Package ‘ssym’

January 27, 2015

Type Package

Title Fitting Semi-parametric Log-symmetric Regression Models

Version 1.5.2

Date 2015-01-08

Author Luis Hernando Vanegas <hvanegasp@gmail.com> and Gilberto A. Paula

Maintainer Luis Hernando Vanegas <hvanegasp@gmail.com>

Description This package allows to to fit a semi-parametric regression model suitable for analysis of data sets in which the response variable is continuous, strictly positive, and asymmetric. In this setup, both median and skewness of the response variable distribution are explicitly modeled through semi-parametric functions, whose nonparametric components may be approximated by natural cubic splines or P-splines. Supported distributions for the model error include log-normal, log-Student-t, log-power-exponential, log-hyperbolic, log-contaminated-normal, log-slash, Birnbaum-Saunders and Birnbaum-Saunders-t distributions.

License GPL-2 | GPL-3

Depends GIGrvg, numDeriv, splines, normalp, Formula

Suggests NISTnls, gam, sn

NeedsCompilation no

Repository CRAN

Date/Publication 2015-01-08 17:02:06

R topics documented:

`ssym-package` *Fitting Semiparametric Log-symmetric Regression Models*

Description

This package allows to fit a semi-parametric regression model suitable for analysis of data sets in which the response variable is continuous, strictly positive, and asymmetric.

Details

Package:	ssym
Type:	Package
Version:	1.5.2
Date:	2015-01-08
License:	GPL-2 | GPL-3

Author(s)

Luis Hernando Vanegas <hvanegasp@gmail.com> and Gilberto A. Paula

Maintainer: Luis Hernando Vanegas

References

Vanegas, L.H. and Paula, G.A. (2015a) A Semiparametric Approach for Joint Modeling of Median and Skewness. TEST (to appear)

Vanegas, L.H. and Paula, G.A. (2015b) Log-symmetric distributions: statistical properties and parameter estimation. Brazilian Journal of Probability and Statistics (to appear)

Examples

```
data("Snacks", package="ssym")
fit <- ssym.l(log(texture) ~ type + ncs(week) | type, data=Snacks,
                 family='Student', xi=15)
summary(fit)
```

`AIC.ssym` *AIC.ssym*

Description

AIC.ssym calculates the goodness-of-fit statistic AIC from an object of class “"ssym".

`Biaxial` *Brown and Miller’s Biaxial Fatigue*

Description

This data set describes the life of a metal piece in cycles to failure. The response is the number of cycles to failure and the explanatory variable is the work per cycle.

Usage

```
data(Biaxial)
```

Format

A data frame with 46 observations on the following 2 variables.

`Work` work per cycle.

`Life` number of cycles to failure.

References

J.R. Rieck and J.R. Nedelman (1991) A log-linear model for the Birnbaum-Saunders distribution, Technometrics 33, 51:60.

Examples

```
data("Biaxial", package="ssym")
plot(Biaxial$Work, Biaxial$Life, xlim=range(Biaxial$Work), ylim=range(Biaxial$Life), type="p",
     cex=0.3, lwd=3, ylab="Life", xlab="Work per cycle",
 main="Brown and Miller's Biaxial Fatigue Data")
```

BIC.ssym *BIC.ssym*

Description

BIC.ssym calculates the goodness-of-fit statistic BIC from an object of class “"ssym".

Claims *Personal Injure Insurance*

Description

This data set contains information on 540 settled personal injury insurance claims from an Australian insurance company, which is a sample of the original data set. These claims had legal representation were obtained for accidents that occurred from January 1998 to June 1999.

Usage

```
data(Claims)
```

Format

A data frame with 540 observations on the following 2 variables.

`total` amount of paid money by an insurance policy in thousands of Australian dollars.

`accmonth` month of occurrence of the accident coded 103 (January 1998) through to 120 (June 1999).

`op_time` operational time in percentage.

References

de Jong P, Heller GZ. Generalized Linear Models for Insurance Data. Cambridge University Press: Cambridge, England, 2008.

Paula, G.A., Leiva, V., Barros, M. and Liu, S. (2012) Robust statistical modeling using the Birnbaum-Saunders-t distribution applied to insurance distribution, Applied Stochastic Model in Business and Industry, 28:16-34.

Examples

```
data("Claims", package="ssym")
plot(Claims$op_time, Claims$total, xlim=range(Claims$op_time), ylim=range(Claims$total),
     type="p", cex=0.3, lwd=3, ylab="Amount of paid money",xlab="Operational time",
 main="Personal Injure Insurance Data")
```

---

coef.ssym *coef.ssym*

Description

coef.ssym extracts the parameter estimates for both submodels from an object of class “"ssym".

