Package ‘gss’

November 19, 2025

Version 2.2-10

Date 2025-11-18

Title General Smoothing Splines

Author Chong Gu [aut, cre]

Maintainer Chong Gu `<chong@purdue.edu>`

Depends R (>= 3.0.0), stats

Description A comprehensive package for structural multivariate function estimation using smoothing splines.

License GPL (>= 2)

NeedsCompilation yes

Repository CRAN

Date/Publication 2025-11-19 06:10:18 UTC

Contents

`aids` *AIDS Incubation*

Description

A data set collected by Centers for Disease Control and Prevention concerning AIDS patients who were infected with the HIV virus through blood transfusion.

Usage

```
data(aids)
```

Format

A data frame containing 295 observations on the following variables.

`incu`	Time from HIV infection to AIDS diagnosis.
`infe`	Time from HIV infection to end of data collection (July 1986).
`age`	Age at time of blood transfusion.

Source

Wang, M.-C. (1989), A semiparametric model for randomly truncated data. *Journal of the American Statistical Association*, **84**, 742–748.

`bacteriuria` *Treatment of Bacteriuria*

Description

Bacteriuria patients were randomly assigned to two treatment groups. Weekly binary indicator of bacteriuria was recorded for every patient over 4 to 16 weeks. A total of 72 patients were represented in the data, with 36 each in the two treatment groups.

Usage

```
data(bacteriuria)
```

Format

A data frame containing 820 observations on the following variables.

`id`	Identification of patients, a factor.
`trt`	Treatments 1 or 2, a factor.
`time`	Weeks after randomization.
`infect`	Binary indicator of bacteriuria (bacteria in urine).

Source

Joe, H. (1997), *Multivariate Models and Dependence Concepts*. London: Chapman and Hall.

References

Gu, C. and Ma, P. (2005), Generalized nonparametric mixed-effect models: computation and smoothing parameter selection. *Journal of Computational and Graphical Statistics*, **14**, 485–504.

`buffalo` *Buffalo Annual Snowfall*

Description

Annual snowfall accumulations in Buffalo, NY from 1910 to 1973.

Usage

```
data(buffalo)
```

Format

A vector of 63 numerical values.

Source

Scott, D. W. (1985), Average shifted histograms: Effective nonparametric density estimators in several dimensions. *The Annals of Statistics*, **13**, 1024–1040.

`cdsscden` *Evaluating Conditional PDF, CDF, and Quantiles of Smoothing Spline Conditional Density Estimates*

Description

Evaluate conditional pdf, cdf, and quantiles of f(y1|x,y2) for smoothing spline conditional density estimates f(y|x).

Usage

```
cdsscden(object, y, x, cond, int=NULL)
cpsscden(object, q, x, cond)
cqsscden(object, p, x, cond)
```

Arguments

`object`	Object of class `"sscden"` or `"sscden1"`.
`x`	Data frame of x values on which conditional density f(y1|x,y2) is to be evaluated.
`y`	Data frame or vector of y1 points on which conditional density f(y1|x,y2) is to be evaluated.
`cond`	One row data frame of conditioning variables y2.
`q`	Vector of points on which cdf is to be evaluated.
`p`	Vector of probabilities for which quantiles are to be calculated.
`int`	Vector of normalizing constants.

Details

The arguments `x` and `y` are of the same form as the argument `newdata` in `predict.lm`, but `y` in `cdsscden` can take a vector for 1-D y1.

`cpsscden` and `cqsscden` naturally only work for 1-D y1.

Value

`cdsscden` returns a list object with the following elements.

`pdf`	Matrix or vector of conditional pdf f(y1|x,y2), with each column corresponding to a distinct x value.
`int`	Vector of normalizing constants.

`cpsscden` and `cqsscden` return a matrix or vector of conditional cdf or quantiles of f(y1|x,y2).

Note

If variables other than factors or numerical vectors are involved in `y1`, the normalizing constants can not be computed.

See Also

Fitting function `sscden` and `dsscden`.

`cdsscopu` *Evaluating 1-D Conditional PDF, CDF, and Quantiles of Copula Density Estimates*

Description

Evaluate conditional pdf, cdf, and quantiles of copula density estimates.

Usage

```
cdsscopu(object, x, cond, pos=1, int=NULL)
cpsscopu(object, q, cond, pos=1)
cqsscopu(object, p, cond, pos=1)
```

Arguments

`object`	Object of class `"sscopu"`.
`x`	Vector of points on which conditional pdf is to be evaluated.
`cond`	Value of conditioning variables.
`pos`	Position of variable of interest.
`int`	Normalizing constant.
`q`	Vector of points on which conditional cdf is to be evaluated.
`p`	Vector of probabilities for which conditional quantiles are to be calculated.

Value

A vector of conditional pdf, cdf, or quantiles.

See Also

Fitting functions `sscopu` and `sscopu2`, and `dsscopu`.

`cdssden` *Evaluating Conditional PDF, CDF, and Quantiles of Smoothing Spline Density Estimates*

Description

Evaluate conditional pdf, cdf, and quantiles for smoothing spline density estimates.

Usage

```
cdssden(object, x, cond, int=NULL)
cpssden(object, q, cond)
cqssden(object, p, cond)
```

Arguments

`object`	Object of class `"ssden"`.
`x`	Data frame or vector of points on which conditional density is to be evaluated.
`cond`	One row data frame of conditioning variables.
`int`	Normalizing constant.
`q`	Vector of points on which conditional cdf is to be evaluated.
`p`	Vector of probabilities for which conditional quantiles are to be calculated.

Details

The argument `x` in `cdssden` is of the same form as the argument `newdata` in `predict.lm`, but can take a vector for 1-D conditional densities.

`cpssden` and `cqssden` naturally only work for 1-D conditional densities of a numerical variable.

Value

`cdssden` returns a list object with the following elements.

`pdf`	Vector of conditional pdf.
`int`	Normalizing constant.

`cpssden` and `cqssden` return a vector of conditional cdf or quantiles.

Note

If variables other than factors or numerical vectors are involved in `x`, the normalizing constant can not be computed.

See Also

Fitting function `ssden` and `dssden`.

`clim` *Average Temperatures During December 1980 Through February 1981*

Description

Average temperatures at 690 weather stations during December 1980 through February 1981.

Usage

```
data(clim)
```

Format

A data frame containing 690 observations on the following variables.

`temp`	Average temperature, in Celsius.
`geog`	Geographic location (latitude,longitude), in degrees, as a matrix.

Source

This is reformulated from the data frame `climate` in the R package `assist` by Yuedong Wang and Chunlei Ke.

`ColoCan` *Colorectal Cancer Mortality Rate in Indiana Counties*

Description

County-wise death counts of colorectal cancer patients in Indiana during years 2000 through 2004.

Usage

```
data(ColoCan)
```

Format

A data frame containing 184 observations on the following variables.

`event`	Death counts.
`pop`	Population from Census 2000.
`sex`	Gender of population.
`wrt`	Proportion of Whites.
`brt`	Proportion of Blacks.
`ort`	Proportion of other minorities.
`lat`	Latitude.
`lon`	Longitude.
`geog`	Geographic location, derived from `lat` and `lon`.
`scrn`	Colorectal cancer screening rate.
`name`	County name.

Details

`geog` was generated from `lat` and `lon` using the code given in the example section.

Source

Dr. Tonglin Zhang.

References

Zhang, T. and Lin, G. (2009), Cluster detection based on spatial associations and iterated residuals in generalized linear mixed models. *Biometrics*, **65**, 353–360.

Examples

```
## Converting latitude and longitude to x-y coordinates
## The 49th county is Marion, where Indianapolis is located.
## Not run: ltln2xy <- function(latlon,latlon0) {
  lat <- latlon[,1]*pi/180; lon <- latlon[,2]*pi/180
  lt0 <- latlon0[1]*pi/180; ln0 <- latlon0[2]*pi/180
  x <- cos(lt0)*sin(lon-ln0); y <- sin(lat-lt0)
  cbind(x,y)
}
data(ColoCan)
latlon <- as.matrix(ColoCan[,c("lat","lon")])
ltln2xy(latlon,latlon[49,])
## Clean up
rm(ltln2xy,ColoCan,latlon)
## End(Not run)
```

`DiaRet` *Diabetic Retinopathy*

Description

Time to blindness of 197 diabetic retinopathy patients who received a laser treatment in one eye.

Usage

```
data(DiaRet)
```

Format

A data frame containing 197 observations on the following variables.

`id`	Patient ID.
`time1`	Follow-up time of left eye.
`time2`	Follow-up time of right eye.
`status1`	Censoring indicator of left eye.
`status2`	Censoring indicator of right eye.
`trt1`	Treatment indicator of left eye.
`trt2`	Treatment indicator of right eye.
`type`	Type of diabetes.
`age`	Age of patient at diagnosis.
`time.t`	Follow-up time of treated eye.
`time.u`	Follow-up time of untreated eye.
`status.t`	Censoring indicator of treated eye.
`status.u`	Censoring indicator of untreated eye.

Source

This is reformatted from the data frame `diabetes` in the R package `timereg` by Thomas H. Scheike.

References

Huster, W.J., Brookmeyer, R., and Self, S.G. (1989), Modelling paired survival data with covariates. *Biometrics*, **45**, 145–56.

`dsscden` *Evaluating PDF, CDF, and Quantiles of Smoothing Spline Conditional Density Estimates*

Description

Evaluate pdf, cdf, and quantiles for smoothing spline conditional density estimates.

Usage

```
dsscden(object, y, x)
psscden(object, q, x)
qsscden(object, p, x)
d.sscden(object, x, y)
d.sscden1(object, x, y, scale=TRUE)
```

Arguments

`object`	Object of class `"sscden"` or `"sscden1"`.
`x`	Data frame of x values on which conditional density f(y|x) is to be evaluated.
`y`	Data frame or vector of points on which conditional density f(y|x) is to be evaluated.
`q`	Vector of points on which cdf is to be evaluated.
`p`	Vector of probabilities for which quantiles are to be calculated.
`scale`	Flag indicating whether to use approximate scaling without quadrature.

Details

The arguments `x` and `y` are of the same form as the argument `newdata` in `predict.lm`, but `y` in `dsscden` can take a vector for 1-D responses.

`psscden` and `qsscden` naturally only work for 1-D responses.

Value

A matrix or vector of pdf, cdf, or quantiles of f(y|x), with each column corresponding to a distinct x value.

See Also

Fitting function `sscden` and `cdsscden`.

`dsscopu` *Evaluating Copula Density Estimates*

Description

Evaluate copula density estimates.

Usage

```
dsscopu(object, x, copu=TRUE)
```

Arguments

`object`	Object of class `"sscopu"`.
`x`	Vector or matrix of point(s) on which copula density is to be evaluated.
`copu`	Flag indicating whether to apply copularization.

Value

A vector of copula density values.

See Also

Fitting functions `sscopu` and `sscopu2`.

`dssden` *Evaluating PDF, CDF, and Quantiles of Smoothing Spline Density Estimates*

Description

Evaluate pdf, cdf, and quantiles for smoothing spline density estimates.

Usage

```
dssden(object, x)
pssden(object, q)
qssden(object, p)
d.ssden(object, x)
d.ssden1(object, x)
```

Arguments

`object`	Object of class `"ssden"`.
`x`	Data frame or vector of points on which density is to be evaluated.
`q`	Vector of points on which cdf is to be evaluated.
`p`	Vector of probabilities for which quantiles are to be calculated.

Details

The argument `x` in `dssden` is of the same form as the argument `newdata` in `predict.lm`, but can take a vector for 1-D densities.

`pssden` and `qssden` naturally only work for 1-D densities.

Value

A vector of pdf, cdf, or quantiles.

See Also

Fitting function `ssden` and `cdssden`.

`esc` *Embryonic Stem Cell from Mouse*

Description

Data concerning mouse embryonic stem cell gene expression and transcription factor association strength.

Usage

```
data(esc)
```

Format

A data frame containing 1027 genes with the following variables.

`y1`	Gene expression after 4 days.
`y2`	Gene expression after 8 days.
`y3`	Gene expression after 14 days.
`klf4`	Score of TFAS with KLF4.
`nanog`	Score of TFAS with NANOG.
`oct4`	Score of TFAS with OCT4.
`sox2`	Score of TFAS with SOX2.
`clusterID`	Cluster identification.

References

Cai, J., Xie, D., Fan, Z., Chipperfield, H., Marden, J., Wong, W. H., and Zhong, S. (2010), Modeling co-expression across species for complex traits: insights to the difference of human and mouse embryonic stem cells. *PLoS Computational Biology*, **6**, e1000707.

Ouyang, Z., Zhou, Q., and Wong, W. H. (2009), chip-seq of transcription factors predicts absolute and differential gene expression in embryonic stem cells. *Proceedings of the National Academy of Sciences of USA*, **106**, 21521–21526.

`eyetrack` *Eyesight Fixation in Eyetracking Experiments*

Description

Eyesight Fixation during some eyetracking experiments in liguistic studies.

Usage

```
data(eyetrack)
```

Format

A data frame containing 13891 observations on the following variables.

`time`	Time, in ms.
`color`	Binary indicator, 1 if eyesight fixed on target or color competitor, a factor.
`object`	Binary indicator, 1 if eyesight fixed on target or object competitor, a factor.
`id`	Identification of homogeneous sessions, a factor.
`cnt`	Multiplicity count.

Source

Dr. Anouschka Foltz.

References

Gu, C. and Ma, P. (2011), Nonparametric regression with cross-classified responses. Manuscript.

`fitted.ssanova` *Fitted Values and Residuals from Smoothing Spline ANOVA Fits*

Description

Methods for extracting fitted values and residuals from smoothing spline ANOVA fits.

Usage

```
## S3 method for class 'ssanova'
fitted(object, ...)
## S3 method for class 'ssanova'
residuals(object, ...)

## S3 method for class 'gssanova'
fitted(object, ...)
## S3 method for class 'gssanova'
residuals(object, type="working", ...)
```

Arguments

`object`	Object of class `"ssanova"` or `"gssanova"`.
`type`	Type of residuals desired, with two alternatives `"working"` (default) or `"deviance"`.
`...`	Ignored.

Details

The fitted values for `"gssanova"` objects are on the link scale, so are the `"working"` residuals.

`gastric` *Gastric Cancer Data*

Description

Survival of gastric cancer patients under chemotherapy and chemotherapy-radiotherapy combination.

Usage

```
data(gastric)
```

Format

A data frame containing 90 observations on the following variables.

