a general/generalized/correlated smoothing spline fit.

Usage

```
ssr(formula, rk, data = list(), subset, weights = NULL,
 correlation = NULL, family = "gaussian", scale = FALSE,
 spar = "v", varht = NULL, limnla = c(-10, 3), control = list())
```

Arguments

`formula`	a `formula` object, with the response on the left of a ~ operator, and the bases of the null space H_0, separated by + operators, on the right. Thus it specifies the parametric part of the model that contains functions which are not penalized.
`rk`	a list of expressions specifying reproducing kernels R^1,...,R^p for H_1,...,H_p. For $p = 1$, rk may be specified with given functions. Supported functions are: "linear", "cubic", "quintic", and "septic" for linear, cubic, quintic and septic polynomial splines with "linear2", "cubic2", "quintic2", and "septic2" for another construction; "periodic" for periodic splines; "shrink0" and "shrink1" for Stein's shrink-toward-zero and shrink-toward-mean estimates; "tp" for thin-plate-splines; "lspline" for L-splines. For details on these kernels, see their help files. Users may also write their own functions.
`data`	a data frame containing the variables occurring in the formula and the `rk`. If this option is not specified, the variables should be on the search list. Missing values are not allowed.
`subset`	an optional expression indicating which subset of the rows of the data should be used in the fit. This can be a logical vector (which is replicated to have length equal to the number of observations), a numeric vector indicating which observation numbers are to be included, or a character vector of the row names to be included. All observations are included by default.
`weights`	a vector or a matrix specifying known weights for weighted smoothing, or a varFunc structure specifying a variance function structure. Its length, if a vector, or its number of columns and rows, if a matrix, must be equal to the length of responses. See documentations of nlme for availabe varFunc structures. The default is that all weights are equal.

`correlation` a corStruct object describing the correlation structure for random errors. See documentations of corClasses for availabe correlation structures. Default is NULL for no correlation.

`family` an optional string specifying the distribution family. Families supported are "binary", "binomial", "poisson", "gamma" and "gaussian" for Bernoulli, binomial, poisson, gamma and Gaussian distributions respectively. Default is "gaussian".

`scale` an optional logical value. If 'TRUE', all covariates appearing in "rk" will be scaled into interval [0, 1]. This transformation will affect predict.ssr. Default is FALSE.

`spar` a character string specifying a method for choosing the smoothing parameter. "v", "m" and "u" represent GCV, GML and UBR respectively. "u$\sim$", only used for non-Gaussian families, specifies UBR with an estimated variance. Default is "v".

`varht` needed only when 'u' is chosen for 'method', which gives the fixed variance in calculation of the UBR function. Default is NULL for 'family="gaussian"' and 1 for all other families.

`limnla` a vector of length one or two, specifying a search range for log10(n*lambda), where lambda is the smoothing parameter and n is the sample size. If it is a single value, the smoothing parameter will be fixed at this value. This option is only applicable to spline smoothing with a single smoothing parameter.

`control` a list of iteration and algorithmic constants. See ssr.control for details and default values.

Details

We adopt notations in Wahba (1990) for the general spline and smoothing spline ANOVA models. Specifically, the functional relationship between the predictor and independent variable is unknown and is assumed to be in a reproducing kernel Hilbert space H. H is decomposed into H_0 and $H_1 + ... + H_p$, where the null space H_0 is a finite dimensional space spanned by bases specified at the right side of ~ in formula, and $H_1,\ldots,H_p$ are reproducing kernel Hilbert spaces with reproducing kernel specified in the list rk.

The function is estimated from weighted penalized least square. ssr can be used to fit the general spline and smoothing spline ANOVA models (Wahba, 1990), generalized spline models (Wang, 1997) and correlated spline models (Wang, 1998). ssr can also fit partial spline model with additional parametric terms specified in the formula (Wahba, 1990).

ssr could be slow and memory intensive, especially for large sample size and/or when p is large. For fitting a cubic spline with CV or GCV estimate of the smoothing parameter, the S-Plus function `smooth.spline` is more efficient.