Erabbits *Age and Eye Lens Weight for Rabbits in Australia*

Description

In this data set, the dry weight of the eye lens was measured for 71 free-living wild rabbits of known age. Eye lens weight tends to vary much less with environmental conditions than does total body weight, and therefore may be a much better indicator of age.

Usage

```
data(Erabbits)
```

Format

A data frame with 71 observations on the following 2 variables.

`age` age of rabbit in days.

`wlens` dry weight of eye lens in milligrams.

References

Dudzinski, M.L. and Mykytowycz, R. (1961) The eye lens as an indicator of age in the wild rabbit in Australia. CSIRO Wildlife Research, 6: 156-159.

Ratkowsky, D. A. (1983). Nonlinear Regression Modelling. Marcel Dekker, New York.

Wei, B. C. (1998). Exponential Family Nonlinear Models. Springer, Singapore.

Examples

```
data("Erabbits", package="ssym")
plot(Erabbits$age, Erabbits$wlens, xlim=range(Erabbits$age), ylim=range(Erabbits$wlens),
     type="p", cex=0.3, lwd=3, ylab="Dry weight of eye lens (in milligrams)",
 xlab="Age of the animal (in days)")
```

`estfun.ssym` *estfun.ssym*

Description

estfun.ssym extracts the score functions evaluated at observed data and estimated parameters from an object of class “"ssym".

`fitted.ssym` *fitted.ssym*

Description

fitted.ssym extracts the fitted values for both submodels from an object of class “"ssym".

`gdp` *Gross Domestic Product (per capita)*

Description

This dataset corresponds to the per capita gross domestic product (current US$) of 190 countries during 2010.

Usage

```
data(gdp)
```

Format

A data frame with 190 observations on the following 2 variables.

`Country` Country.

`gdp2010` The per capita gross domestic product (current US$).

References

World Bank’s DataBank website (http://databank.worldbank.org/data/).

Examples

```
data("gdp", package="ssym")
par(mfrow=c(1,2))
hist(gdp$gdp2010, xlim=range(gdp$gdp2010), ylim=c(0,0.00015), prob=TRUE, breaks=55,
     col="light gray",border="dark gray", xlab="GDP per capita 2010", main="Histogram")
plot(ecdf(gdp$gdp2010), xlim=range(gdp$gdp2010), ylim=c(0,1), verticals=TRUE,
     do.points=FALSE, col="dark gray", xlab="GDP per capita 2010",
 main="Empirical Cumulative Distribution Function")
```

`influence.ssym` *Tool to perform sensitivity analysis on the fitted model using local influence measures.*

Description

influence extracts from a object of class “ssym" the local influence measures and displays their graphs versus the index of the observations.

Author(s)

Luis Hernando Vanegas <hvanegasp@gmail.com> and Gilberto A. Paula

References

Cook, R.D. (1986). Assessment Local Influence (with discussion). Journal of the Royal Statistical Society Series B (Methodological). 48, 133-169.

Poon, W.Y. and Poon, Y.S. (1999). Conformal Normal Curvature and Assessment of Local Influence. Journal of the Royal Statistical Society Series B (Methodological). 61, 51-61.

`itpE` *itpE*

Description

itpE runs the E-step of the iterative process to fit models whose error distribution can be obtained as a shape mixture of the log-normal distribution.

`itpE2` *itpE2*

Description

itpE2 runs the E-step of the iterative process to fit models whose error distribution can be obtained as a shape mixture of the Birnbaum-Saunders distribution.

`itpE3` *itpE3*

Description

itpE3 runs the iterative process to fit models whose error distribution cannot be obtained as a shape mixture of log-normal or Birnbaum-Saunders distributions.

itpM	*itpM*

Description

itpM runs the M-step of the iterative process to fit models whose error distribution can be obtained as a shape mixture of log-normal distribution.

itpM2	*itpM2*

Description

itpM2 runs the M-step of the iterative process to fit models whose error distribution can be obtained as a shape mixture of Birnbaum-Saunders distribution.

itpM3	*itpM3*

Description

itpM3 calls the iterative process to fit models whose error distribution cannot be obtained as a shape mixture of log-normal or Birnbaum-Saunders distributions.

lambda.hat	*Tool to choose smoothing parameters.*

Description

lambda.hat is used to choose the smoothing parameter using the unweighted cross-validation score.

Usage

```
lambda.hat(response,xx,lambda,type,plot)
```

Arguments

`response`	the response variable.
`xx`	the explanatory variable.
`lambda`	an optional numeric value of starting estimate for the smoothing parameter. Default is 1.
`type`	an integer indicating the type of nonparametric effect, e.g., 1 indicates natural cubic spline, and 2 indicates P-spline.
`plot`	logical. If *plot=TRUE*, it shows a graph with the shape of the cross-validation score.