`futime`	Follow-up time, in days.
`status`	Censoring status.
`trt`	Factor indicating the treatments: 1 – chemothrapy, 2 – combination.

Source

Moreau, T., O'Quigley, J., and Mesbah, M. (1985), A global goodness-of-fit statistic for the proportional hazards model. *Applied Statistics*, **34**, 212-218.

`gauss.quad` *Generating Gauss-Legendre Quadrature*

Description

Generate Gauss-Legendre quadratures using the FORTRAN routine `gaussq.f` found on NETLIB.

Usage

```
gauss.quad(size, interval)
```

Arguments

`size`	Size of quadrature.
`interval`	Interval to be covered.

Value

`gauss.quad` returns a list object with the following elements.

`pt`	Quadrature nodes.
`wt`	Quadrature weights.

`gssanova` *Fitting Smoothing Spline ANOVA Models with Non-Gaussian Responses*

Description

Fit smoothing spline ANOVA models in non-Gaussian regression. The symbolic model specification via `formula` follows the same rules as in `lm` and `glm`.

Usage

```
gssanova(formula, family, type=NULL, data=list(), weights, subset,
         offset, na.action=na.omit, partial=NULL, alpha=NULL, nu=NULL,
         id.basis=NULL, nbasis=NULL, seed=NULL, random=NULL,
         skip.iter=FALSE)
```

Arguments

`formula`	Symbolic description of the model to be fit.
`family`	Description of the error distribution. Supported are exponential families `"binomial"`, `"poisson"`, `"Gamma"`, `"inverse.gaussian"`, and `"nbinomial"`. Also supported are accelerated life model families `"weibull"`, `"lognorm"`, and `"loglogis"`. Further more, proportional odds logistic regression `"polr"` for ordinal response is also supported.
`type`	List specifying the type of spline for each variable. See `mkterm` for details.
`data`	Optional data frame containing the variables in the model.
`weights`	Optional vector of weights to be used in the fitting process.
`subset`	Optional vector specifying a subset of observations to be used in the fitting process.
`offset`	Optional offset term with known parameter 1.
`na.action`	Function which indicates what should happen when the data contain NAs.
`partial`	Optional symbolic description of parametric terms in partial spline models.
`alpha`	Tuning parameter defining cross-validation; larger values yield smoother fits. Defaults are `alpha=1` for `family="binomial"` and `alpha=1.4` otherwise.
`nu`	Inverse scale parameter in accelerated life model families. Ignored for exponential families.
`id.basis`	Index designating selected "knots".
`nbasis`	Number of "knots" to be selected. Ignored when `id.basis` is supplied.
`seed`	Seed for reproducible random selection of "knots". Ignored when `id.basis` is supplied.
`random`	Input for parametric random effects in nonparametric mixed-effect models. See `mkran` for details.
`skip.iter`	Flag indicating whether to use initial values of theta and skip theta iteration. See `ssanova` for notes on skipping theta iteration.

Details

The model specification via `formula` is intuitive. For example, `y~x1*x2` yields a model of the form

$$y = C + f_1(x1) + f_2(x2) + f_{12}(x1, x2) + e$$

with the terms denoted by `"1"`, `"x1"`, `"x2"`, and `"x1:x2"`.

The model terms are sums of unpenalized and penalized terms. Attached to every penalized term there is a smoothing parameter, and the model complexity is largely determined by the number of smoothing parameters.

Only one link is implemented for each `family`. It is the logit link for `"binomial"`, and the log link for `"poisson"`, and `"Gamma"`. For `"nbinomial"`, the working parameter is the logit of the probability p; see `NegBinomial`. For `"weibull"`, `"lognorm"`, and `"loglogis"`, it is the location parameter for the log lifetime.

The selection of smoothing parameters is through direct cross-validation. The cross-validation score used for `family="poisson"` is taken from density estimation as in Gu and Wang (2003), and those used for other families are derived following the lines of Gu and Xiang (2001).

A subset of the observations are selected as "knots." Unless specified via `id.basis` or `nbasis`, the number of "knots" q is determined by $max(30, 10n^{2/9})$, which is appropriate for the default cubic splines for numerical vectors.

Value

`gssanova` returns a list object of class `c("gssanova","ssanova")`.

The method `summary.gssanova` can be used to obtain summaries of the fits. The method `predict.ssanova` can be used to evaluate the fits at arbitrary points along with standard errors, on the link scale. The method `project.gssanova` can be used to calculate the Kullback-Leibler projection for model selection. The methods `residuals.gssanova` and `fitted.gssanova` extract the respective traits from the fits.

Responses

For `family="binomial"`, the response can be specified either as two columns of counts or as a column of sample proportions plus a column of total counts entered through the argument `weights`, as in `glm`.

For `family="nbinomial"`, the response may be specified as two columns with the second being the known sizes, or simply as a single column with the common unknown size to be estimated through the maximum likelihood.

For `family="weibull"`, `"lognorm"`, or `"loglogis"`, the response consists of three columns, with the first giving the follow-up time, the second the censoring status, and the third the left-truncation time. For data with no truncation, the third column can be omitted.

For `family="polr"`, the response should be an ordered factor.

Note

For simpler models and moderate sample sizes, the exact solution of `gssanova0` can be faster.

The results may vary from run to run. For consistency, specify `id.basis` or set `seed`.

In *gss* versions earlier than 1.0, `gssanova` was under the name `gssanova1`.

References

Gu, C. and Xiang, D. (2001), Cross validating non Gaussian data: generalized approximate cross validation revisited. *Journal of Computational and Graphical Statistics*, **10**, 581–591.

Gu, C. and Wang, J. (2003), Penalized likelihood density estimation: Direct cross-validation and scalable approximation. *Statistica Sinica*, **13**, 811–826.

Gu, C. (2013), *Smoothing Spline ANOVA Models (2nd Ed)*. New York: Springer-Verlag.

Gu, C. (2014), Smoothing Spline ANOVA Models: R Package gss. *Journal of Statistical Software*, 58(5), 1-25. URL http://www.jstatsoft.org/v58/i05/.

Examples

```
## Fit a cubic smoothing spline logistic regression model
test <- function(x)
        {.3*(1e6*(x^11*(1-x)^6)+1e4*(x^3*(1-x)^10))-2}
x <- (0:100)/100
p <- 1-1/(1+exp(test(x)))
y <- rbinom(x,3,p)
logit.fit <- gssanova(cbind(y,3-y)~x,family="binomial")
## The same fit
logit.fit1 <- gssanova(y/3~x,"binomial",weights=rep(3,101),
                       id.basis=logit.fit$id.basis)
## Obtain estimates and standard errors on a grid
est <- predict(logit.fit,data.frame(x=x),se=TRUE)
## Plot the fit and the Bayesian confidence intervals
plot(x,y/3,ylab="p")
lines(x,p,col=1)
lines(x,1-1/(1+exp(est$fit)),col=2)
lines(x,1-1/(1+exp(est$fit+1.96*est$se)),col=3)
lines(x,1-1/(1+exp(est$fit-1.96*est$se)),col=3)

## Fit a mixed-effect logistic model
data(bacteriuria)
bact.fit <- gssanova(infect~trt+time,family="binomial",data=bacteriuria,
                     id.basis=(1:820)[bacteriuria$id%in%c(3,38)],random=~1|id)
## Predict fixed effects
predict(bact.fit,data.frame(time=2:16,trt=as.factor(rep(1,15))),se=TRUE)
## Estimated random effects
bact.fit$b

## Clean up
## Not run: rm(test,x,p,y,logit.fit,logit.fit1,est,bacteriuria,bact.fit)
dev.off()
## End(Not run)
```

`gssanova0` *Fitting Smoothing Spline ANOVA Models with Non-Gaussian Responses*

Description

Fit smoothing spline ANOVA models in non-Gaussian regression. The symbolic model specification via `formula` follows the same rules as in `lm` and `glm`.

Usage

```
gssanova0(formula, family, type=NULL, data=list(), weights, subset,
          offset, na.action=na.omit, partial=NULL, method=NULL,
          varht=1, nu=NULL, prec=1e-7, maxiter=30)
```

```
gssanova1(formula, family, type=NULL, data=list(), weights, subset,
          offset, na.action=na.omit, partial=NULL, method=NULL,
          varht=1, alpha=1.4, nu=NULL, id.basis=NULL, nbasis=NULL,
          seed=NULL, random=NULL, skip.iter=FALSE)
```

Arguments

`formula`	Symbolic description of the model to be fit.
`family`	Description of the error distribution. Supported are exponential families `"binomial"`, `"poisson"`, `"Gamma"`, `"inverse.gaussian"`, and `"nbinomial"`. Also supported are accelerated life model families `"weibull"`, `"lognorm"`, and `"loglogis"`. Further more, proportional odds logistic regression `"polr"` for ordinal response is also supported.
`type`	List specifying the type of spline for each variable. See `mkterm` for details.
`data`	Optional data frame containing the variables in the model.
`weights`	Optional vector of weights to be used in the fitting process.
`subset`	Optional vector specifying a subset of observations to be used in the fitting process.
`offset`	Optional offset term with known parameter 1.
`na.action`	Function which indicates what should happen when the data contain NAs.
`partial`	Optional symbolic description of parametric terms in partial spline models.
`method`	Score used to drive the performance-oriented iteration. Supported are `method="v"` for GCV, `method="m"` for GML, and `method="u"` for Mallows' CL.
`varht`	Dispersion parameter needed for `method="u"`. Ignored when `method="v"` or `method="m"` are specified.
`nu`	Inverse scale parameter in accelerated life model families. Ignored for exponential families.
`prec`	Precision requirement for the iterations.
`maxiter`	Maximum number of iterations allowed for performance-oriented iteration, and for inner-loop multiple smoothing parameter selection when applicable.
`alpha`	Tuning parameter modifying GCV or Mallows' CL.
`id.basis`	Index designating selected "knots".
`nbasis`	Number of "knots" to be selected. Ignored when `id.basis` is supplied.
`seed`	Seed for reproducible random selection of "knots". Ignored when `id.basis` is supplied.
`random`	Input for parametric random effects in nonparametric mixed-effect models. See `mkran` for details.
`skip.iter`	Flag indicating whether to use initial values of theta and skip theta iteration. See `ssanova` for notes on skipping theta iteration.

Details

The model specification via `formula` is intuitive. For example, `y~x1*x2` yields a model of the form

$$y = C + f_1(x1) + f_2(x2) + f_{12}(x1, x2) + e$$

with the terms denoted by `"1"`, `"x1"`, `"x2"`, and `"x1:x2"`.

The model terms are sums of unpenalized and penalized terms. Attached to every penalized term there is a smoothing parameter, and the model complexity is largely determined by the number of smoothing parameters.

Only one link is implemented for each `family`. It is the logit link for `"binomial"`, and the log link for `"poisson"`, `"Gamma"`, and `"inverse.gaussian"`. For `"nbinomial"`, the working parameter is the logit of the probability p; see `NegBinomial`. For `"weibull"`, `"lognorm"`, and `"loglogis"`, it is the location parameter for the log lifetime.

The models are fitted by penalized likelihood method through the performance-oriented iteration as described in the reference. For `family="binomial"`, `"poisson"`, `"nbinomial"`, `"weibull"`, `"lognorm"`, and `"loglogis"`, the score driving the performance-oriented iteration defaults to `method="u"` with `varht=1`. For `family="Gamma"` and `"inverse.gaussian"`, the default is `method="v"`.

`gssanova0` uses the algorithm of `ssanova0` for the iterated penalized least squares problems, whereas `gssanova1` uses the algorithm of `ssanova`.

In `gssanova1`, a subset of the observations are selected as "knots." Unless specified via `id.basis` or `nbasis`, the number of "knots" q is determined by $max(30, 10n^{2/9})$, which is appropriate for the default cubic splines for numerical vectors.

Value

`gssanova0` returns a list object of class `c("gssanova0","ssanova0","gssanova")`.

`gssanova1` returns a list object of class `c("gssanova","ssanova")`.

The method `summary.gssanova0` or `summary.gssanova` can be used to obtain summaries of the fits. The method `predict.ssanova0` or `predict.ssanova` can be used to evaluate the fits at arbitrary points along with standard errors, on the link scale. The methods `residuals.gssanova` and `fitted.gssanova` extract the respective traits from the fits.

Responses

For `family="binomial"`, the response can be specified either as two columns of counts or as a column of sample proportions plus a column of total counts entered through the argument `weights`, as in `glm`.

For `family="nbinomial"`, the response may be specified as two columns with the second being the known sizes, or simply as a single column with the common unknown size to be estimated through the maximum likelihood.

For `family="weibull"`, `"lognorm"`, or `"loglogis"`, the response consists of three columns, with the first giving the follow-up time, the second the censoring status, and the third the left-truncation time. For data with no truncation, the third column can be omitted.

For `family="polr"`, the response should be an ordered factor.

Note

The direct cross-validation of `gssanova` can be more effective, and more stable for complex models.

For large sample sizes, the approximate solutions of `gssanova1` and `gssanova` can be faster than `gssanova0`.

The results from `gssanova1` may vary from run to run. For consistency, specify `id.basis` or set `seed`.

The method `project` is not implemented for `gssanova0`, nor is the mixed-effect model support through `mkran`.

In *gss* versions earlier than 1.0, `gssanova0` was under the name `gssanova`.

References

Gu, C. (1992), Cross-validating non Gaussian data. *Journal of Computational and Graphical Statistics*, **1**, 169-179.

Gu, C. (2013), *Smoothing Spline ANOVA Models (2nd Ed)*. New York: Springer-Verlag.

GU, C. (2014), Smoothing Spline ANOVA Models: R Package gss. *Journal of Statistical Software*, 58(5), 1-25. URL http://www.jstatsoft.org/v58/i05/.

Examples

```
## Fit a cubic smoothing spline logistic regression model
test <- function(x)
        {.3*(1e6*(x^11*(1-x)^6)+1e4*(x^3*(1-x)^10))-2}
x <- (0:100)/100
p <- 1-1/(1+exp(test(x)))
y <- rbinom(x,3,p)
logit.fit <- gssanova0(cbind(y,3-y)~x,family="binomial")
## The same fit
logit.fit1 <- gssanova0(y/3~x,"binomial",weights=rep(3,101))
## Obtain estimates and standard errors on a grid
est <- predict(logit.fit,data.frame(x=x),se=TRUE)
## Plot the fit and the Bayesian confidence intervals
plot(x,y/3,ylab="p")
lines(x,p,col=1)
lines(x,1-1/(1+exp(est$fit)),col=2)
lines(x,1-1/(1+exp(est$fit+1.96*est$se)),col=3)
lines(x,1-1/(1+exp(est$fit-1.96*est$se)),col=3)
## Clean up
## Not run: rm(test,x,p,y,logit.fit,logit.fit1,est)
dev.off()
## End(Not run)
```

`hzdrate.sshzd` *Evaluating Smoothing Spline Hazard Estimates*

Description

Evaluate smoothing spline hazard estimates by `sshzd`.