Components can be extracted using extractor functions predict, deviance, residuals, and summary. The output can be modified using update.

Value

an object of class `ssr` is returned. See ssr.object for details.

Note: output from earlier versions of `ssr` shows incorrect smoothing spline parameters for SSANOVA, which is corrected in this version.

Author(s)

Yuedong Wang <yuedong@pstat.ucsb.edu> and Chunlei Ke <chunlei_ke@yahoo.com>

References

Gu, C. (1989). RKPACK and its applications: Fitting smoothing spline models. Proceedings of the Statistical Computing Section, ASA, 42-51.

Gu, C. (2002). Smoothing Spline ANOVA. Spinger, New York.

Wahba, G. (1990). Spline Models for Observational Data. SIAM, Vol. 59.

Wang, Y. (1995). GRKPACK: Fitting Smoothing Spline ANOVA Models for Exponential Families. Communications in Statistics: Simulation and Computation, 24: 1037-1059.

Wang, Y. (1998) Smoothing Spline Models with Correlated Random Errors. JASA, 93:341-348.

Ke, C. and Wang, Y. (2002) ASSIST: A Suite of S-plus functions Implementing Spline smoothing Techniques. Available at: https://yuedong.faculty.pstat.ucsb.edu/

See Also

deviance.ssr, hat.ssr, plot.ssr, ssr.control, predict.ssr, print.ssr, ssr.object, summary.ssr, smooth.spline.

Examples

```
## Not run:
library(MASS)
# fitting a cubic spline
fit1<- ssr(accel~times, data=mcycle, scale=T, rk=cubic(times))
summary(fit1)

# using GML to choose the smoothing parameter
fit2<- update(fit1, spar="m")

data(acid)
## fit an additive thin plate spline
acid.fit<- ssr( ph ~ t1 + x1 + x2, rk = list(tp(t1), tp(list(x1, x2))),
        data = acid, spar = "m", scale = FALSE)
acid.fit

## End(Not run)
```

`ssr.control` *Set Control Parameters for 'ssr'*

Description

The values supplied in the function call replace the defaults and a list with all possible arguments is returned. The returned list is used as the 'control' argument to the 'ssr' function.

Usage

```
ssr.control(job=-1, tol=0.0, init=0.0, theta, prec=1e-06,
  maxit=30, tol.g=0.0, prec.g=1e-06, maxit.g=30)
```

Arguments

job	an integer representing the optimization method used to find the smoothing parameter. The options are job=-1: golden-section search on (limnla(1), limnla(2)); job=0: golden-section search with interval specified automatically; job >0: regular grid search on $[limnla(1), limnla(2)]$ with the number of grids = job + 1. Default is -1. This is only applicable to smoothing spline model with a single smoothing parameter.
tol	tolerance for truncation used in 'dsidr' or 'dmudr'. Default is 0.0 which sets to square of machine precision.
init	init=0 means no initial values are provided for smoothing parameters theta; init=1 means initial values are provided for the theta. Default is 0. This option is only applicable to smoothing spline models with multiple smoothing parameters.
theta	If init=1, theta includes intial values for smoothing parameters. Default is NULL. This is only applicable to smoothing spline models with multiple smoothing parameters.
prec	precision requested for the minimum score value in 'dmudr', where precision is the weaker of the absolute and relative precisions. Default is 1e-06. This is only applicable to smoothing spline models with multiple smoothing parameters.
maxit	maximum number of iterations allowed in 'dmudr'. Default is 30. This is only applicable to smoothing spline model with multiple smoothing parameters.
tol.g	the tolerance for elements of w's in GRKPK. Default is 0.0 which means using the machine precision. This is only applicable to generalized spline smoothing.
prec.g	precision for stopping the iteration in GRKPK. Default is 1e-06. This is only applicale to generalized spline smoothing.
maxit.g	maximum number of iterations allowed for the iteration in GRKPACK. Default is 30. This is only applicale to generalized spline smoothing.