Value

`lh`	a list with the chosen smoothing parameter and the value of the cross-validation score.

Author(s)

Luis Hernando Vanegas <hvanegasp@gmail.com> and Gilberto A. Paula

References

Eilers, P.H.C. and Marx, B.D. (1996). Flexible smoothing with B-splines and penalties. Statistical Science, 11, 89-121.

Green, P.J. and Silverman, B.W. (1994) Nonparametric Regression and Generalized Linear Models, Boca Raton: Chapman and Hall.

Examples

```
n <- 300
t <- sort(round(runif(n),digits=2))
y <- cos(4*pi*t) + rnorm(n)
datas <- data.frame(t,y)

par(mfrow=c(1,2))
lambda <- lambda.hat(y,t,1,1,plot=TRUE)
lambda <- lambda$lambda_hat
fit <- ssym.l(y ~ ncs(t, lambda=lambda), family="Normal", data=datas)
np.graph(fit, which=1, main="Natural Cubic Spline")

#x11()
par(mfrow=c(1,2))
lambda <- lambda.hat(y,t,1,2,plot=TRUE)
lambda <- lambda$lambda_hat
fit2 <- ssym.l(y ~ psp(t, lambda=lambda), family="Normal", data=datas)
np.graph(fit2, which=1, main="P-spline")
```

logLik.ssym — *logLik.ssym*

Description

logLik.ssym extracts the value of the log-likelihood function avaliated at observed data and parameter estimates from an object of class “"ssym".

ncs — *Tool to build incidence and penalty matrices, which can be used to approximate smooth functions by natural cubic splines.*

Description

ncs builds the incidence and penalty matrices.

Usage

```
ncs(xx, lambda)
```

Arguments

`xx`	the explanatory variable.
`lambda`	an optional positive value, which corresponds to the smoothing parameter.

Value

`xx`	a matrix with the following attributes: incidence matrix, penalty matrix, smoothing parameter (if it has been specified), and another interest matrices.

Author(s)

Luis Hernando Vanegas <hvanegasp@gmail.com> and Gilberto A. Paula

References

Green, P.J. and Silverman, B.W. (1994) Nonparametric Regression and Generalized Linear Models, Boca Raton: Chapman and Hall.

Examples

```
n <- 300
t <- sort(round(runif(n),digits=1))

t2 <- ncs(t)
N <- attr(t2, "N") ## Incidence Matrix
M <- attr(t2, "K") ## Penalty Matrix
```

`np.graph` *Tool for plotting natural cubic splines and P-splines*

Description

np.graph displays a graph of a fitted non-parametric effect (either natural cubic spline or P-spline) from a object of class “ssym".

Usage

```
np.graph(object, which, exp, xlab, ylab, main)
```

Arguments

`object`	object of class “ssym" produced by *ssym.l()* or *ssym.nl()*.
`which`	an integer indicating the interest submodel, e.g., 1 indicates location submodel, and 2 indicates dispersion submodel.
`exp`	logical. Should the fitted non-parametric effect be plotted in exponential scale? Default is *FALSE*.
`xlab`	an optional title for the *x* axis.
`ylab`	an optional title for the *y* axis.
`main`	an optional overall title for the plot.

Author(s)

Luis Hernando Vanegas <hvanegasp@gmail.com> and Gilberto A. Paula

References

Green, P.J. and Silverman, B.W. (1994) Nonparametric Regression and Generalized Linear Models, Boca Raton: Chapman and Hall. Eilers P.H.C. and Marx B.D. (1996). Flexible smoothing with B-splines and penalties. Statistical Science. 11, 89-121.

Examples

```
data("Ovocytes", package="ssym")
fit <- ssym.l(fraction ~ type + psp(time) | type + psp(time), data=Ovocytes,
              family='Powerexp', xi=-0.65)

par(mfrow = c(1,2))
np.graph(fit, which=1, xlab="Time", main="Location")
np.graph(fit, which=2, exp=TRUE, xlab="Time", main="Dispersion")
```

Ovocytes *Fraction of cell volume*

Description

This data set comes from an experiment comparing the responses of immature and mature goat ovocytes to an hyper-osmotic test. As a compound permeates, water reenters the cell, and the cell re-expands until the system reaches an osmotic equilibrium. The results are obtained using immature and ovulated (mature) ovocytes exposed to propanediol, a permeable compound. Then, the cell volume during equilibration is recorded at each time *t*.

Usage

```
data(Ovocytes)
```

Format

A data frame with 161 observations on the following 3 variables.

`type` stage of the goat ovocyte: *Mature* or *Immature*.

`time` time since exposition to propanediol.

`fraction` fraction of initial isotonic cell volume at any given time *t* during equilibration.

References

Huet, S., Bouvier, A., Gruet, M.A. and Jolivet, E. (1996). Statistical Tools for Nonlinear Regression. Springer, New York.