Usage

```
hzdrate.sshzd(object, x, se=FALSE, include=c(object$terms$labels,object$lab.p))
hzdcurve.sshzd(object, time, covariates=NULL, se=FALSE)
survexp.sshzd(object, time, covariates=NULL, start=0)
```

Arguments

`object`	Object of class `"sshzd"`.
`x`	Data frame or vector of points on which hazard is to be evaluated.
`se`	Flag indicating if standard errors are required.
`include`	List of model terms to be included in the evaluation.
`time`	Vector of time points.
`covariates`	Vector of covariate values.
`start`	Optional starting times of the intervals.

Value

For `se=FALSE`, `hzdrate.sshzd` returns a vector of hazard evaluations, and `hzdcurve.sshzd` returns a vector or columns of hazard curve(s) evaluated on `time` points at the `covariates` values. For `se=TRUE`, `hzdrate.sshzd` and `hzdcurve.sshzd` return a list consisting of the following elements.

`fit`	Vector or columns of hazard.
`se.fit`	Vector or columns of standard errors for log hazard.

`survexp.sshzd` returns a vector or columns of expected survivals based on the cumulative hazards over (`start`, `time`) at the `covariates` values, which in fact are the (conditional) survival probabilities $S(time)/S(start)$.

Note

For left-truncated data, `start` must be at or after the earliest truncation point.

See Also

Fitting function `sshzd`.

`hzdrate.sshzd2d` *Evaluating 2-D Smoothing Spline Hazard Estimates*

Description

Evaluate 2-D smoothing spline hazard estimates by `sshzd2d`.

Usage

```
hzdrate.sshzd2d(object, time, covariates=NULL)
survexp.sshzd2d(object, time, covariates=NULL, job=3)
```

Arguments

`object`	Object of class `"sshzd2d"`.
`time`	Matrix or vector of time points on which hazard or survival function is to be evaluated.
`covariates`	Data frame of covariate values.
`job`	Flag indicating which survival function to evaluate.

Value

A vector of hazard or survival values.

Note

For `job=1,2`, `survexp.sshzd2d` returns marginal survival $S1(t)$ or $S2(t)$. For `job=3`, `survexp.sshzd2d` returns the 2-D survival $S(t1,t2)$.

For `hzdrate.sshzd2d` and `survexp.sshzd2d` with `job=3`, `time` should be a matrix of two columns. For `survexp.sshzd2d` with `job=1,2`, `time` should be a vector.

When `covariates` is present, its length should be either 1 or that of `time`.

See Also

Fitting function `sshzd2d`.

`LakeAcidity` *Water Acidity in Lakes*

Description

Data extracted from the Eastern Lake Survey of 1984 conducted by the United States Environmental Protection Agency, concerning 112 lakes in the Blue Ridge.

Usage

```
data(LakeAcidity)
```

Format

A data frame containing 112 observations on the following variables.

`ph`	Surface ph.
`cal`	Calcium concentration.
`lat`	Latitude.
`lon`	Longitude.
`geog`	Geographic location, derived from `lat` and `lon`

Details

`geog` was generated from `lat` and `lon` using the code given in the Example section.

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), Semiparametric analysis of variance with tensor product thin plate splines. *Journal of the Royal Statistical Society Ser. B*, **55**, 353–368.

Examples

```
## Converting latitude and longitude to x-y coordinates
## Not run: ltln2xy <- function(latlon,latlon0) {
  lat <- latlon[,1]*pi/180; lon <- latlon[,2]*pi/180
  lt0 <- latlon0[1]*pi/180; ln0 <- latlon0[2]*pi/180
  x <- cos(lt0)*sin(lon-ln0); y <- sin(lat-lt0)
  cbind(x,y)
}
data(LakeAcidity)
latlon <- as.matrix(LakeAcidity[,c("lat","lon")])
m.lat <- (min(latlon[,1])+max(latlon[,1]))/2
```

```
m.lon <- (min(latlon[,2])+max(latlon[,2]))/2
ltln2xy(latlon,c(m.lat,m.lon))
## Clean up
rm(ltln2xy,LakeAcidity,latlon,m.lat,m.lon)
## End(Not run)
```

`nlm0` *Minimizing Univariate Functions on Finite Intervals*

Description

Minimize univariate functions on finite intervals using 3-point quadratic fit, with golden-section safe-guard.

Usage

```
nlm0(fun, range, prec=1e-7)
```

Arguments

`fun`	Function to be minimized.
`range`	Interval on which the function to be minimized.
`prec`	Desired precision of the solution.

Value

`nlm0` returns a list object with the following elements.

`estimate`	Minimizer.
`minimum`	Minimum.
`evaluations`	Number of function evaluations.

`NO2` *Air Pollution and Road Traffic*

Description

A subset of 500 hourly observations collected by the Norwegian Public Roads Administration at Alnabru in Oslo, Norway, between October 2001 and August 2003.

Usage

```
data(NO2)
```

Format

A data frame containing 500 observations on the following variables.

`no2`	Concentration of NO2, on log scale.
`cars`	Traffic volume of the hour, on log scale.
`temp`	Temperature 2 meters above ground, in Celsius.
`wind`	wind speed, meters/second.
`temp2`	Temperature difference between 25 and 2 meters above ground, in Celsius.
`wind2`	Wind direction, in degrees between 0 and 360.

Source

Statlib Datasets Archive at `http://lib.stat.cmu.edu/datasets`, contributed by Magne Aldrin.

`nox` *NOx in Engine Exhaust*

Description

Data from an experiment in which a single-cylinder engine was run with ethanol to see how the NOx concentration in the exhaust depended on the compression ratio and the equivalence ratio.

Usage

```
data(nox)
```

Format

A data frame containing 88 observations on the following variables.

`nox`	NOx concentration in exhaust.
`comp`	Compression ratio.
`equi`	Equivalence ratio.

Source

Brinkman, N. D. (1981), Ethanol fuel – a single-cylinder engine study of efficiency and exhaust emissions. *SAE Transactions*, **90**, 1410–1424.

References

Cleveland, W. S. and Devlin, S. J. (1988), Locally weighted regression: An approach to regression analysis by local fitting. *Journal of the American Statistical Association*, **83**, 596–610.

Breiman, L. (1991), The pi method for estimating multivariate functions from noisy data. *Technometrics*, **33**, 125–160.

`ozone` *Ozone Concentration in Los Angeles Basin*

Description

Daily measurements of ozone concentration and eight meteorological quantities in the Los Angeles basin for 330 days of 1976.

Usage

```
data(ozone)
```

Format

A data frame containing 330 observations on the following variables.

`upo3`	Upland ozone concentration, in ppm.
`vdht`	Vandenberg 500 millibar height, in meters.
`wdsp`	Wind speed, in miles per hour.
`hmdt`	Humidity.
`sbtp`	Sandburg Air Base temperature, in Celsius.
`ibht`	Inversion base height, in foot.
`dgpg`	Dagget pressure gradient, in mmHg.
`ibtp`	Inversion base temperature, in Fahrenheit.
`vsty`	Visibility, in miles.
`day`	Calendar day, between 1 and 366.

Source

Unknown.

References

Breiman, L. and Friedman, J. H. (1985), Estimating optimal transformations for multiple regression and correlation. *Journal of the American Statistical Association*, **80**, 580–598.

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

`penny` *Thickness of US Lincoln Pennies*

Description

Thickness of US Lincoln pennies minted during years 1945 through 1989.

Usage

```
data(nox)
```

Format

A data frame containing 90 observations on the following variables.

`year`	Year minted.
`mil`	Thickness in mils.

Source

Scott, D. W. (1992), *Multivariate Density Estimation: Theory, Practice and Visualization*. New York: Wiley.

References

Gu, C. (1995), Smoothing spline density estimation: Conditional distribution, *Statistica Sinica*, **5**, 709–726.

Scott, D. W. (1992), *Multivariate Density Estimation: Theory, Practice and Visualization*. New York: Wiley.

`predict.ssanova` *Predicting from Smoothing Spline ANOVA Fits*

Description

Evaluate terms in a smoothing spline ANOVA fit at arbitrary points. Standard errors of the terms can be requested for use in constructing Bayesian confidence intervals.

Usage

```
## S3 method for class 'ssanova'
predict(object, newdata, se.fit=FALSE,
                          include=c(object$terms$labels,object$lab.p), ...)
## S3 method for class 'ssanova0'
predict(object, newdata, se.fit=FALSE,
                           include=c(object$terms$labels,object$lab.p), ...)
## S3 method for class 'ssanova'
predict1(object, contr=c(1,-1), newdata, se.fit=TRUE,
                           include=c(object$terms$labels,object$lab.p), ...)
```

Arguments

`object`	Object of class inheriting from `"ssanova"`.
`newdata`	Data frame or model frame in which to predict.
`se.fit`	Flag indicating if standard errors are required.
`include`	List of model terms to be included in the prediction. The `offset` term, if present, is to be specified by `"offset"`.
`contr`	Contrast coefficients.
`...`	Ignored.

Value

For `se.fit=FALSE`, `predict.ssanova` returns a vector of the evaluated fit.

For `se.fit=TRUE`, `predict.ssanova` returns a list consisting of the following elements.

`fit`	Vector of evaluated fit.
`se.fit`	Vector of standard errors.

Note

For mixed-effect models through `ssanova` or `gssanova`, the Z matrix is set to 0 if not supplied. To supply the Z matrix, add an element `random=I(...)` in `newdata`, where the as-is function `I(...)` preserves the integrity of the Z matrix in data frame.

`predict1.ssanova` takes a list of data frames in `newdata` representing `x1`, `x2`, etc. By default, it calculates f(x1)-f(x2) along with standard errors. While pairwise contrast is the targeted application, all linear combinations can be computed.

For `"gssanova"` objects, the results are on the link scale. See also `predict9.gssanova`.

References

Gu, C. (1992), Penalized likelihood regression: a Bayesian analysis. *Statistica Sinica*, **2**, 255–264.

Gu, C. and Wahba, G. (1993), Smoothing spline ANOVA with component-wise Bayesian "confidence intervals." *Journal of Computational and Graphical Statistics*, **2**, 97–117.

Kim, Y.-J. and Gu, C. (2004), Smoothing spline Gaussian regression: more scalable computation via efficient approximation. *Journal of the Royal Statistical Society, Ser. B*, **66**, 337–356.

See Also

Fitting functions `ssanova`, `ssanova0`, `gssanova`, `gssanova0` and methods `summary.ssanova`, `summary.gssanova`, `summary.gssanova0`, `project.ssanova`, `fitted.ssanova`.

Examples

```
## THE FOLLOWING EXAMPLE IS TIME-CONSUMING
## Not run:
## Fit a model with cubic and thin-plate marginals, where geog is 2-D
data(LakeAcidity)
fit <- ssanova(ph~log(cal)*geog,,LakeAcidity)
## Obtain estimates and standard errors on a grid
new <- data.frame(cal=1,geog=I(matrix(0,1,2)))
new <- model.frame(~log(cal)+geog,new)
predict(fit,new,se=TRUE)
## Evaluate the geog main effect
predict(fit,new,se=TRUE,inc="geog")
## Evaluate the sum of the geog main effect and the interaction
predict(fit,new,se=TRUE,inc=c("geog","log(cal):geog"))
## Evaluate the geog main effect on a grid
grid <- seq(-.04,.04,len=21)
new <- model.frame(~geog,list(geog=cbind(rep(grid,21),rep(grid,rep(21,21)))))
est <- predict(fit,new,se=TRUE,inc="geog")
## Plot the fit and standard error
par(pty="s")
contour(grid,grid,matrix(est$fit,21,21),col=1)
contour(grid,grid,matrix(est$se,21,21),add=TRUE,col=2)
## Clean up
rm(LakeAcidity,fit,new,grid,est)
dev.off()

## End(Not run)
```

`predict.sscox` *Evaluating Smoothing Spline ANOVA Estimate of Relative Risk*

Description

Evaluate terms in a smoothing spline ANOVA estimate of relative risk at arbitrary points. Standard errors of the terms can be requested for use in constructing Bayesian confidence intervals.

Usage

```
## S3 method for class 'sscox'
predict(object, newdata, se.fit=FALSE,
                        include=c(object$terms$labels,object$lab.p), ...)
```

Arguments

`object`	Object of class `"sscox"`.
`newdata`	Data frame or model frame in which to predict.
`se.fit`	Flag indicating if standard errors are required.
`include`	List of model terms to be included in the prediction.
`...`	Ignored.

Value

For `se.fit=FALSE`, `predict.sscox` returns a vector of the evaluated relative risk.

For `se.fit=TRUE`, `predict.sscox` returns a list consisting of the following elements.

`fit`	Vector of evaluated relative risk.
`se.fit`	Vector of standard errors for log relative risk.

Note

For mixed-effect models through `sscox`, the Z matrix is set to 0 if not supplied. To supply the Z matrix, add an element `random=I(...)` in `newdata`, where the as-is function `I(...)` preserves the integrity of the Z matrix in data frame.

See Also

Fitting functions `sscox` and method `project.sscox`.

`predict.ssllrm` *Evaluating Log-Linear Regression Model Fits*

Description

Evaluate conditional density in a log-linear regression model fit at arbitrary x, or contrast of log conditional density possibly with standard errors for constructing Bayesian confidence intervals.

Usage

```
## S3 method for class 'ssllrm'
predict(object, x, y=object$qd.pt, odds=NULL, se.odds=FALSE, ...)
```

Arguments

`object`	Object of class `"ssllrm"`.
`x`	Data frame of x values.
`y`	Data frame of y values; y-variables must be factors.
`odds`	Optional coefficients of contrast.
`se.odds`	Flag indicating if standard errors are required. Ignored when `odds=NULL`.
`...`	Ignored.

Value

For `odds=NULL`, `predict.ssanova` returns a vector/matrix of the estimated `f(y|x)`.

When `odds` is given, it should match `y` in length and the coefficients must add to zero; `predict.ssanova` then returns a vector of estimated "odds ratios" if `se.odds=FALSE` or a list consisting of the following elements if `se.odds=TRUE`.