Value

a list with components for each of the possible arguments.

See Also

ssr

Examples

```
## Not run:
# use regular grid seach method with 100 grid points
ssr.control(job=99)

## End(Not run)
```

ssr.object *A fitted ssr Object*

Description

An object returned by the ssr function, inheriting from class ssr, and representing a fitted smoothing spline model. Objects of this class have methods for the generic functions predict, print and summary.

Value

The following components must be included in a legitimate ssr object:

call	a list containing an image of the ssr call that produced the object
coef	estimated coefficients for the spline estimate
lambda	a vector representing the estimate smoothing parameters
fitted	fitted values of the unknown mean function
family	the distribution family used
cor.est	estiamted parameters, if any, in corMatrix
var.est	estiamted parameters, if any, in varFunc
s	design matrix extracted from formula
q	a list of matrices representing reproducing kernels evaluated at design points.
residuals	working residuals from the fit.
df	equivalent degrees of freedom. It is calculated as the trace of the hat matrix.
weight	a matrix representing the covariance matrix. It is NULL for iid data.
rkpk.obj	an object representing fits from dsidr/dmudr/gdsidr/gdmudr. See help files for dsidr/dmudr/gdsidr/gdmudr for more details.
scale	a logical value, specifying if scaling is used.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

See Also

ssr, predict.ssr, summary.ssr, plot.ssr, dsidr, dmudr, gdsidr, gdmudr

`star` *Magnitude of the Mira Variable R Hydrae*

Description

The `star` data frame has 1086 rows and 2 columns of data from the Mira Variable R Hydrae

Usage

```
data(star)
```

Details

This dataset contains magnitude (brightness) of the Mira variable R Hydrae during 1900-1950.

Format

The data frame contains the following columns:

time a numeric vector of the observation time in days

magnitude a numeric vector of brightness of the Mira variable R Hydrae

Source

Genton, M. G. and Hall, P. (2007). Statistical Inference for Envolving Periodic Functions, Journal of the Royal Statistical Society B 69, 643-657.

references

Yuedong Wang and Chunlei Ke (2009), Smoothing Spline Semi-parametric Nonlinear Regression Models, Journal of Computational and Graphical Statistics 18, 165-183.

`Stratford` *Daily maximum temperatures in Stratford*

Description

The `Stratford` data frame has 73 rows and 2 columns of data containing daily maximum temperatures in Stratford every five days in 1990

Usage

```
data(Stratford)
```

Details

Daily maximum temperatures from the station in Stratford, Texas, in the year 1990 were extracted. The year was divided into 73 five-day periods and measurements on the third day in each period were selected as observations.

Format

The data frame contains the following columns:

x a numeric vector representing time in a year scaled into [0,1]

y a numeric vector of the observed maximum temperature in Fahrenheit

Source

This is part of a climate dataset downloaded from the Carbon Dioxide Information Analysis Center at http://cdiac.ornl.gov/ftp/ndp070.

`summary.nnr` *Object Summaries*

Description

Summarize a `nnr` object

Usage

```
## S3 method for class 'nnr'
summary(object, ...)
```

Arguments

`object`	a fitted `nnr` object.
`...`	unused argument

Details

This is a method for the function `summary` for objects inheriting from class `nnr`. See summary for the general behavior of this function.

Author(s)

Chunlei Ke `<chunlei_ke@yahoo.com>` and Yuedong Wang `<yuedong@pstat.ucsb.edu>`

See Also

`nnr`, `print.nnr`

summary.slm *Object Summaries*

Description

Summarize a `slm` object

Usage

```
## S3 method for class 'slm'
summary(object, ...)
```

Arguments

`object`	a fitted `slm` object.
`...`	unused argument

Details

This is a method for the function `summary` for objects inheriting from class `slm`.