Le Gal F., Gasqui P., Renard J.P. (1994) Differential Osmotic Behavior of Mammalian Oocytes before and after Maturation: A Quantitative Analysis Using Goat Oocytes as a Model. Cryobiology, 31: 154-170.

Huet S., Bouvier A., Gruet M.A., Jolivet E. (1996) Statistical Tools for Nonlinear Regression. Springer-Verlag: New York.

Examples

```
data("Ovocytes", package="ssym")
xl <- "Time"
yl <- "Fraction of Cell Volume"
mm <- "Fraction of Cell Volume for Mature and Immature Goat Ovocytes"
rx <- range(Ovocytes$time)
ry <- range(Ovocytes$fraction)
plot(Ovocytes$time[Ovocytes$type=='Mature'], Ovocytes$fraction[Ovocytes$type=='Mature'],
     xlim=rx, ylim=ry, type="p", cex=0.5, lwd=1, ylab="", xlab="")
par(new=TRUE)
plot(Ovocytes$time[Ovocytes$type=='Immature'], Ovocytes$fraction[Ovocytes$type=='Immature'],
     xlim=rx, ylim=ry, type="p", cex=0.5, lwd=2, ylab=yl, xlab=xl, main=mm)
legend(rx[1], ry[2], pt.lwd=c(1,2), bty="n", legend=c("Mature","Immature"), pt.cex=0.5, pch=1)
```

plot.ssym	*plot.ssym*

Description

plot.ssym displays graphs of standardized individual-specific weights, overall goodness-of-fit statistic and deviance-type residuals from an object of class “"ssym".

print.ssym	*print.ssym*

Description

print.ssym displays a summary (simpler than summary.ssym) of the fitted model including parameter estimates, associated (approximated) standard errors and goodness-of-fit statistics from an object of class “"ssym".

psp	*Tool to build B-spline basis and penalty matrices, which can be used to approximate functions by P-splines.*

Description

psp builds the B-spline basis and penalty matrices.

Usage

```
psp(xx, lambda, nknots, diff)
```

Arguments

`xx`	the explanatory variable.
`lambda`	an optional positive value, which corresponds to the smoothing parameter.
`nknots`	an optional integer specifying the number of internal knots. Defaul is $[n^(1/3)]$.
`diff`	an optional integer specifying the order of the difference penalty term. Default is 2.

Value

`xx`	a matrix with the following attributes: B-spline basis matrix, penalty matrix, and smoothing parameter (if it has been specified).

Author(s)

Luis Hernando Vanegas <hvanegasp@gmail.com> and Gilberto A. Paula

References

Eilers P.H.C. and Marx B.D. (1996). Flexible smoothing with B-splines and penalties. Statistical Science. 11, 89-121.

Examples

```
n <- 300
t <- sort(round(runif(n),digits=2))

t2 <- psp(t, diff=3)
N <- attr(t2, "N") ## B-spline basis matrix
M <- attr(t2, "K") ## Penalty Matrix
```

`residuals.ssym` *residuals.ssym*

Description

residuals.ssym extracts the deviance-type residuals for both submodels from an object of class “"ssym".

`rvgs` *Random generation for some symmetric continuous distributions.*

Description

rvgs is used to random generation from some standard symmetric continuous distributions.

Usage

```
rvgs(n, family, xi)
```

Arguments

`n`	number of observations.
`family`	Supported families include *Normal*, *Student*, *Contnormal*, *Powerexp*, *Hyperbolic*, *Slash*, *Sinh-normal* and *Sinh-t*, which correspond to normal, Student-t, contaminated normal, power exponential, symmetric hyperbolic, slash, sinh-normal and sinh-t distributions, respectively.
`xi`	a numeric value or numeric vector that represents the extra parameter of the specified distribution.

Value

`x` a vector of n observations.

Author(s)

Luis Hernando Vanegas <hvanegasp@gmail.com> and Gilberto A. Paula

Examples

```
m1 <- "Standard Sinh-t distributions"
n <- 1000000
xi <- c(10,6,4)
plot(density(rvgs(n,"Sinh-t",xi=c(25,10))), xlim=c(-4.5,4.5), ylim=c(0,0.3), xlab="",
     ylab="", col=1, main="")
par(new=TRUE)
plot(density(rvgs(n,"Sinh-t",xi=c(25,6))), xlim=c(-4.5,4.5), ylim=c(0,0.3), xlab="",
     ylab="", col=2, main="")
par(new=TRUE)
plot(density(rvgs(n,"Sinh-t",xi=c(25,4))), xlim=c(-4.5,4.5), ylim=c(0,0.3), xlab="y",
     ylab="f(y)", main=m1, col=3)
legend(-4, 0.3, bty="n", legend=paste("xi = (",25,",",xi,")"), col=1:4, lty=1)
```