`fit`	Vector of evaluated fit.
`se.fit`	Vector of standard errors.

See Also

Fitting function `ssllrm`.

`predict9.gssanova` *Predicting from Smoothing Spline ANOVA Fits with Non-Gaussian Responses*

Description

Evaluate smoothing spline ANOVA fits with non-Gaussian responses at arbitrary points, with results on the response scale.

Usage

```
## S3 method for class 'gssanova'
predict9(object, newdata, ci=FALSE, level=.95, nu=NULL, ...)
```

Arguments

`object`	Object of class inheriting from `"gssanova"`.
`newdata`	Data frame or model frame in which to predict.
`ci`	Flag indicating if Bayesian confidence intervals are required. Ignored for `family="polr"`.
`level`	Confidence level. Ignored when `ci=FALSE`.
`nu`	Sizes for `"nbinomial"` fits with known sizes. Ignored otherwise.
`...`	Ignored.

Value

For `ci=FALSE`, `predict9.gssanova` returns a vector of the evaluated fit,

For `ci=TRUE`, `predict9.gssanova` returns a list of three elements.

`fit`	Vector of evaluated fit on response scale.
`lcl`	Vector of lower confidence limit on response scale.
`ucl`	Vector of upper confidence limit on response scale.

For `family="polr"`, `predict9.gssanova` returns a matrix of probabilities with each row adding up to 1.

Note

For mixed-effect models through `gssanova` or `gssanova1`, the Z matrix is set to 0 if not supplied. To supply the Z matrix, add an element `random=I(...)` in `newdata`, where the as-is function `I(...)` preserves the integrity of the Z matrix in data frame.

Unlike on the link scale, partial sums make no sense on the response scale, so all terms are forced in here.

References

Gu, C. (2013), *Smoothing Spline ANOVA Models (2nd Ed).* New York: Springer-Verlag.

See Also

Fitting functions `gssanova`, `gssanova1` and methods `predict.ssanova`, `summary.gssanova`, `project.gssanova`, `fitted.gssanova`.

`print` *Print Functions for Smoothing Spline ANOVA Models*

Description

Print functions for Smoothing Spline ANOVA models.

Usage

```
## S3 method for class 'ssanova'
print(x, ...)
## S3 method for class 'ssanova0'
print(x, ...)
## S3 method for class 'gssanova'
print(x, ...)
## S3 method for class 'ssden'
print(x, ...)
## S3 method for class 'sscden'
print(x, ...)
## S3 method for class 'sshzd'
print(x, ...)
## S3 method for class 'sscox'
print(x, ...)
## S3 method for class 'ssllrm'
print(x, ...)
## S3 method for class 'summary.ssanova'
print(x, digits=6, ...)
## S3 method for class 'summary.gssanova'
print(x, digits=6, ...)
## S3 method for class 'summary.gssanova0'
print(x, digits=6, ...)
```

Arguments

`x`	Object of class `ssanova`, `summary.ssanova`, `summary.gssanova`, or `ssden`.
`digits`	Number of significant digits to be printed in values.
`...`	Ignored.

See Also

`ssanova`, `ssanova0`, `gssanova`, `gssanova0`, `ssden`, `ssllrm`, `sshzd`, `summary.ssanova`, `summary.gssanova`, `summary.gssanova0`.

`project` *Projecting Smoothing Spline ANOVA Fits for Model Diagnostics*

Description

Calculate Kullback-Leibler projection of smoothing spline ANOVA fits for model diagnostics.

Usage

```
project(object, ...)
## S3 method for class 'ssanova'
project(object, include, ...)
## S3 method for class 'ssanova9'
project(object, include, ...)
## S3 method for class 'gssanova'
project(object, include, ...)
## S3 method for class 'ssden'
project(object, include, mesh=FALSE, ...)
## S3 method for class 'ssden1'
project(object, include, drop1=FALSE, ...)
## S3 method for class 'sscden'
project(object, include, ...)
## S3 method for class 'sscden1'
project(object, include, ...)
## S3 method for class 'sshzd'
project(object, include, mesh=FALSE, ...)
## S3 method for class 'sscox'
project(object, include, ...)
## S3 method for class 'sshzd1'
project(object, include, ...)
## S3 method for class 'ssllrm'
project(object, include, ...)
```

Arguments

`object`	Object of class `"ssanova"`, `"gssanova"`, `"ssden"`, `"ssden1"`, `"sscden"`, `"sscden1"`, `"sshzd"`, `"sshzd1"`, or `"ssllrm"`.
`...`	Additional arguments. Ignored in `project.x`.
`include`	List of model terms to be included in the reduced model space. The `partial` and `offset` terms, if present, are to be specified by `"partial"` and `"offset"`, respectively.
`mesh`	Flag indicating whether to return evaluations of the projection.
`drop1`	If TRUE, calculate `p<-length(include)` projections with `include[-i]`, `i=1,...,p`.

Details

The entropy KL(fit0,null) can be decomposed as the sum of KL(fit0,fit1) and KL(fit1,null), where fit0 is the fit to be projected, fit1 is the projection in the reduced model space, and null is the constant fit. The ratio KL(fit0,fit1)/KL(fit0,null) serves as a diagnostic of the feasibility of the reduced model.

For regression fits, smoothness safe-guard is used to prevent interpolation, and KL(fit0,fit1)+KL(fit1,null) may not match KL(fit0,null) perfectly.

For mixed-effect models from `ssanova` and `gssanova`, the estimated random effects are treated as offset.

Value

The functions return a list consisting of the following elements.

`ratio`	KL(fit0,fit1)/KL(fit0,null); the smaller the value, the more feasible the reduced model is.
`kl`	KL(fit0,fit1).

For regression fits, the list also contains the following element.

`check`	KL(fit0,fit1)/KL(fit0,null)+KL(fit1,null)/KL(fit0,null); a value closer to 1 is preferred.

For density and hazard fits, the list may contain the following optional element.

`mesh`	The evaluations of the projection.

Note

`project.ssden1`, `project.sscden1`, and `project.sshzd1` calculates square error projections.

References

Gu, C. (2004), Model diagnostics for smoothing spline ANOVA models. *The Canadian Journal of Statistics*, **32**, 347–358.

See Also

Fitting functions `ssanova`, `gssanova`, `ssden`, `sshzd`, and `sshzd1`.

`Sachs` *Protein Expression in Human Immune System Cells*

Description

Data concerning protein expression levels in human immune system cells under stimulations.

Usage

```
data(Sachs)
```

Format

A data frame containing 7466 cells, with flow cytometry measurements of 11 phosphorylated proteins and phospholipids, on the `log10` scale of the original.

`praf`	Raf phosphorylated at S259.
`pmek`	Mek1/mek2 phosphorylated at S217/S221.
`plcg`	Phosphorylation of phospholipase $C-\gamma$ on Y783.
`pip2`	Phophatidylinositol 4,5-biphosphate.
`pip3`	Phophatidylinositol 3,4,5-triphosphate.
`p44.42`	Erk1/erk2 phosphorylated at T202/Y204.
`pakts473`	AKT phosphorylated at S473.
`pka`	Phosphorylation of of protein kinase A substrates on 3 sites.
`pkc`	Phosphorylation of of protein kinase C substrates on S660.
`p38`	Erk1/erk2 phosphorylated at T180/Y182.
`pjnk`	Erk1/erk2 phosphorylated at T183/Y185.

Source

Sachs, K., Perez, O., Pe’er, D., Lauffenburger, D. A., and Nolan, G. P. (2005), Causal protein-signaling networks derived from multiparameter single-cell data. *Science*, **308 (5732)**, 523–529.

`smolyak` *Generating Smolyak Cubature*

Description

Generate delayed Smolyak cubatures using C routines modified from `smolyak.c` found in Knut Petras’ SMOLPACK.

Usage

```
smolyak.quad(d, k)

smolyak.size(d, k)
```

Arguments

`d`	Dimension of unit cube.
`k`	Depth of algorithm.

Value

`smolyak.quad` returns a list object with the following elements.

`pt`	Quadrature nodes in rows of matrix.
`wt`	Quadrature weights.

`smolyak.size` returns an integer.

`ssanova` *Fitting Smoothing Spline ANOVA Models*

Description

Fit smoothing spline ANOVA models in Gaussian regression. The symbolic model specification via `formula` follows the same rules as in `lm`.

Usage

```
ssanova(formula, type=NULL, data=list(), weights, subset, offset,
        na.action=na.omit, partial=NULL, method="v", alpha=1.4,
        varht=1, id.basis=NULL, nbasis=NULL, seed=NULL, random=NULL,
        skip.iter=FALSE)
```

Arguments

`formula`	Symbolic description of the model to be fit.
`type`	List specifying the type of spline for each variable. See `mkterm` for details.
`data`	Optional data frame containing the variables in the model.
`weights`	Optional vector of weights to be used in the fitting process.
`subset`	Optional vector specifying a subset of observations to be used in the fitting process.
`offset`	Optional offset term with known parameter 1.
`na.action`	Function which indicates what should happen when the data contain NAs.
`partial`	Optional symbolic description of parametric terms in partial spline models.
`method`	Method for smoothing parameter selection. Supported are `method="v"` for GCV, `method="m"` for GML (REML), and `method="u"` for Mallows' CL.
`alpha`	Parameter modifying GCV or Mallows' CL; larger absolute values yield smoother fits; negative value invokes a stable and more accurate GCV/CL evaluation algorithm but may take two to five times as long. Ignored when `method="m"` are specified.

`varht`	External variance estimate needed for `method="u"`. Ignored when `method="v"` or `method="m"` are specified.
`id.basis`	Index designating selected "knots".
`nbasis`	Number of "knots" to be selected. Ignored when `id.basis` is supplied.
`seed`	Seed to be used for the random generation of "knots". Ignored when `id.basis` is supplied.
`random`	Input for parametric random effects in nonparametric mixed-effect models. See `mkran` for details.
`skip.iter`	Flag indicating whether to use initial values of theta and skip theta iteration. See notes on skipping theta iteration.

Details

The model specification via `formula` is intuitive. For example, `y~x1*x2` yields a model of the form

$$y = C + f_1(x1) + f_2(x2) + f_{12}(x1, x2) + e$$

with the terms denoted by `"1"`, `"x1"`, `"x2"`, and `"x1:x2"`.

The model terms are sums of unpenalized and penalized terms. Attached to every penalized term there is a smoothing parameter, and the model complexity is largely determined by the number of smoothing parameters.

A subset of the observations are selected as "knots." Unless specified via `id.basis` or `nbasis`, the number of "knots" q is determined by $max(30, 10n^{2/9})$, which is appropriate for the default cubic splines for numerical vectors.

Using q "knots," `ssanova` calculates an approximate solution to the penalized least squares problem using algorithms of the order $O(nq^2)$, which for $q << n$ scale better than the $O(n^3)$ algorithms of `ssanova0`. For the exact solution, one may set $q = n$ in `ssanova`, but `ssanova0` would be much faster.

Value

`ssanova` returns a list object of class `"ssanova"`.

The method `summary.ssanova` can be used to obtain summaries of the fits. The method `predict.ssanova` can be used to evaluate the fits at arbitrary points along with standard errors. The method `project.ssanova` can be used to calculate the Kullback-Leibler projection for model selection. The methods `residuals.ssanova` and `fitted.ssanova` extract the respective traits from the fits.

Skipping Theta Iteration

For the selection of multiple smoothing parameters, `nlm` is used to minimize the selection criterion such as the GCV score. When the number of smoothing parameters is large, the process can be time-consuming due to the great amount of function evaluations involved.

The starting values for the `nlm` iteration are obtained using Algorith 3.2 in Gu and Wahba (1991). These starting values usually yield good estimates themselves, leaving the subsequent quasi-Newton iteration to pick up the "last 10%" performance with extra effort many times of the initial one. Thus, it is often a good idea to skip the iteration by specifying `skip.iter=TRUE`, especially in high-dimensions and/or with multi-way interactions.

`skip.iter=TRUE` could be made the default in future releases.

Note

To use GCV and Mallows' CL unmodified, set `alpha=1`.

For simpler models and moderate sample sizes, the exact solution of `ssanova0` can be faster.

The results may vary from run to run. For consistency, specify `id.basis` or set `seed`.

In *gss* versions earlier than 1.0, `ssanova` was under the name `ssanova1`.

References

Wahba, G. (1990), *Spline Models for Observational Data*. Philadelphia: SIAM.

Gu, C. and Wahba, G. (1991), Minimizing GCV/GML scores with multiple smoothing parameters via the Newton method. *SIAM Journal on Scientific and Statistical Computing*, **12**, 383–398.

Kim, Y.-J. and Gu, C. (2004), Smoothing spline Gaussian regression: more scalable computation via efficient approximation. *Journal of the Royal Statistical Society, Ser. B*, **66**, 337–356.

Gu, C. (2013), *Smoothing Spline ANOVA Models (2nd Ed)*. New York: Springer-Verlag.

Gu, C. (2014), Smoothing Spline ANOVA Models: R Package gss. *Journal of Statistical Software*, 58(5), 1-25. URL http://www.jstatsoft.org/v58/i05/.

Examples

```
## Fit a cubic spline
x <- runif(100); y <- 5 + 3*sin(2*pi*x) + rnorm(x)
cubic.fit <- ssanova(y~x)
## Obtain estimates and standard errors on a grid
new <- data.frame(x=seq(min(x),max(x),len=50))
est <- predict(cubic.fit,new,se=TRUE)
## Plot the fit and the Bayesian confidence intervals
plot(x,y,col=1); lines(new$x,est$fit,col=2)
lines(new$x,est$fit+1.96*est$se,col=3)
lines(new$x,est$fit-1.96*est$se,col=3)
## Clean up
## Not run: rm(x,y,cubic.fit,new,est)
dev.off()
## End(Not run)

## Fit a tensor product cubic spline
data(nox)
nox.fit <- ssanova(log10(nox)~comp*equi,data=nox)
## Fit a spline with cubic and nominal marginals
nox$comp<-as.factor(nox$comp)
nox.fit.n <- ssanova(log10(nox)~comp*equi,data=nox)
## Fit a spline with cubic and ordinal marginals
nox$comp<-as.ordered(nox$comp)
nox.fit.o <- ssanova(log10(nox)~comp*equi,data=nox)
## Clean up
## Not run: rm(nox,nox.fit,nox.fit.n,nox.fit.o)
```

`ssanova0` *Fitting Smoothing Spline ANOVA Models*

Description

Fit smoothing spline ANOVA models in Gaussian regression. The symbolic model specification via `formula` follows the same rules as in `lm`.