Author(s)

Chunlei Ke `<chunlei_ke@yahoo.com>` and Yuedong Wang `<yuedong@pstat.ucsb.edu>`

See Also

`slm`, `print.slm`

summary.snm *Object Summaries*

Description

Summarize a `snm` object

Usage

```
## S3 method for class 'snm'
summary(object, ...)
```

Arguments

`object`	a fitted 'snm' object.
`...`	unused argument

Details

This is a method for the function summary for objects inheriting from class snm.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

See Also

snm, print.snm

---

summary.snr *Object Summaries*

Description

Summarize a snr object

Usage

```
## S3 method for class 'snr'
summary(object, ...)
```

Arguments

object	a fitted snr object.
...	unused argument

Details

This is a method for the function summary for objects inheriting from class snr. See summary for the general behavior of this function.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

See Also

snr, print.snr

summary.ssr — *Summarize a ssr object*

Description

Provides a synopsis of a `ssr` object and perform tests.

Usage

```
## S3 method for class 'ssr'
summary(object, ...)
```

Arguments

`object`	a fitted `ssr` object.
`...`	unused option.

Details

This is a method for the function `summary` for objects inheriting from class `ssr`.

Author(s)

Chunlei Ke <`chunlei_ke@yahoo.com`> and Yuedong Wang <`yuedong@pstat.ucsb.edu`>

See Also

`ssr`, `print.ssr`

Thin — *Calculate Reproducing Kernels for Thin Plate Splines*

Description

Return a matrix evaluating reproducing kernels for thin plate splines at observed points.

Usage

```
tp.pseudo(s, u=s, order=2)
tp(s, u=s, order=2)
tp.linear(s, u=s)
```

Arguments

`s`	a list or matrix of observations. One component, if a list, and one column, if a matrix, contains observations on one variable. If a list, all components must be of the same length.
`u`	a list or matrix of observations. If a list, all components must be of the same length. The number of componets of the list, or the number of column of the matrix must be the same as that for s. Default is s.
`order`	an optional integer specifying the order of the thin plate spline. Default is 2. Let d be the dimension of s (and u). Then order must satisfy $2 * order - d > 0$.

Details

The pseudo kernel, which is conditional definite positive instead of definite positive, is easy to calculate, while the true reproducing kernel is complicated. Pseudo Kernels are enough to compute spline estimates, but to calcualte Bayesian confidnece intervals, the true kernel is required. For the special case of d=2 and order=2, the function tp.linear computes evaluations of the reproducing kernel of the space spanned by linear basis.

Value

a matrix with the numbers of row and column equal to the common length of componets or the number of row of s and t respectively. The [i, j] element is the pseudo, true, or linear reproducing kernel evaluated at the ith element of s and jth element of u.

Author(s)

Chunlei Ke <chunlei_ke@yahoo.com> and Yuedong Wang <yuedong@pstat.ucsb.edu>

References

Wahba, G. (1990). Spline Models for Observational Data. SIAM, Vol. 59.

Gu, C. and Wahba, G (1993). Smoothing Spline ANOVA with component-wise Bayesian confidence intervals. Journal of Computational and Graphical Statistics 55, 353–368.

See Also

`ssr`, `cubic`

Examples

```
data(acid)
## Not run: tp.pseudo(list(acid$x1, acid$x2))
## Not run: tp.pseud0(list(acid$x1, acid$x2), order=3)
```

TXtemp *Texas Historical Climate Data*

Description

The data frame TXtemp, obtained from the Carbon Dioxide Information and Analysis Center at Oak Ridge National Laboratory, has 17280 rows and 6 columns of data representing monthly temperature records for stations in Texas.

Usage

```
data(TXtemp)
```

Format

The data frame contains the following columns:

stacode a numeric vector of the unique station code formed by combining the two-digit state number [state numbers range from 1 to 48] and the four-digit station number (values range from 0008 to 9933);

lat, long numeric vectors identifying the lattitudes and longitudes of the stations in decimal degree.

year a numeric vector comprising the year for the records

month a numeric vector of values 1 to 12, represeting the month for the data

mmtemp a numeric vector of monthly average temperature in Fahrenheit scale.