`Snacks` *Textures of five different types of snacks*

Description

This data set comes from an experiment developed in the School of Public Health - University of S\~ao Paulo, in which four different forms of light snacks (denoted by B, C, D, and E) were compared with a traditional snack (denoted by A) for 20 weeks. For the light snacks, the hydrogenated vegetable fat (hvf) was replaced by canola oil using different proportions: B (0% hvf, 22% canola oil), C (17% hvf, 5% canola oil), D (11% hvf, 11% canola oil) and E (5% hvf, 17% canola oil); A (22% hvf, 0% canola oil) contained no canola oil. The experiment was conducted such that a random sample of 15 units of each snack type was analyzed in a laboratory in each even week to measure various variables. A total of 75 units was analyzed in each even week; with 750 units being analyzed during the experiment.

Usage

```
data(Snacks)
```

Format

A data frame with 750 observations on the following 3 variables.

`texture` texture of the snack unit.

`type` a factor with levels 1-5 which correspond to A-E types of snacks.

`week` week in which the snack unit was analyzed.

References

Paula, G.A., de Moura, A.S., Yamaguchi, A.M. (2004) Sensorial stability of snacks with canola oil and hydrogenated vegetable fat. Technical Report. Center of Applied Statistics, University of Sao Paulo (in Portuguese).

Paula, G.A. (2013) On diagnostics in double generalized linear models. Computational Statistics and Data Analysis, 68: 44-51.

Examples

```
data("Snacks", package="ssym")
boxplot(log(Snacks$texture) ~ Snacks$type, xlab="Type of Snack", ylab="Log(texture)")
```

`ssym.l` *Fitting Semi-parametric Log-symmetric Regression Models*

Description

ssym.l is used to fit a semi-parametric regression model suitable for analysis of data sets in which the response variable is continuous, strictly positive, and asymmetric. In this setup, both median and skewness of the response variable distribution are explicitly modeled through semi-parametric functions, whose nonparametric components may be approximated by natural cubic splines or P-splines.

Usage

```
ssym.l(formula, family, xi, data, epsilon, maxiter, subset, local.influence)
```

Arguments

`formula`	a symbolic description of the systematic component of the model to be fitted. See details for further information.
`family`	a description of the (log) error distribution to be used in the model. Supported families include *Normal*, *Student*, *Contnormal*, *Powerexp*, *Hyperbolic*, *Slash*, *Sinh-normal* and *Sinh-t*, which correspond to normal, Student-t, contaminated normal, power exponential, symmetric hyperbolic, slash, sinh-normal and sinh-t distributions, respectively.
`xi`	a numeric value or numeric vector that represents the extra parameter of the specified error distribution.
`data`	an optional data frame, list or environment containing the variables in the model.
`epsilon`	an optional positive value, which represents the convergence criterion. Default value is 1e-07.
`maxiter`	an optional positive integer giving the maximal number of iterations for the estimating process. Default value is 1e03.
`subset`	an optional expression specifying a subset of individuals to be used in the fitting process.

`local.influence`
: logical. If TRUE, local influence measures under two perturbation schemes are calculated.

Details

The argument *formula* comprises of three parts (separated by the symbols "~" and "|"), namely: observed response variable in log-scale, predictor of the median submodel (having logarithmic link) and predictor of the skewness submodel (having logarithmic link). A non-parametric effect may be specified in the predictors, either approximated by a natural cubic spline or a P-spline using the functions *ncs()* or *psp()*, respectively.

The iterative estimation process is based on the Fisher scoring and backfitting algorithms. Because some distributions such as log-Student-t, log-contaminated-normal, log-power-exponential, log-slash and log-hyperbolic may be obtained as a power mixture of the log-normal distribution, the expectation-maximization (EM) algorithm is applied in those cases to obtain a more efficient iterative process for the parameter estimation. Furthermore, because the Birnbaum-Saunders-t distribution can be obtained as a scale mixture of the Birnbaum-Saunders distribution, the expectation-maximization algorithm is also applied in this case to obtain a more efficient iterative process for the parameter estimation. The smoothing parameter(s) is(are) chosen using the unweighted cross-validation score.

The function *ssym.l()* calculates deviance-type residuals for both submodels as well as local influence measures under case-weight and response perturbation schemes.