Usage

```
ssanova0(formula, type=NULL, data=list(), weights, subset,
         offset, na.action=na.omit, partial=NULL, method="v",
         varht=1, prec=1e-7, maxiter=30)
```

Arguments

`formula`	Symbolic description of the model to be fit.
`type`	List specifying the type of spline for each variable. See `mkterm` for details.
`data`	Optional data frame containing the variables in the model.
`weights`	Optional vector of weights to be used in the fitting process.
`subset`	Optional vector specifying a subset of observations to be used in the fitting process.
`offset`	Optional offset term with known parameter 1.
`na.action`	Function which indicates what should happen when the data contain NAs.
`partial`	Optional symbolic description of parametric terms in partial spline models.
`method`	Method for smoothing parameter selection. Supported are `method="v"` for GCV, `method="m"` for GML (REML), and `method="u"` for Mallow's CL.
`varht`	External variance estimate needed for `method="u"`. Ignored when `method="v"` or `method="m"` are specified.
`prec`	Precision requirement in the iteration for multiple smoothing parameter selection. Ignored when only one smoothing parameter is involved.
`maxiter`	Maximum number of iterations allowed for multiple smoothing parameter selection. Ignored when only one smoothing parameter is involved.

Details

The model specification via `formula` is intuitive. For example, `y~x1*x2` yields a model of the form

$$y = C + f_1(x1) + f_2(x2) + f_{12}(x1, x2) + e$$

with the terms denoted by `"1"`, `"x1"`, `"x2"`, and `"x1:x2"`.

The model terms are sums of unpenalized and penalized terms. Attached to every penalized term there is a smoothing parameter, and the model complexity is largely determined by the number of smoothing parameters.

`ssanova0` and the affiliated methods provide a front end to RKPACK, a collection of RATFOR routines for nonparametric regression via the penalized least squares. The algorithms implemented in RKPACK are of the order $O(n^3)$.

Value

`ssanova0` returns a list object of class `c("ssanova0","ssanova")`.

The method `summary.ssanova0` can be used to obtain summaries of the fits. The method `predict.ssanova0` can be used to evaluate the fits at arbitrary points along with standard errors. The methods `residuals.ssanova` and `fitted.ssanova` extract the respective traits from the fits.

Note

For complex models and large sample sizes, the approximate solution of `ssanova` can be faster.

The method `project` is not implemented for `ssanova0`, nor is the mixed-effect model support through `mkran`.

In *gss* versions earlier than 1.0, `ssanova0` was under the name `ssanova`.

References

Wahba, G. (1990), *Spline Models for Observational Data*. Philadelphia: SIAM.

Gu, C. (2013), *Smoothing Spline ANOVA Models (2nd Ed)*. New York: Springer-Verlag.

Gu, C. (2014), Smoothing Spline ANOVA Models: R Package gss. *Journal of Statistical Software*, 58(5), 1-25. URL http://www.jstatsoft.org/v58/i05/.

Examples

```
## Fit a cubic spline
x <- runif(100); y <- 5 + 3*sin(2*pi*x) + rnorm(x)
cubic.fit <- ssanova0(y~x,method="m")
## Obtain estimates and standard errors on a grid
new <- data.frame(x=seq(min(x),max(x),len=50))
est <- predict(cubic.fit,new,se=TRUE)
## Plot the fit and the Bayesian confidence intervals
plot(x,y,col=1); lines(new$x,est$fit,col=2)
lines(new$x,est$fit+1.96*est$se,col=3)
lines(new$x,est$fit-1.96*est$se,col=3)
## Clean up
## Not run: rm(x,y,cubic.fit,new,est)
dev.off()
## End(Not run)

## Fit a tensor product cubic spline
data(nox)
nox.fit <- ssanova0(log10(nox)~comp*equi,data=nox)
## Fit a spline with cubic and nominal marginals
nox$comp<-as.factor(nox$comp)
nox.fit.n <- ssanova0(log10(nox)~comp*equi,data=nox)
## Fit a spline with cubic and ordinal marginals
nox$comp<-as.ordered(nox$comp)
nox.fit.o <- ssanova0(log10(nox)~comp*equi,data=nox)
## Clean up
## Not run: rm(nox,nox.fit,nox.fit.n,nox.fit.o)
```

`ssanova9` *Fitting Smoothing Spline ANOVA Models with Correlated Data*

Description

Fit smoothing spline ANOVA models with correlated Gaussian data. The symbolic model specification via `formula` follows the same rules as in `lm`.

Usage

```
ssanova9(formula, type=NULL, data=list(), subset, offset,
         na.action=na.omit, partial=NULL, method="v", alpha=1.4,
         varht=1, id.basis=NULL, nbasis=NULL, seed=NULL, cov,
         skip.iter=FALSE)

para.arma(fit)
```

Arguments

`formula`	Symbolic description of the model to be fit.
`type`	List specifying the type of spline for each variable. See `mkterm` for details.
`data`	Optional data frame containing the variables in the model.
`subset`	Optional vector specifying a subset of observations to be used in the fitting process.
`offset`	Optional offset term with known parameter 1.
`na.action`	Function which indicates what should happen when the data contain NAs.
`partial`	Optional symbolic description of parametric terms in partial spline models.
`method`	Method for smoothing parameter selection. Supported are `method="v"` for V, `method="m"` for M, and `method="u"` for U; see the reference for definitions of U, V, and M.
`alpha`	Parameter modifying V or U; larger absolute values yield smoother fits. Ignored when `method="m"` are specified.
`varht`	External variance estimate needed for `method="u"`. Ignored when `method="v"` or `method="m"` are specified.
`id.basis`	Index designating selected "knots".
`nbasis`	Number of "knots" to be selected. Ignored when `id.basis` is supplied.
`seed`	Seed to be used for the random generation of "knots". Ignored when `id.basis` is supplied.
`cov`	Input for covariance functions. See `mkcov` for details.
`skip.iter`	Flag indicating whether to use initial values of theta and skip theta iteration. See notes on skipping theta iteration.
`fit`	`ssanova9` fit with ARMA error.

Details

The model specification via `formula` is intuitive. For example, `y~x1*x2` yields a model of the form

$$y = C + f_1(x1) + f_2(x2) + f_{12}(x1, x2) + e$$

with the terms denoted by `"1"`, `"x1"`, `"x2"`, and `"x1:x2"`.

The model terms are sums of unpenalized and penalized terms. Attached to every penalized term there is a smoothing parameter, and the model complexity is largely determined by the number of smoothing parameters.

A subset of the observations are selected as "knots." Unless specified via `id.basis` or `nbasis`, the number of "knots" q is determined by $max(30, 10n^{2/9})$, which is appropriate for the default cubic splines for numerical vectors.

Using q "knots," `ssanova` calculates an approximate solution to the penalized least squares problem using algorithms of the order $O(nq^2)$, which for $q << n$ scale better than the $O(n^3)$ algorithms of `ssanova0`. For the exact solution, one may set $q = n$ in `ssanova`, but `ssanova0` would be much faster.

Value

`ssanova9` returns a list object of class `c("ssanova9","ssanova")`.

The method `summary.ssanova9` can be used to obtain summaries of the fits. The method `predict.ssanova` can be used to evaluate the fits at arbitrary points along with standard errors. The method `project.ssanova9` can be used to calculate the Kullback-Leibler projection for model selection. The methods `residuals.ssanova` and `fitted.ssanova` extract the respective traits from the fits.

`para.arma` returns the fitted ARMA coefficients for `cov=list("arma",c(p,q))` in the call to `ssanova9`.

Skipping Theta Iteration

For the selection of multiple smoothing parameters, `nlm` is used to minimize the selection criterion such as the GCV score. When the number of smoothing parameters is large, the process can be time-consuming due to the great amount of function evaluations involved.

The starting values for the `nlm` iteration are obtained using Algorith 3.2 in Gu and Wahba (1991). These starting values usually yield good estimates themselves, leaving the subsequent quasi-Newton iteration to pick up the "last 10%" performance with extra effort many times of the initial one. Thus, it is often a good idea to skip the iteration by specifying `skip.iter=TRUE`, especially in high-dimensions and/or with multi-way interactions.

`skip.iter=TRUE` could be made the default in future releases.

Note

The results may vary from run to run. For consistency, specify `id.basis` or set `seed`.

References

Han, C. and Gu, C. (2008), Optimal smoothing with correlated data, *Sankhya*, **70-A**, 38–72.

Gu, C. (2013), *Smoothing Spline ANOVA Models (2nd Ed)*. New York: Springer-Verlag.

Gu, C. (2014), Smoothing Spline ANOVA Models: R Package gss. *Journal of Statistical Software*, 58(5), 1-25. URL http://www.jstatsoft.org/v58/i05/.

Examples

```
x <- runif(100); y <- 5 + 3*sin(2*pi*x) + rnorm(x)
## independent fit
fit <- ssanova9(y~x,cov=list("known",diag(1,100)))
## AR(1) fit
fit <- ssanova9(y~x,cov=list("arma",c(1,0)))
para.arma(fit)
## MA(1) fit
e <- rnorm(101); e <- e[-1]-.5*e[-101]
x <- runif(100); y <- 5 + 3*sin(2*pi*x) + e
fit <- ssanova9(y~x,cov=list("arma",c(0,1)))
para.arma(fit)
## Clean up
## Not run: rm(x,y,e,fit)
```

`sscden` *Estimating Conditional Probability Density Using Smoothing Splines*

Description

Estimate conditional probability densities using smoothing spline ANOVA models. The symbolic model specification via `formula` follows the same rules as in `lm`.

Usage

```
sscden(formula, response, type=NULL, data=list(), weights, subset,
       na.action=na.omit, alpha=1.4, id.basis=NULL, nbasis=NULL,
       seed=NULL, ydomain=as.list(NULL), yquad=NULL, prec=1e-7,
       maxiter=30, skip.iter=FALSE)

sscden1(formula, response, type=NULL, data=list(), weights, subset,
        na.action=na.omit, alpha=1.4, id.basis=NULL, nbasis=NULL,
        seed=NULL, rho=list("xy"), ydomain=as.list(NULL), yquad=NULL,
        prec=1e-7, maxiter=30, skip.iter=FALSE)
```

Arguments

`formula`	Symbolic description of the model to be fit.
`response`	Formula listing response variables.
`type`	List specifying the type of spline for each variable. See `mkterm` for details.

`data`	Optional data frame containing the variables in the model.
`weights`	Optional vector of counts for duplicated data.
`subset`	Optional vector specifying a subset of observations to be used in the fitting process.
`na.action`	Function which indicates what should happen when the data contain NAs.
`alpha`	Parameter defining cross-validation scores for smoothing parameter selection.
`id.basis`	Index of observations to be used as "knots."
`nbasis`	Number of "knots" to be used. Ignored when `id.basis` is specified.
`seed`	Seed to be used for the random generation of "knots." Ignored when `id.basis` is specified.
`ydomain`	Data frame specifying marginal support of conditional density.
`yquad`	Quadrature for calculating integral on Y domain. Mandatory if response variables other than factors or numerical vectors are involved.
`prec`	Precision requirement for internal iterations.
`maxiter`	Maximum number of iterations allowed for internal iterations.
`skip.iter`	Flag indicating whether to use initial values of theta and skip theta iteration. See `ssanova` for notes on skipping theta iteration.
`rho`	rho function needed for sscden1.

Details

The model is specified via `formula` and `response`, where `response` lists the response variables. For example, `sscden(~y*x,~y)` prescribe a model of the form

$$logf(y|x) = g_y(y) + g_{xy}(x,y) + C(x)$$

with the terms denoted by `"y"`, `"y:x"`; the term(s) not involving response(s) are removed and the constant `C(x)` is determined by the fact that a conditional density integrates to one on the `y` axis. `sscden1` does keep terms not involving response(s) during estimation, although those terms cancel out when one evaluates the estimated conditional density.

The model terms are sums of unpenalized and penalized terms. Attached to every penalized term there is a smoothing parameter, and the model complexity is largely determined by the number of smoothing parameters.

A subset of the observations are selected as "knots." Unless specified via `id.basis` or `nbasis`, the number of "knots" q is determined by $max(30, 10n^{2/9})$, which is appropriate for the default cubic splines for numerical vectors.

Value

`sscden` returns a list object of class `"sscden"`. `sscden1` returns a list object of class `c("sscden1","sscden")`.

`dsscden` and `cdsscden` can be used to evaluate the estimated conditional density $f(y|x)$ and $f(y1|x,y2)$; `psscden`, `qsscden`, `cpsscden`, and `cqsscden` can be used to evaluate conditional cdf and quantiles. The methods `project.sscden` or `project.sscden1` can be used to calculate the Kullback-Leibler or square-error projections for model selection.

Note

Default quadrature on the Y domain will be constructed for numerical vectors on a hyper cube, then outer product with factor levels will be taken if factors are involved. The sides of the hyper cube are specified by `ydomain`; for `ydomain$y` missing, the default is `c(min(y),max(y))+c(-1,1)*(max(y)-mimn(y))*.05`.

On a 1-D interval, the quadrature is the 200-point Gauss-Legendre formula returned from `gauss.quad`. For multiple numerical vectors, delayed Smolyak cubatures from `smolyak.quad` are used on cubes with the marginals properly transformed; see Gu and Wang (2003) for the marginal transformations.

The results may vary from run to run. For consistency, specify `id.basis` or set `seed`.

For reasonable execution time in high dimensions, set `skip.iter=TRUE`.

References

Gu, C. (1995), Smoothing spline density estimation: Conditional distribution. *Statistica Sinica*, **5**, 709–726. Springer-Verlag.

Gu, C. (2014), Smoothing Spline ANOVA Models: R Package gss. *Journal of Statistical Software*, 58(5), 1-25. URL http://www.jstatsoft.org/v58/i05/.

Examples

```
data(penny); set.seed(5732)
fit <- sscden1(~year*mil,~mil,data=penny,
               ydomain=data.frame(mil=c(49,61)))
yy <- 1944+(0:92)/2
quan <- qsscden(fit,c(.05,.25,.5,.75,.95),
                data.frame(year=yy))
plot(penny$year+.1*rnorm(90),penny$mil,ylim=c(49,61))
for (i in 1:5) lines(yy,quan[i,])
## Clean up
## Not run: rm(penny,yy,quan)
```

`sscomp` *Composition Estimation*

Description

Estimate composition using multinomial counts.