Details

The data set was extracted from a large national historical climate data, containing data for 48 stations in Texas from 1961 to 1990. Monthly temperature records as well as the latitude and longitude for each station were available.

Of note, the missing values were coded as -99.99.

Source

Data are downloadable from https://ess-dive.lbl.gov/

`ultrasound` *Ultrasound imaging of the tongue shape*

Description

The 'ultrasound' data frame has 1,215 rows and 4 columns of data from an ultrasound experiment

Usage

```
data(ultrasound)
```

Details

A Russian speaker produced the consonant sequence, /gd/, in three different linguistic environments: '2words', 'cluster' and 'Schwa', with three replications for each environment. 15 points from each of 9 slices of toungue curves separated by 30 ms (milliseconds) are extracted. Therefore, in total there are 15*9*3*3=1,215 observations.

Format

The data frame contains the following columns:

height a numeric vector of toungue height in mm

length a numeric vector of toungue length in mm

time a numeric vector of time in ms

env a factor with three levels: 1 2 and 3 for environment '2words', 'cluster' and 'Schwa' respectively

Source

Phonetics-Phonology Lab of New York University.

references

Davidson, L. (2006). Comparing Tongue Shapes from Ultrasound Imaging Using Smoothing Spline Analysis of Variance. Journal of the Acoustical Society of America 120, 407-415.

USAtemp *Average Winter temperature in the United States*

Description

The USAtemp data frame has 1214 rows and 3 columns of data containing average Winter temperatures in 1981 from 1205 stations in USA.

Usage

```
data(USAtemp)
```

Format

The data frame contains the following columns:

temp a numeric vector of average temperatures (Fahrenheit)

lat a numeric vector of the latitude of a station

long a numeric vector of the longitude of a station

details

The average Winter temperatures are calculated as the averages of temperatures in December, January and February. The geological locations of 1214 stations are given in terms of longitude and latitude.

---

wesdr *Wisconsin Epidemiological Study of Diabetic Retinopathy*

Description

The wesdr data frame has 669 rows and 5 columns of data from an ongoing epidemiological study of a cohort of patients receiving their medical care in an 11-country area in southern Wisconsin.

Usage

```
data(wesdr)
```

Details

The progression of diabetic retinopathy was assessed together with a number of medical, demographic, ocular and other covariates and the retinopathy scores.

Format

This data frame contains the following columns:

num a numeric vector giving IDs for individuals.

dur a numeric vector of duration of at baseline in year.

gly a numeric vector of glycosylated hemoglobin, a measuer of hyperglycemia.

bmi a numeric vecttor of body mass index, weight in $kg/(heightinmeter)^2$.

prg a vector of 0 or 1's representing disease progression for each individual.

Source

Klein, R., Klein, B. E. K., Moss, S. E., Davis, M. D. and Demets, D. L. (1989a). The Wisconsin epidemiologic study of diabetic retinopathy. IX. Four year incidence and progression of diabetic retinopathy when age at diagnosis is less than 30 years. Arch. Ophthalmal. 107, 237-243.

Klein, R., Klein, B. E. K., Moss, S. E., Davis, M. D. and Demets, D. L. (1989b). The Wisconsin epidemiologic study of diabetic retinopathy. X. Four year incidence and progression of diabetic retinopathy when age at diagnosis is less than 30 years. Arch. Ophthalmal. 107, 244-249.

`xyplot2` *Extension of XYPLOT*

Description

Extend `xyplot` to superpose one or more symbols to each panel.

Usage

```
xyplot2(formula, data, type = "l", ...)
```

Arguments

`formula`	a two-sided formula as accepted in `xyplot`
`data`	a list of data frames. Each component shall be able to evaluate the vatiables appearing in `formula`
`type`	a vector of characters to indicate what type of plots are to draw. Default is line.
`...`	any options as accepted in `xyplot`

Value

On each panel, several plot types, the length of `data`, are superposed.

Index