Value

`coefs.mu`	a vector of parameter estimates associated with the median submodel.
`coefs.phi`	a vector of parameter estimates associated with the skewness submodel.
`vcov.mu`	approximate variance-covariance matrix associated with the median submodel.
`vcov.phi`	approximate variance-covariance matrix associated with the skewness submodel.
`weights`	final weights of the iterative process.
`lambda.mu`	estimate for the smoothing parameter associated with the nonparametric part of the median submodel.
`dfe.mu`	degrees of freedom associated with the nonparametric part of the median submodel.
`lambda.phi`	estimate for the smoothing parameter associated with the nonparametric part of the skewness submodel.
`dfe.phi`	degrees of freedom associated with the nonparametric part of the skewness submodel.
`deviance.mu`	a vector of *deviances* associated with the median submodel.
`deviance.phi`	a vector of *deviances* associated with the skewness submodel.
`mu.fitted`	a vector of fitted values for the (log) median submodel.
`phi.fitted`	a vector of fitted values for the skewness submodel.
`lpdf`	a vector of individual contributions to the log-likelihood function.

cw	if *local.influence=TRUE*, a matrix of local influence and total local influence measures (under the case-weight perturbation scheme) associated with the median submodel.
pr	if *local.influence=TRUE*, a matrix of local influence and total local influence measures (under the response perturbation scheme) associated with the median submodel.
cw.theta	if *local.influence=TRUE*, a matrix of local influence and total local influence measures (under the case-weight perturbation scheme).
pr.theta	if *local.influence=TRUE*, a matrix of local influence and total local influence measures (under the response perturbation scheme).

Author(s)

Luis Hernando Vanegas <hvanegasp@gmail.com> and Gilberto A. Paula

References

Vanegas, L.H. and Paula, G.A. (2015a) A Semiparametric Approach for Joint Modeling of Median and Skewness. TEST (to appear)

Vanegas, L.H. and Paula, G.A. (2015b) Log-symmetric distributions: statistical properties and parameter estimation. Brazilian Journal of Probability and Statistics (to appear)

See Also

ssym.nl

Examples

```
###################################################################################
######### Fraction of Cell Volume Data - a log-power-exponential model  ###########
###################################################################################

data("Ovocytes", package="ssym")
fit <- ssym.l(log(fraction) ~ type + psp(time) | type + psp(time), data=Ovocytes,
              family='Powerexp', xi=-0.55, maxiter=5000, local.influence=TRUE)
summary(fit)

################## Graph of the nonparametric effects ##################

par(mfrow=c(1,2))
np.graph(fit, which=1, exp=TRUE)
np.graph(fit, which=2, exp=TRUE)

################## Graph of deviance-type residuals ##################

plot(fit)

################## Graph of local influence measures ##################
```

```
ilm <- influence.ssym(fit)

###################################################################################
############### Textures of snacks Data - a log-Student-t model  #################
###################################################################################

data("Snacks", package="ssym")
fit <- ssym.l(log(texture) ~ type + ncs(week) | type, data=Snacks,
              family='Student', xi=15)
summary(fit)

################## Graph of the nonparametric effect ##################

np.graph(fit, which=1, exp=TRUE)

################## Graph of deviance-type residuals ##################

plot(fit)

###################################################################################
####################### gam.data - a Power-exponential model   ####################
###################################################################################

data("gam.data", package="gam")

fit <- ssym.l(y~psp(x),data=gam.data,family="Powerexp",xi=-0.5)
summary(fit)

################## Graph of the nonparametric effect ##################

np.graph(fit, which=1)

################## Graph of deviance-type residuals ##################

plot(fit)

###################################################################################
######### Personal Injury Insurance Data - a Birnbaum-Saunders-t model   ##########
###################################################################################

data("Claims", package="ssym")
fit <- ssym.l(log(total) ~ op_time | op_time, data=Claims,
              family='Sinh-t', xi=c(0.1,4))
summary(fit)

################## Plot of deviance-type residuals ##################

plot(fit)

###################################################################################
######### Body Fat Percentage Data - a Birnbaum-Saunders-t model   ##########
###################################################################################
```

```
data("ais", package="sn")
fit <- ssym.l(log(Bfat)~1, data=ais, family='Sinh-t', xi=c(4.5,4))
summary(fit)

id <- sort(ais$Bfat, index=TRUE)$ix
par(mfrow=c(1,2))
hist(ais$Bfat[id],xlim=range(ais$Bfat),ylim=c(0,0.1),prob=TRUE,breaks=15,
     col="light gray",border="dark gray",xlab="",ylab="",main="")
par(new=TRUE)
plot(ais$Bfat[id],exp(fit$lpdf[id])/ais$Bfat[id],xlim=range(ais$Bfat),
     ylim=c(0,0.1),type="l",xlab="",ylab="Density",main="Histogram")

plot(ais$Bfat[id],fit$cdfz[id],xlim=range(ais$Bfat),ylim=c(0,1),type="l",
     xlab="",ylab="",main="")
par(new=TRUE)
plot(ecdf(ais$Bfat[id]),xlim=range(ais$Bfat),ylim=c(0,1),verticals=TRUE,
     do.points=FALSE,col="dark gray",ylab="Probability.",xlab="",main="ECDF")

###################################################################################
################### Boston Housing Data - a log-Slash model   ####################
###################################################################################

#data("Boston", package="MASS")
#fit <- ssym.l(log(medv)~psp(lstat)|psp(lstat),data=Boston,family="Slash",xi=1.7)
#summary(fit)
#plot(fit)
```