Usage

```
sscomp(x,wt=rep(1,length(x)),alpha=1.4)

sscomp2(x,alpha=1.4)
```

Arguments

`x`	Numerical vector or matrix of multinomial counts.
`wt`	Numerical vector of integration weights.
`alpha`	Parameter defining cross-validation score for smoothing parameter selection.

Details

`sscomp` takes a vector `x` to estimate composition using density estimation on a nominal discrete domain; zero counts must be included in `x` to specify the domain. `wt` mimicking the shape of the unknown density could improve performance.

`sscomp2` takes a matrix `x`, collapses columns to estimate a density using `sscomp`, then using that as `wt` in further `sscomp` calls to estimate composition for each column.

Value

`sscomp` returns a column of estimated probabilities.

`sscomp2` returns a matrix of estimated probabilities, matching the input `x` in dimensions.

References

Gu, C. (2020), Composition estimation via shrinkage. *manuscript*.

`sscopu` *Estimating Copula Density Using Smoothing Splines*

Description

Estimate copula densities using tensor-product cubic splines.

Usage

```
sscopu(x, symmetry=FALSE, alpha=1.4, order=NULL, exclude=NULL,
       weights=NULL, id.basis=NULL, nbasis=NULL, seed=NULL,
       qdsz.depth=NULL, prec=1e-7, maxiter=30, skip.iter=dim(x)[2]!=2)

sscopu2(x, censoring=NULL, truncation=NULL, symmetry=FALSE, alpha=1.4,
        weights=NULL, id.basis=NULL, nbasis=NULL, seed=NULL, prec=1e-7,
        maxiter=30)
```

Arguments

`x`	Matrix of observations on unit cubes.
`symmetry`	Flag indicating whether to enforce symmetry, or invariance under coordinate permutation.
`order`	Highest order of interaction terms in log density. When `NULL`, it is set to `dim(x)[2]` internally.
`exclude`	Pair(s) of marginals whose interactions to be excluded in log density.
`alpha`	Parameter defining cross-validation score for smoothing parameter selection.
`weights`	Optional vector of bin-counts for histogram data.
`id.basis`	Index of observations to be used as "knots."

`nbasis`	Number of "knots" to be used. Ignored when `id.basis` is specified.
`seed`	Seed to be used for the random generation of "knots." Ignored when `id.basis` is specified.
`qdsz.depth`	Depth to be used in `smolyak.quad` for the generation of quadrature.
`prec`	Precision requirement for internal iterations.
`maxiter`	Maximum number of iterations allowed for internal iterations.
`skip.iter`	Flag indicating whether to use initial values of theta and skip theta iteration. See `ssanova` for notes on skipping theta iteration.
`censoring`	Optional censoring indicator.
`truncation`	Optional truncation points.

Details

`sscopu` is essentially `ssden` applied to observations on unit cubes. Instead of variables in data frames, the data are entered as a numerical matrix, and model complexity is globally controlled by the highest order of interactions allowed in log density.

`sscopu2` further restricts the domain to the unit square, but allows for possible censoring and truncation. With `censoring==0,1,2,3`, a data point $(x1, x2)$ represents exact observation, $[0, x1]xx2$, $x1x[0, x2]$, or $[0, x1]x[0, x2]$. With `truncation` point $(t1, t2)$, the sample is taken from $[0, t1]x[0, t2]$ instead of the unit square.

With `symmetriy=TRUE`, one may enforce the interchangeability of coordinates so that $f(x1, x2) = f(x2, x1)$, say.

When `(1,2)` is a row in `exclude`, interaction terms involving coordinates `1` and `2` are excluded.

Value

`sscopu` and `sscopu2` return a list object of class `"sscopu"`. `dsscopu` can be used to evaluate the estimated copula density. A "copularization" process is applied to the estimated density by default so the resulting marginal densities are guaranteed to be uniform.

`cdsscopu`, `cpsscopu`, and `cqsscopu` can be used to evaluate 1-D conditional pdf, cdf, and quantiles.

Note

For reasonable execution time in higher dimensions, set `skip.iter=TRUE` in calls to `sscopu`.

When `"Newton iteration diverges"` in `sscopu`, try to use a larger `qdsz.depth`; the default values for dimensions 2, 3, 4, 5, 6+ are 24, 14, 12, 11, 10. To be sure a larger `qdsz.depth` indeed makes difference, verify the cubature size using `smolyak.size`.

The results may vary from run to run. For consistency, specify `id.basis` or set `seed`.

Author(s)

Chong Gu, `<chong@stat.purdue.edu>`

References

Gu, C. (2013), *Smoothing Spline ANOVA Models (2nd Ed)*. New York: Springer-Verlag.

Gu, C. (2015), Hazard estimation with bivariate survival data and copula density estimation. *Journal of Computational and Graphical Statistics*, **24**, 1053-1073.

Examples

```
## simulate 2-D data
x <- matrix(runif(200),100,2)
## fit copula density
fit <- sscopu(x)
## "same fit"
fit2 <- sscopu2(x,id=fit$id)
## symmetric fit
fit.s <- sscopu(x,sym=TRUE,id=fit$id)
## Not run:
## Kendall's tau and Spearman's rho
summary(fit); summary(fit2); summary(fit.s)
## clean up
rm(x,fit,fit2,fit.s)

## End(Not run)
```

`sscox` *Estimating Relative Risk Using Smoothing Splines*

Description

Estimate relative risk using smoothing spline ANOVA models. The symbolic model specification via `formula` follows the same rules as in `lm`, but with the response of a special form.

Usage

```
sscox(formula, type=NULL, data=list(), weights=NULL, subset,
      na.action=na.omit, partial=NULL, alpha=1.4, id.basis=NULL,
      nbasis=NULL, seed=NULL, random=NULL, prec=1e-7, maxiter=30,
      skip.iter=FALSE)
```

Arguments

`formula`	Symbolic description of the model to be fit, where the response is of the form `Surv(futime,status,start=0)`.
`type`	List specifying the type of spline for each variable. See `mkterm` for details.
`data`	Optional data frame containing the variables in the model.
`weights`	Optional vector of counts for duplicated data.
`subset`	Optional vector specifying a subset of observations to be used in the fitting process.

`na.action`	Function which indicates what should happen when the data contain NAs.
`partial`	Optional symbolic description of parametric terms in partial spline models.
`alpha`	Parameter defining cross-validation score for smoothing parameter selection.
`id.basis`	Index of observations to be used as "knots."
`nbasis`	Number of "knots" to be used. Ignored when `id.basis` is specified.
`seed`	Seed to be used for the random generation of "knots." Ignored when `id.basis` is specified.
`random`	Input for parametric random effects (frailty) in nonparametric mixed-effect models. See `mkran` for details.
`prec`	Precision requirement for internal iterations.
`maxiter`	Maximum number of iterations allowed for internal iterations.
`skip.iter`	Flag indicating whether to use initial values of theta and skip theta iteration. See `ssanova` for notes on skipping theta iteration.

Details

A proportional hazard model is assumed, and the relative risk is estimated via penalized partial likelihood. The model specification via `formula` is for the log relative risk. For example, `Suve(t,d)~u*v` prescribes a model of the form

$$log f(u,v) = g_u(u) + g_v(v) + g_{u,v}(u,v)$$

with the terms denoted by `"u"`, `"v"`, and `"u:v"`; relative risk is defined only up to a multiplicative constant, so the constant term is not included in the model.

`sscox` takes standard right-censored lifetime data, with possible left-truncation and covariates; in `Surv(futime,status,start=0)~...`, `futime` is the follow-up time, `status` is the censoring indicator, and `start` is the optional left-truncation time.

Parallel to those in a `ssanova` object, the model terms are sums of unpenalized and penalized terms. Attached to every penalized term there is a smoothing parameter, and the model complexity is largely determined by the number of smoothing parameters.

The selection of smoothing parameters is through a cross-validation mechanism designed for density estimation under biased sampling, with a fudge factor `alpha`; `alpha=1` is "unbiased" for the minimization of Kullback-Leibler loss but may yield severe undersmoothing, whereas larger `alpha` yields smoother estimates.

A subset of the observations are selected as "knots." Unless specified via `id.basis` or `nbasis`, the number of "knots" q is determined by $max(30, 10n^{2/9})$, which is appropriate for the default cubic splines for numerical vectors.

Value

`sscox` returns a list object of class `"sscox"`.

The method `predict.sscox` can be used to evaluate the fits at arbitrary points along with standard errors. The method `project.sscox` can be used to calculate the Kullback-Leibler projection for model selection.

Note

The function `Surv(futime,status,start=0)` is defined and parsed inside `sscox`, not quite the same as the one in the `survival` package. The estimation is invariant of monotone transformations of time.

The results may vary from run to run. For consistency, specify `id.basis` or set `seed`.

References

Gu, C. (2013), *Smoothing Spline ANOVA Models (2nd Ed)*. New York: Springer-Verlag.

Gu, C. (2014), Smoothing Spline ANOVA Models: R Package gss. *Journal of Statistical Software*, 58(5), 1-25. URL http://www.jstatsoft.org/v58/i05/.

Examples

```
## Relative Risk
data(stan)
fit.rr <- sscox(Surv(futime,status)~age,data=stan)
est.rr <- predict(fit.rr,data.frame(age=c(35,40)),se=TRUE)
## Base Hazard
risk <- predict(fit.rr,stan)
fit.bh <- sshzd(Surv(futime,status)~futime,data=stan,offset=log(risk))
tt <- seq(0,max(stan$futime),length=51)
est.bh <- hzdcurve.sshzd(fit.bh,tt,se=TRUE)
## Clean up
## Not run: rm(stan,fit.rr,est.rr,risk,fit.bh,tt,est.bh)
```

`ssden` *Estimating Probability Density Using Smoothing Splines*

Description

Estimate probability densities using smoothing spline ANOVA models. The symbolic model specification via `formula` follows the same rules as in `lm`, but with the response missing.

Usage

```
ssden(formula, type=NULL, data=list(), alpha=1.4, weights=NULL,
      subset, na.action=na.omit, id.basis=NULL, nbasis=NULL, seed=NULL,
      domain=as.list(NULL), quad=NULL, qdsz.depth=NULL, bias=NULL,
      prec=1e-7, maxiter=30, skip.iter=FALSE)

ssden1(formula, type=NULL, data=list(), alpha=1.4, weights=NULL,
       subset, na.action=na.omit, id.basis=NULL, nbasis=NULL, seed=NULL,
       domain=as.list(NULL), quad=NULL, prec=1e-7, maxiter=30)
```

Arguments

`formula`	Symbolic description of the model to be fit.
`type`	List specifying the type of spline for each variable. See `mkterm` for details.
`data`	Optional data frame containing the variables in the model.
`alpha`	Parameter defining cross-validation score for smoothing parameter selection.
`weights`	Optional vector of bin-counts for histogram data.
`subset`	Optional vector specifying a subset of observations to be used in the fitting process.
`na.action`	Function which indicates what should happen when the data contain NAs.
`id.basis`	Index of observations to be used as "knots."
`nbasis`	Number of "knots" to be used. Ignored when `id.basis` is specified.
`seed`	Seed to be used for the random generation of "knots." Ignored when `id.basis` is specified.
`domain`	Data frame specifying marginal support of density.
`quad`	Quadrature for calculating integral. Mandatory if variables other than factors or numerical vectors are involved.
`qdsz.depth`	Depth to be used in `smolyak.quad` for the generation of quadrature.
`bias`	Input for sampling bias.
`prec`	Precision requirement for internal iterations.
`maxiter`	Maximum number of iterations allowed for internal iterations.
`skip.iter`	Flag indicating whether to use initial values of theta and skip theta iteration. See `ssanova` for notes on skipping theta iteration.

Details

The model specification via `formula` is for the log density. For example, `~x1*x2` prescribes a model of the form

$$logf(x1, x2) = g_1(x1) + g_2(x2) + g_{12}(x1, x2) + C$$

with the terms denoted by `"x1"`, `"x2"`, and `"x1:x2"`; the constant is determined by the fact that a density integrates to one.

The selective term elimination may characterize (conditional) independence structures between variables. For example, `~x1*x2+x1*x3` yields the conditional independence of x2 and x3 given x1.

Parallel to those in a `ssanova` object, the model terms are sums of unpenalized and penalized terms. Attached to every penalized term there is a smoothing parameter, and the model complexity is largely determined by the number of smoothing parameters.

The selection of smoothing parameters is through a cross-validation mechanism described in the references, with a parameter `alpha`; `alpha=1` is "unbiased" for the minimization of Kullback-Leibler loss but may yield severe undersmoothing, whereas larger `alpha` yields smoother estimates.

A subset of the observations are selected as "knots." Unless specified via `id.basis` or `nbasis`, the number of "knots" q is determined by $max(30, 10n^{2/9})$, which is appropriate for the default cubic splines for numerical vectors.

Value

`ssden` returns a list object of class `"ssden"`. `ssden1` returns a list object of class `c("ssden1","ssden")`.

`dssden` and `cdssden` can be used to evaluate the estimated joint density and conditional density; `pssden`, `qssden`, `cpssden`, and `cqssden` can be used to evaluate (conditional) cdf and quantiles.

The method `project.ssden` can be used to calculate the Kullback-Leibler projection of `"ssden"` objects for model selection; `project.ssden1` can be used to calculate the square error projection of `"ssden1"` objects.

Note

In `ssden`, default quadrature will be constructed for numerical vectors on a hyper cube, then outer product with factor levels will be taken if factors are involved. The sides of the hyper cube are specified by `domain`; for `domain$x` missing, the default is `c(min(x),max(x))+c(-1,1)*(max(x)-mimn(x))*.05`. In 1-D, the quadrature is the 200-point Gauss-Legendre formula returned from `gauss.quad`. In multi-D, delayed Smolyak cubatures from `smolyak.quad` are used on cubes with the marginals properly transformed; see Gu and Wang (2003) for the marginal transformations.

For reasonable execution time in higher dimensions, set `skip.iter=TRUE` in call to `ssden`.

If you get an error message from `ssden` stating `"Newton iteration diverges"`, try to use a larger `qdsz.depth` which will execute slower, or switch to `ssden1`. The default values of `qdsz.depth` for dimensions 4, 5, 6+ are 12, 11, 10.

`ssden1` does not involve multi-D quadrature but does not perform as well as `ssden`. It can be used in very high dimensions where `ssden` is infeasible.

The results may vary from run to run. For consistency, specify `id.basis` or set `seed`.

Author(s)

Chong Gu, `<chong@stat.purdue.edu>`

References

Gu, C. and Wang, J. (2003), Penalized likelihood density estimation: Direct cross-validation and scalable approximation. *Statistica Sinica*, **13**, 811–826.