`ssym.nl` *Fitting Semi-parametric Log-symmetric Regression Models*

Description

ssym.nl is used to fit a semi-parametric regression model suitable for analysis of data sets in which the response variable is continuous, strictly positive, and asymmetric. In this setup, both median and skewness of the response variable distribution are explicitly modeled, the median using a nonlinear function and the skewness using a semi-parametric function, which may be approximated by a natural cubic spline or a P-spline.

Usage

```
ssym.nl(formula, start, family, xi, data, epsilon, maxiter, subset, local.influence)
```

Arguments

`formula`	a symbolic description of the systematic component of the model to be fitted. See details for further information.
`start`	a named numeric vector of starting estimates for the parameters in the specified nonlinear function.

`family`	a description of the (log) error distribution to be used in the model. Supported families include *Normal*, *Student*, *Contnormal*, *Powerexp*, *Hyperbolic*, *Slash*, *Sinh-normal* and *Sinh-t*, which correspond to normal, Student-t, contaminated normal, power exponential, symmetric hyperbolic, slash, sinh-normal and sinh-t distributions, respectively.
`xi`	a numeric value or numeric vector that represents the extra parameter of the specified error distribution.
`data`	an optional data frame, list or environment containing the variables in the model.
`epsilon`	an optional positive value, which represents the convergence criterion. Default value is 1e-07.
`maxiter`	an optional positive integer giving the maximal number of iterations for the estimating process. Default value is 1e03.
`subset`	an optional expression specifying a subset of individuals to be used in the fitting process.
`local.influence`	logical. If TRUE, local influence measures under two perturbation schemes are calculated.

Details

The argument *formula* comprises of three parts (separated by the symbols "~" and "|"), namely: observed response variable in log-scale, predictor of the median submodel (having logarithmic link) and predictor of the skewness submodel (having logarithmic link). A non-parametric effect may be specified in the predictors, either approximated by a natural cubic spline or a P-spline using the functions *ncs()* or *psp()*, respectively.

The iterative estimation process is based on the Fisher scoring and backfitting algorithms. Because some distributions such as log-Student-t, log-contaminated-normal, log-power-exponential, log-slash and log-hyperbolic may be obtained as a power mixture of the log-normal distribution, the expectation-maximization (EM) algorithm is applied in those cases to obtain a more efficient iterative process for the parameter estimation. Furthermore, because the Birnbaum-Saunders-t distribution can be obtained as a scale mixture of the Birnbaum-Saunders distribution, the expectation-maximization algorithm is also applied in this case to obtain a more efficient iterative process for the parameter estimation. The smoothing parameter(s) is(are) chosen using the unweighted cross-validation score.

The function *ssym.nl()* calculates deviance-type residuals for both submodels as well as local influence measures under case-weight and response perturbation schemes.

Value

`coefs.mu`	a vector of parameter estimates associated with the median submodel.
`coefs.phi`	a vector of parameter estimates associated with the skewness submodel.
`vcov.mu`	approximate variance-covariance matrix associated with the median submodel.
`vcov.phi`	approximate variance-covariance matrix associated with the skewness submodel.
`weights`	final weights of the iterative process.

lambda.phi	estimate for the smoothing parameter associated with the nonparametric part of the skewness submodel.
dfe.phi	degrees of freedom associated with the nonparametric part of the skewness submodel.
deviance.mu	a vector of *deviances* associated with the median submodel.
deviance.phi	a vector of *deviances* associated with the skewness submodel.
mu.fitted	a vector of fitted values for the (log) median submodel.
phi.fitted	a vector of fitted values for the skewness submodel.
lpdf	a vector of individual contributions to the log-likelihood function.
cw	if *local.influence=TRUE*, a matrix of local influence and total local influence measures (under the case-weight perturbation scheme) associated with the median submodel.
pr	if *local.influence=TRUE*, a matrix of local influence and total local influence measures (under the response perturbation scheme) associated with the median submodel.
cw.theta	if *local.influence=TRUE*, a matrix of local influence and total local influence measures (under the case-weight perturbation scheme).
pr.theta	if *local.influence=TRUE*, a matrix of local influence and total local influence measures (under the response perturbation scheme).

Author(s)

Luis Hernando Vanegas <hvanegasp@gmail.com> and Gilberto A. Paula

References

Vanegas, L.H. and Paula, G.A. (2015a) A Semiparametric Approach for Joint Modeling of Median and Skewness. TEST (to appear)

Vanegas, L.H. and Paula, G.A. (2015b) Log-symmetric distributions: statistical properties and parameter estimation. Brazilian Journal of Probability and Statistics (to appear)

See Also

ssym.l

Examples

```
###################################################################################
######### Ultrasonic Calibration Data - a log-contaminated-normal model ###########
###################################################################################

data("Chwirut1", package="NISTnls")
sv <- c(b1=0.15, b2=0.005, b3=0.012)
fit <- ssym.nl(log(y) ~ -b1*x - log(b2 + b3*x) | psp(x), start=sv,
       data=Chwirut1, family='Contnormal', xi=c(0.6,0.1), local.influence=TRUE)
summary(fit)
```

```
################## Graph of the nonparametric effect ##################

np.graph(fit, which=2, exp=TRUE)

################## Graph of deviance-type residuals ##################

plot(fit)

################## Graph of local influence measures ##################

ilm <- influence.ssym(fit)

###################################################################################
################## Biaxial Fatigue Data - a Birnbaum-Saunders model   #############
###################################################################################

data("Biaxial", package="ssym")
sv <- c(b1=16, b2=-0.25)
fit <- ssym.nl(log(Life) ~ b1*Work^b2, start=sv, data=Biaxial,
               family='Sinh-normal', xi=1.54)
summary(fit)

################## Graph of deviance-type residuals ##################

plot(fit)

###################################################################################
################## European rabbits Data - a log-normal model   #############
###################################################################################

data("Erabbits", package="ssym")
fit <- ssym.nl(log(wlens) ~ b1 - b2/(b3 + age) | psp(age), start=c(b1=5.6,
               b2=128, b3=36.4), data=Erabbits, family='Normal')
summary(fit)

################## Graph of the nonparametric effect ##################

np.graph(fit, which=2, exp=TRUE)

################## Graph of deviance-type residuals ##################

plot(fit)

###################################################################################
######### Gross Domestic Product per capita Data - a Birnbaum-Saunders model ######
###################################################################################

data("gdp", package="ssym")
fit <- ssym.nl(log(gdp2010) ~ b1, start=c(b1=mean(log(gdp$gdp2010))), data=gdp,
               family='Sinh-normal', xi=2.2)
summary(fit)
```

```
################## Plot of the fitted model ##################

id <- sort(gdp$gdp2010, index=TRUE)$ix
par(mfrow=c(1,2))
hist(gdp$gdp2010[id],xlim=range(gdp$gdp2010),ylim=c(0,0.00025),prob=TRUE,
     breaks=200,col="light gray",border="dark gray",xlab="",ylab="",main="")
par(new=TRUE)
plot(gdp$gdp2010[id],exp(fit$lpdf[id])/gdp$gdp2010[id],xlim=range(gdp$gdp2010),
     ylim=c(0,0.00025),type="l",xlab="",ylab="Density",main="Histogram")

plot(gdp$gdp2010[id],fit$cdfz[id],xlim=range(gdp$gdp2010),ylim=c(0,1),type="l",
     xlab="",ylab="",main="")
par(new=TRUE)
plot(ecdf(gdp$gdp2010[id]),xlim=range(gdp$gdp2010),ylim=c(0,1),verticals=TRUE,
     do.points=FALSE,col="dark gray",ylab="Probability.",xlab="",main="ECDF")


###################################################################################
################# Blood flow Data - a log-power-exponential model #################
###################################################################################

#data("la", package="gamlss.nl")
#fit <- ssym.nl(log(PET60) ~ log(bflow) + log(1+b1*exp(-b2/bflow)) | bflow,
#               data=la, start=c(b1=-0.6,b2=98), family="Powerexp", xi=-0.45)
#summary(fit)
#plot(fit)


###################################################################################
############# Australian Institute of Sport Data - a log-normal model #############
###################################################################################

#data("ais", package="sn")
#sex <- ifelse(ais$sex=="male",1,0)
#ais2 <- data.frame(BMI=ais$BMI,LBM=ais$LBM,sex)
#start = c(b1=7, b2=0.3, b3=2, b4=0)
#fit <- ssym.nl(log(BMI) ~ log(b1 + b2*LBM + b3*sex + b4*LBM*sex) | sex + LBM,
#               data=ais2, start=start, family="Normal")
#summary(fit)
#plot(fit)
```

---

summary.ssym *summary.ssym*

Description

summary.ssym displays the summary of the fitted model including parameter estimates, associated (approximated) standard errors and goodness-of-fit statistics from an object of class “"ssym".

`vcov.ssym` *vcov.ssym*

Description

vcov.ssym extracts the approximate variance-covariance matrix associated to the parameter estimates from an object of class “"ssym".

Index