Gu, C., Jeon, Y., and Lin, Y. (2013), Nonparametric density estimation in high dimensions. *Statistica Sinica*, **23**, 1131–1153.

Gu, C. (2013), *Smoothing Spline ANOVA Models (2nd Ed)*. New York: Springer-Verlag.

Gu, C. (2014), Smoothing Spline ANOVA Models: R Package gss. *Journal of Statistical Software*, 58(5), 1-25. URL http://www.jstatsoft.org/v58/i05/.

Examples

```
## 1-D estimate: Buffalo snowfall
data(buffalo)
buff.fit <- ssden(~buffalo,domain=data.frame(buffalo=c(0,150)))
plot(xx<-seq(0,150,len=101),dssden(buff.fit,xx),type="l")
plot(xx,pssden(buff.fit,xx),type="l")
plot(qq<-seq(0,1,len=51),qssden(buff.fit,qq),type="l")
```

```
## Clean up
## Not run: rm(buffalo,buff.fit,xx,qq)
dev.off()
## End(Not run)

## 2-D with triangular domain: AIDS incubation
data(aids)
## rectangular quadrature
quad.pt <- expand.grid(incu=((1:40)-.5)/40*100,infe=((1:40)-.5)/40*100)
quad.pt <- quad.pt[quad.pt$incu<=quad.pt$infe,]
quad.wt <- rep(1,nrow(quad.pt))
quad.wt[quad.pt$incu==quad.pt$infe] <- .5
quad.wt <- quad.wt/sum(quad.wt)*5e3
## additive model (pre-truncation independence)
aids.fit <- ssden(~incu+infe,data=aids,subset=age>=60,
                  domain=data.frame(incu=c(0,100),infe=c(0,100)),
                  quad=list(pt=quad.pt,wt=quad.wt))
## conditional (marginal) density of infe
jk <- cdssden(aids.fit,xx<-seq(0,100,len=51),data.frame(incu=50))
plot(xx,jk$pdf,type="l")
## conditional (marginal) quantiles of infe (TIME-CONSUMING)
## Not run:
cqssden(aids.fit,c(.05,.25,.5,.75,.95),data.frame(incu=50))

## End(Not run)
## Clean up
## Not run: rm(aids,quad.pt,quad.wt,aids.fit,jk,xx)
dev.off()
## End(Not run)

## One factor plus one vector
data(gastric)
gastric$trt
fit <- ssden(~futime*trt,data=gastric)
## conditional density
cdssden(fit,c("1","2"),cond=data.frame(futime=150))
## conditional quantiles
cqssden(fit,c(.05,.25,.5,.75,.95),data.frame(trt=as.factor("1")))
## Clean up
## Not run: rm(gastric,fit)

## Sampling bias
## (X,T) is truncated to T<X<1 for T~U(0,1), so X is length-biased
rbias <- function(n) {
  t <- runif(n)
  x <- rnorm(n,.5,.15)
  ok <- (x>t)&(x<1)
  while(m<-sum(!ok)) {
    t[!ok] <- runif(m)
    x[!ok] <- rnorm(m,.5,.15)
    ok <- (x>t)&(x<1)
  }
  cbind(x,t)
```

```
}
xt <- rbias(100)
x <- xt[,1]; t <- xt[,2]
## length-biased
bias1 <- list(t=1,wt=1,fun=function(t,x){x[,]})
fit1 <- ssden(~x,domain=list(x=c(0,1)),bias=bias1)
plot(xx<-seq(0,1,len=101),dssden(fit1,xx),type="l")
## truncated
bias2 <- list(t=t,wt=rep(1/100,100),fun=function(t,x){x[,]>t})
fit2 <- ssden(~x,domain=list(x=c(0,1)),bias=bias2)
plot(xx,dssden(fit2,xx),type="l")
## Clean up
## Not run: rm(rbias,xt,x,t,bias1,fit1,bias2,fit2)
```

`sshzd` *Estimating Hazard Function Using Smoothing Splines*

Description

Estimate hazard function using smoothing spline ANOVA models. The symbolic model specification via `formula` follows the same rules as in `lm`, but with the response of a special form.

Usage

```
sshzd(formula, type=NULL, data=list(), alpha=1.4, weights=NULL,
      subset, offset, na.action=na.omit, partial=NULL, id.basis=NULL,
      nbasis=NULL, seed=NULL, random=NULL, prec=1e-7, maxiter=30,
      skip.iter=FALSE)

sshzd1(formula, type=NULL, data=list(), alpha=1.4, weights=NULL,
       subset, na.action=na.omit, rho="marginal", partial=NULL,
       id.basis=NULL, nbasis=NULL, seed=NULL, random=NULL, prec=1e-7,
       maxiter=30, skip.iter=FALSE)
```

Arguments

`formula`	Symbolic description of the model to be fit, where the response is of the form `Surv(futime,status,start=0)`.
`type`	List specifying the type of spline for each variable. See `mkterm` for details.
`data`	Optional data frame containing the variables in the model.
`alpha`	Parameter defining cross-validation score for smoothing parameter selection.
`weights`	Optional vector of counts for duplicated data.
`subset`	Optional vector specifying a subset of observations to be used in the fitting process.
`offset`	Optional offset term with known parameter 1.
`na.action`	Function which indicates what should happen when the data contain NAs.

`partial`	Optional symbolic description of parametric terms in partial spline models.
`id.basis`	Index of observations to be used as "knots."
`nbasis`	Number of "knots" to be used. Ignored when `id.basis` is specified.
`seed`	Seed to be used for the random generation of "knots." Ignored when `id.basis` is specified.
`random`	Input for parametric random effects (frailty) in nonparametric mixed-effect models. See `mkran` for details.
`prec`	Precision requirement for internal iterations.
`maxiter`	Maximum number of iterations allowed for internal iterations.
`skip.iter`	Flag indicating whether to use initial values of theta and skip theta iteration. See `ssanova` for notes on skipping theta iteration.
`rho`	Choice of rho function for sshzd1: `"marginal"` or `"weibull"`.

Details

The model specification via `formula` is for the log hazard. For example, `Suve(t,d)~t*u` prescribes a model of the form

$$log f(t,u) = C + g_t(t) + g_u(u) + g_{t,u}(t,u)$$

with the terms denoted by `"1"`, `"t"`, `"u"`, and `"t:u"`. Replacing `t*u` by `t+u` in the `formula`, one gets a proportional hazard model with $g_{t,u} = 0$.

`sshzd` takes standard right-censored lifetime data, with possible left-truncation and covariates; in `Surv(futime,status,start=0)~...`, `futime` is the follow-up time, `status` is the censoring indicator, and `start` is the optional left-truncation time. The main effect of `futime` must appear in the model terms specified via `...`.

Parallel to those in a `ssanova` object, the model terms are sums of unpenalized and penalized terms. Attached to every penalized term there is a smoothing parameter, and the model complexity is largely determined by the number of smoothing parameters.

The selection of smoothing parameters is through a cross-validation mechanism described in Gu (2002, Sec. 7.2), with a parameter `alpha`; `alpha=1` is "unbiased" for the minimization of Kullback-Leibler loss but may yield severe undersmoothing, whereas larger `alpha` yields smoother estimates.

A subset of the observations are selected as "knots." Unless specified via `id.basis` or `nbasis`, the number of "knots" q is determined by $max(30, 10n^{2/9})$, which is appropriate for the default cubic splines for numerical vectors.

Value

`sshzd` returns a list object of class `"sshzd"`. `sshzd1` returns a list object of class `c("sshzd1","sshzd")`.

`hzdrate.sshzd` can be used to evaluate the estimated hazard function. `hzdcurve.sshzd` can be used to evaluate hazard curves with fixed covariates. `survexp.sshzd` can be used to calculated estimated expected survival.

The method `project.sshzd` can be used to calculate the Kullback-Leibler projection of `"sshzd"` objects for model selection; `project.sshzd1` can be used to calculate the square error projection of `"sshzd1"` objects.

Note

The function `Surv(futime,status,start=0)` is defined and parsed inside `sshzd`, not quite the same as the one in the `survival` package.

Integration on the time axis is done by the 200-point Gauss-Legendre formula on `c(min(start),max(futime))`, returned from `gauss.quad`.

`sshzd1` can be up to 50 times faster than `sshzd`, at the cost of performance degradation.

The results may vary from run to run. For consistency, specify `id.basis` or set `seed`.

References

Du, P. and Gu, C. (2006), Penalized likelihood hazard estimation: efficient approximation and Bayesian confidence intervals. *Statistics and Probability Letters*, **76**, 244–254.

Du, P. and Gu, C. (2009), Penalized Pseudo-Likelihood Hazard Estimation: A Fast Alternative to Penalized Likelihood. *Journal of Statistical Planning and Inference*, **139**, 891–899.

Du, P. and Ma, S. (2010), Frailty Model with Spline Estimated Nonparametric Hazard Function, *Statistica Sinica*, **20**, 561–580.

Gu, C. (2013), *Smoothing Spline ANOVA Models (2nd Ed)*. New York: Springer-Verlag.

Gu, C. (2014), Smoothing Spline ANOVA Models: R Package gss. *Journal of Statistical Software*, 58(5), 1-25. URL http://www.jstatsoft.org/v58/i05/.

Examples

```
## Model with interaction
data(gastric)
gastric.fit <- sshzd(Surv(futime,status)~futime*trt,data=gastric)
## exp(-Lambda(600)), exp(-(Lambda(1200)-Lambda(600))), and exp(-Lambda(1200))
survexp.sshzd(gastric.fit,c(600,1200,1200),data.frame(trt=as.factor(1)),c(0,600,0))
## Clean up
## Not run: rm(gastric,gastric.fit)
dev.off()
## End(Not run)

## THE FOLLOWING EXAMPLE IS TIME-CONSUMING
## Proportional hazard model
## Not run:
data(stan)
stan.fit <- sshzd(Surv(futime,status)~futime+age,data=stan)
## Evaluate fitted hazard
hzdrate.sshzd(stan.fit,data.frame(futime=c(10,20),age=c(20,30)))
## Plot lambda(t,age=20)
tt <- seq(0,60,leng=101)
hh <- hzdcurve.sshzd(stan.fit,tt,data.frame(age=20))
plot(tt,hh,type="l")
## Clean up
rm(stan,stan.fit,tt,hh)
dev.off()

## End(Not run)
```

---

sshzd2d *Estimating 2-D Hazard Function Using Smoothing Splines*

Description

Estimate 2-D hazard function using smoothing spline ANOVA models.

Usage

```
sshzd2d(formula1, formula2, symmetry=FALSE, data, alpha=1.4,
        weights=NULL, subset=NULL, id.basis=NULL, nbasis=NULL, seed=NULL,
        prec=1e-7, maxiter=30, skip.iter=FALSE)

sshzd2d1(formula1, formula2, symmetry=FALSE, data, alpha=1.4,
         weights=NULL, subset=NULL, rho="marginal",
         id.basis=NULL, nbasis=NULL, seed=NULL, prec=1e-7, maxiter=30,
         skip.iter=FALSE)
```

Arguments

`formula1`	Description of the hazard model to be fit on the first axis.
`formula2`	Description of the hazard model to be fit on the second axis.
`symmetry`	Flag indicating whether to enforce symmetry of the two axes.
`data`	Data frame containing the variables in the model.
`alpha`	Parameter defining cross-validation scores for smoothing parameter selection.
`weights`	Optional vector of counts for duplicated data.
`subset`	Optional vector specifying a subset of observations to be used in the fitting process.
`id.basis`	Index of observations to be used as "knots."
`nbasis`	Number of "knots" to be used. Ignored when `id.basis` is specified.
`seed`	Seed to be used for the random generation of "knots." Ignored when `id.basis` is specified.
`prec`	Precision requirement for internal iterations.
`maxiter`	Maximum number of iterations allowed for internal iterations.
`skip.iter`	Flag indicating whether to use initial values of theta and skip theta iteration in marginal hazard estimation.
`rho`	Choice of rho function for sshzd2d1: `"marginal"` or `"weibull"`.

Details

The 2-D survival function is expressed as $S(t1,t2)=C(S1(t1),S2(t2))$, where $S1(t1)$, $S2(t2)$ are marginal survival functions and $C(u1,u2)$ is a 2-D copula. The marginal survival functions are estimated via the marginal hazards as in `sshzd`, and the copula is estimated nonparametrically by calling `sscopu2`.

When `symmetry=TRUE`, a common marginal survial function S1(t)=S2(t) is estimated, and a symmetric copula is estimated such that $C(u1,u2)=C(u2,u1)$.

Covariates can be incorporated in the marginal hazard models as in `sshzd`, including parametric terms via `partial` and frailty terms via `random`. Arguments `formula1` and `formula2` are typically model formulas of the same form as the argument `formula` in `sshzd`, but when `partial` or `random` are needed, `formula1` and `formula2` should be lists with model formulas as the first elements and `partial/random` as named elements; when necessary, variable configurations (that are done via argument `type` in `sshzd`) should also be entered as named elements of lists `formula1/formula2`.

When `symmetry=TRUE`, parallel model formulas must be consistent of each other, such as

```
formula1=list(Surv(t1,d1)~t1*u1,partial=~z1,random=~1|id1)
 formula2=list(Surv(t2,d2)~t2*u2,partial=~z2,random=~1|id2)
```

where pairs `t1`-`t2`, `d2`-`d2` respectively are different elements in `data`, pairs `u1`-`u2`, `z1`-`z2` respectively may or may not be different elements in `data`, and factors `id1` and `id2` are typically the same but at least should have the same levels.

Value

`sshzd2d` and `sshzd2d1` return a list object of class `"sshzd2d"`.

`hzdrate.sshzd2d` can be used to evaluate the estimated 2-D hazard function. `survexp.sshzd2d` can be used to calculate estimated survival functions.

Note

`sshzd2d1` executes faster than `sshzd2d`, but often at the cost of performance degradation.

The results may vary from run to run. For consistency, specify `id.basis` or set `seed`.

Author(s)

Chong Gu, <`chong@stat.purdue.edu`>

References

Gu, C. (2015), Hazard estimation with bivariate survival data and copula density estimation. *Journal of Computational and Graphical Statistics*, **24**, 1053-1073.

Examples

```
## THE FOLLOWING EXAMPLE IS TIME-CONSUMING
## Not run:
data(DiaRet)
## Common proportional hazard model on the margins
fit <- sshzd2d(Surv(time1,status1)~time1+trt1*type,
               Surv(time2,status2)~time2+trt2*type,
               data=DiaRet,symmetry=TRUE)
## Evaluate fitted survival and hazard functions
time <- cbind(c(50,70),c(70,70))
cova <- data.frame(trt1=as.factor(c(1,1)),trt2=as.factor(c(1,0)),
                   type=as.factor(c("juvenile","adult")))
survexp.sshzd2d(fit,time,cov=cova)
hzdrate.sshzd2d(fit,time,cov=cova)
## Association between margins: Kendall's tau and Spearman's rho
summary(fit$copu)
## Clean up
rm(DiaRet,fit,time,cova)
dev.off()

## End(Not run)
```

`ssllrm` *Fitting Smoothing Spline Log-Linear Regression Models*

Description

Fit smoothing spline log-linear regression models. The symbolic model specification via `formula` follows the same rules as in `lm`.

Usage

```
ssllrm(formula, response, type=NULL, data=list(), weights, subset,
       na.action=na.omit, alpha=1, id.basis=NULL, nbasis=NULL,
       seed=NULL, random=NULL, prec=1e-7, maxiter=30, skip.iter=FALSE)
```

Arguments

`formula`	Symbolic description of the model to be fit.
`response`	Formula listing response variables.
`type`	List specifying the type of spline for each variable. See `mkterm` for details.
`data`	Optional data frame containing the variables in the model.
`weights`	Optional vector of weights to be used in the fitting process.
`subset`	Optional vector specifying a subset of observations to be used in the fitting process.
`na.action`	Function which indicates what should happen when the data contain NAs.

`alpha`	Parameter modifying GCV or Mallows' CL; larger absolute values yield smoother fits; negative value invokes a stable and more accurate GCV/CL evaluation algorithm but may take two to five times as long. Ignored when `method="m"` are specified.
`id.basis`	Index designating selected "knots".
`nbasis`	Number of "knots" to be selected. Ignored when `id.basis` is supplied.
`seed`	Seed to be used for the random generation of "knots". Ignored when `id.basis` is supplied.
`random`	Input for parametric random effects in nonparametric mixed-effect models. See `mkran` for details.
`prec`	Precision requirement for internal iterations.
`maxiter`	Maximum number of iterations allowed for internal iterations.
`skip.iter`	Flag indicating whether to use initial values of theta and skip theta iteration. See `ssanova` for notes on skipping theta iteration.

Details

The model is specified via `formula` and `response`, where `response` lists the response variables. For example, `ssllrm(~y1*y2*x,~y1+y2)` prescribe a model of the form

$$logf(y1,y2|x) = g_1(y1)+g_2(y2)+g_{12}(y1,y2)+g_{x1}(x,y1)+g_{x2}(x,y2)+g_{x12}(x,y1,y2)+C(x)$$

with the terms denoted by `"y1"`, `"y2"`, `"y1:y2"`, `"y1:x"`, `"y2:x"`, and `"y1:y2:x"`; the term(s) not involving response(s) are removed and the constant `C(x)` is determined by the fact that a conditional density integrates (adds) to one on the `y` axis.

The model terms are sums of unpenalized and penalized terms. Attached to every penalized term there is a smoothing parameter, and the model complexity is largely determined by the number of smoothing parameters.

A subset of the observations are selected as "knots." Unless specified via `id.basis` or `nbasis`, the number of "knots" q is determined by $max(30, 10n^{2/9})$, which is appropriate for the default cubic splines for numerical vectors.

Value

`ssllrm` returns a list object of class `"ssllrm"`.

The method `predict.ssllrm` can be used to evaluate `f(y|x)` at arbitrary x, or contrasts of `log{f(y|x)}` such as the odds ratio along with standard errors. The method `project.ssllrm` can be used to calculate the Kullback-Leibler projection for model selection.

Note

The responses, or y-variables, must be factors, and there must be at least one numerical x's. For `response`, there is no difference between `~y1+y2` and `~y1*y2`.

The results may vary from run to run. For consistency, specify `id.basis` or set `seed`.

References

Gu, C. and Ma, P. (2011), Nonparametric regression with cross-classified responses. *The Canadian Journal of Statistics*, **39**, 591–609.

Gu, C. (2014), Smoothing Spline ANOVA Models: R Package gss. *Journal of Statistical Software*, 58(5), 1-25. URL http://www.jstatsoft.org/v58/i05/.

Examples

```
## Simulate data
test <- function(x)
        {.3*(1e6*(x^11*(1-x)^6)+1e4*(x^3*(1-x)^10))-2}
x <- (0:100)/100
p <- 1-1/(1+exp(test(x)))
y <- rbinom(x,3,p)
y1 <- as.ordered(y)
y2 <- as.factor(rbinom(x,1,p))
## Fit model
fit <- ssllrm(~y1*y2*x,~y1+y2)

## Evaluate f(y|x)
est <- predict(fit,data.frame(x=x),
               data.frame(y1=as.factor(0:3),y2=as.factor(rep(0,4))))
## f(y|x) at all y values (fit$qd.pt)
est <- predict(fit,data.frame(x=x))

## Evaluate contrast of log f(y|x)
est <- predict(fit,data.frame(x=x),odds=c(-1,.5,.5,0),
               data.frame(y1=as.factor(0:3),y2=as.factor(rep(0,4))),se=TRUE)
## Odds ratio log{f(0,0|x)/f(3,0|x)}
est <- predict(fit,data.frame(x=x),odds=c(1,-1),
               data.frame(y1=as.factor(c(0,3)),y2=as.factor(c(0,1))),se=TRUE)

## KL projection
kl <- project(fit,include=c("y2:x","y1:y2","y1:x","y2:x"))

## Clean up
## Not run: rm(test,x,p,y,y1,y2,fit,est,kl)
dev.off()
## End(Not run)
```

`stan` *Stanford Heart Transplant Data*

Description

Survival of patients from the Stanford heart transplant program.

Usage

```
data(stan)
```

Format

A data frame containing 184 observations on the following variables.

`time`	Follow-up time after transplant, in days.
`status`	Censoring status.
`age`	Age at transplant.
`futime`	Square root of `time`.

Source

Miller, R. G. and Halpern, J. (1982), Regression with censored data. *Biometrika*, **69**, 521–531.

`summary.gssanova` *Assessing Smoothing Spline ANOVA Fits with Non-Gaussian Responses*

Description

Calculate various summaries of smoothing spline ANOVA fits with non-Gaussian responses.

Usage

```
## S3 method for class 'gssanova'
summary(object, diagnostics=FALSE, ...)
```

Arguments

`object`	Object of class `"gssanova"`.
`diagnostics`	Flag indicating if diagnostics are required.
`...`	Ignored.

Details

Similar to the iterated weighted least squares fitting of `glm`, penalized likelihood regression fit can be calculated through iterated penalized weighted least squares.

The diagnostics are based on the "pseudo" Gaussian response model behind the weighted least squares problem at convergence.

Value

`summary.gssanova` returns a list object of `class` `"summary.gssanova"` consisting of the following elements. The entries `pi`, `kappa`, `cosines`, and `roughness` are only calculated if `diagnostics=TRUE`.

`call`	Fitting call.
`family`	Error distribution.
`alpha`	Parameter used to define cross-validation in model fitting.

`fitted`	Fitted values on the link scale.
`dispersion`	Assumed or estimated dispersion parameter.
`residuals`	Working residuals on the link scale.
`rss`	Residual sum of squares.
`dev.resid`	Deviance residuals.
`deviance`	Deviance of the fit.
`dev.null`	Deviance of the null model.
`penalty`	Roughness penalty associated with the fit.
`pi`	"Percentage decomposition" of "explained variance" into model terms.
`kappa`	Concurvity diagnostics for model terms. Virtually the square roots of variance inflation factors of a retrospective linear model.
`cosines`	Cosine diagnostics for practical significance of model terms.
`roughness`	Percentage decomposition of the roughness penalty `penalty` into model terms.

References

Gu, C. (1992), Diagnostics for nonparametric regression models with additive terms. *Journal of the American Statistical Association*, **87**, 1051–1058.

See Also

Fitting function `gssanova` and methods `predict.ssanova`, `project.gssanova`, `fitted.gssanova`.

`summary.gssanova0` *Assessing Smoothing Spline ANOVA Fits with Non-Gaussian Responses*

Description

Calculate various summaries of smoothing spline ANOVA fits with non-Gaussian responses.

Usage

```
## S3 method for class 'gssanova0'
summary(object, diagnostics=FALSE, ...)
```

Arguments

`object`	Object of class `"gssanova"`.
`diagnostics`	Flag indicating if diagnostics are required.
`...`	Ignored.

Details

Similar to the iterated weighted least squares fitting of `glm`, penalized likelihood regression fit can be calculated through iterated penalized weighted least squares.

The diagnostics are based on the "pseudo" Gaussian response model behind the weighted least squares problem at convergence.

Value

`summary.gssanova0` returns a list object of `class` `"summary.gssanova0"` consisting of the following elements. The entries `pi`, `kappa`, `cosines`, and `roughness` are only calculated if `diagnostics=TRUE`.

`call`	Fitting call.
`family`	Error distribution.
`method`	Method for smoothing parameter selection.
`dispersion`	Assumed or estimated dispersion parameter.
`iter`	Number of performance-oriented iterations performed.
`fitted`	Fitted values on the link scale.
`residuals`	Working residuals on the link scale.
`rss`	Residual sum of squares.
`dev.resid`	Deviance residuals.
`deviance`	Deviance of the fit.
`dev.null`	Deviance of the null model.
`alpha`	Estimated size for `family="nbinomial"` with one column responses. Estimated inverse scale of log life time for `family="nbinomial"`, `"lognorm"`, or `"loglogis"`.
`penalty`	Roughness penalty associated with the fit.
`pi`	"Percentage decomposition" of "explained variance" into model terms.
`kappa`	Concurvity diagnostics for model terms. Virtually the square roots of variance inflation factors of a retrospective linear model.
`cosines`	Cosine diagnostics for practical significance of model terms.
`roughness`	Percentage decomposition of the roughness penalty `penalty` into model terms.

References

Gu, C. (1992), Diagnostics for nonparametric regression models with additive terms. *Journal of the American Statistical Association*, **87**, 1051–1058.

See Also

Fitting function `gssanova0` and methods `predict.ssanova0`, `fitted.gssanova`.

`summary.ssanova` *Assessing Smoothing Spline ANOVA Fits*

Description

Calculate various summaries of smoothing spline ANOVA fits.

Usage

```
## S3 method for class 'ssanova'
summary(object, diagnostics=FALSE, ...)
## S3 method for class 'ssanova0'
summary(object, diagnostics=FALSE, ...)
## S3 method for class 'ssanova9'
summary(object, diagnostics=FALSE, ...)
```

Arguments

`object`	Object of class `"ssanova"`.
`diagnostics`	Flag indicating if diagnostics are required.
`...`	Ignored.

Value

`summary.ssanova` returns a list object of `class` `"summary.ssanova"` consisting of the following elements. The entries `pi`, `kappa`, `cosines`, and `roughness` are only calculated if `diagnostics=TRUE`; see the reference below for details concerning the diagnostics.

`call`	Fitting call.
`method`	Method for smoothing parameter selection.
`fitted`	Fitted values.
`residuals`	Residuals.
`sigma`	Assumed or estimated error standard deviation.
`r.squared`	Fraction of "explained variance" by the fitted model.
`rss`	Residual sum of squares.
`penalty`	Roughness penalty associated with the fit.
`pi`	"Percentage decomposition" of "explained variance" into model terms.
`kappa`	Concurvity diagnostics for model terms. Virtually the square roots of variance inflation factors of a retrospective linear model.
`cosines`	Cosine diagnostics for practical significance of model terms.
`roughness`	Percentage decomposition of the roughness penalty `penalty` into model terms.

References

Gu, C. (1992), Diagnostics for nonparametric regression models with additive terms. *Journal of the American Statistical Association*, **87**, 1051–1058.

See Also

Fitting functions `ssanova`, `ssanova0` and methods `predict.ssanova`, `project.ssanova`, `fitted.ssanova`.

`summary.sscopu` *Calculating Kendall's Tau and Spearman's Rho for 2-D Copula Density Estimates*

Description

Calculate Kendall's tau and Spearman's rho for 2-D copula density estimates.

Usage

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

Arguments

`object`	Object of class `"sscopu"`.
`...`	Ignored.

Value

A list containing Kendall's tau and Spearman's rho.

See Also

Fitting functions `sscopu` and `sscopu2`.

`wesdr` *Progression of Diabetic Retinopathy*

Description

Data derived from the Wisconsin Epidemiological Study of Diabetic Retinopathy.

Usage

```
data(wesdr)
```

Format

A data frame containing 669 observations on the following variables.

`dur`	Duration of diabetes at baseline, in years.
`gly`	Percent of glycosylated hemoglobin at baseline.
`bmi`	Body mass index at baseline.
`ret`	Binary indicator of retinopathy progression at first follow-up.

Source

Wang, Y. (1997), GRKPACK: Fitting smoothing spline ANOVA models for exponential families. *Communications in Statistics – Simulations and Computation*, **26**, 765–782.

References

Klein, R., Klein, B. E. K., Moss, S. E., Davis, M. D., and DeMets, D. L. (1988), Glycosylated hemoglobin predicts the incidence and progression of diabetic retinopathy. *Journal of the American Medical Association*, **260**, 2864–2871.

Klein, R., Klein, B. E. K., Moss, S. E., Davis, M. D., and DeMets, D. L. (1989), The Wisconsin Epidemiologic Study of Diabetic Retinopathy. X. Four incidence and progression of diabetic retinopathy when age at diagnosis is 30 or more years. *Archive Ophthalmology*, **107**, 244–249.

Wahba, G., Wang, Y., Gu, C., Klein, R., and Klein, B. E. K. (1995), Smoothing spline ANOVA for exponential families, with application to the Wisconsin Epidemiological Study of Diabetic Retinopathy. *The Annals of Statistics*, **23**, 1865–1895.

`wesdr1` *Stages of Diabetic Retinopathy*

Description

Data derived from the Wisconsin Epidemiological Study of Diabetic Retinopathy.

Usage

```
data(wesdr1)
```

Format

A data frame containing 2049 observations on the following variables.

`age`	Age of patient.
`dur`	Duration of diabetes, in years.
`gly`	Percent of glycosylated hemoglobin.
`upro`	Ordinal urine protein level.
`insl`	Binary indicator of insulin usage.
`ret1`	Ordinal retinopathy stage, right eye.
`ret2`	Ordinal retinopathy stage, left eye.

Index