Package 'VarReg'

May 7, 2026

Type Package

Title Semi-Parametric Variance Regression

Version 2.0

Maintainer Kristy Robledo <robledo.kristy@gmail.com>

Description Methods for fitting semi-parametric mean and variance models, with normal or censored data. Extended to allow a regression in the location, scale and shape parameters, and further for multiple regression in each.

License GPL-3

Depends R (>= 2.10)

LazyData TRUE

Suggests testthat

Imports splines, stats, graphics, sn, survival, utils

RoxygenNote 7.2.3

Encoding UTF-8

NeedsCompilation no

Author Kristy Robledo [aut, cre],
Ian Marschner [ths]

Repository CRAN

Date/Publication 2023-05-15 23:50:02 UTC

Contents

`censlinVarReg` *Censored Linear mean and variance regression*

Description

`censlinVarReg` performs censored multivariate mean and multivariate variance regression. This function is designed to be used by the `semiVarReg` function.

Usage

```
censlinVarReg(
  dat,
  mean.ind = c(2),
  var.ind = c(2),
  cens.ind = c(3),
  mean.intercept = TRUE,
  para.space = c("all", "positive", "negative"),
  mean.init = NULL,
  var.init = NULL,
  control = list(...),
  ...
)
```

Arguments

`dat`	Dataframe containing outcome and covariate data. Outcome data must be in the first column, with censored values set to the limits. Covariates for mean and variance model in next columns.
`mean.ind`	Vector containing the column numbers of the data in 'dat' to be fit as covariates in the mean model. 0 indicates constant mean option. NULL indicates zero mean option.
`var.ind`	Vector containing the column numbers of the data in 'dat' to be fit as covariates in the variance model. FALSE indicates constant variance option.

cens.ind	Vector containing the column number of the data in 'dat' to indicate the censored data. 0 indicates no censoring, -1 indicates left (lower) censoring and 1 indicates right (upper) censoring.
mean.intercept	Logical to indicate if an intercept is to be included in the mean model. Default is TRUE.
para.space	Parameter space to search for variance parameter estimates. "positive" means only search positive parameter space, "negative" means search only negative parameter space and "all" means search all. Default is all.
mean.init	Vector of initial estimates to be used for the mean model.
var.init	Vector of initial estimates to be used for the variance model.
control	List of control parameters. See VarReg.control.
...	arguments to be used to form the default control argument if it is not supplied directly

Value

censlinVarReg returns a list of output including:

- converged: Logical argument indicating if convergence occurred.
- iterations: Total iterations performed of the EM algorithm.
- reldiff: the positive convergence tolerance that occured at the final iteration.
- loglik: Numeric variable of the maximised log-likelihood.
- boundary: Logical argument indicating if estimates are on the boundary.
- aic.c: Akaike information criterion corrected for small samples
- aic: Akaike information criterion
- bic: Bayesian information criterion
- hqc: Hannan-Quinn information criterion
- mean.ind: Vector of integer(s) indicating the column number(s) in the dataframe data that were fit in the mean model.
- mean: Vector of the maximum likelihood estimates of the mean parameters.
- var.ind: Vector of integer(s) indicating the column(s) in the dataframe data that were fit in the variance model.
- variance: Vector of the maximum likelihood estimates of the variance parameters.
- cens.ind: Integer indicating the column in the dataframe data that corresponds to the censoring indicator.
- data: Dataframe containing the variables included in the model.

censloop_em *The Censored data EM loop*

Description

censloop_em is an EM loop function for censored data to be utilised by various other higher level functions.

Usage

```
censloop_em(
  meanmodel,
  theta.old,
  beta.old,
  p.old,
  x.0,
  X,
  censor.ind,
  mean.intercept,
  maxit,
  eps
)
```

Arguments

meanmodel	Dataframe containing only the covariates to be fit in the mean model. NULL for zero mean model and FALSE for constant mean model.
theta.old	Vector containing the initial variance parameter estimates to be fit in the variance model.
beta.old	Vector containing the initial mean parameter estimates to be fit in the mean model.
p.old	Vector of length n containing the initial variance estimate.
x.0	Matrix of covariates (length n) to be fit in the variance model. All have been rescaled so zero is the minimum. If NULL, then its a constant variance model.
X	Vector of length n of the outcome variable.
censor.ind	Vector of length n of the censoring indicator. 0=uncensored, -1=left censored and 1 is right censored.
mean.intercept	Logical to indicate if mean intercept is to be included in the model.
maxit	Number of maximum iterations for the EM algorithm.
eps	Very small number for the convergence criteria.

Value

A list of the results from the EM algorithm, including:

- conv: Logical argument indicating if convergence occurred
- it: Total iterations performed of the EM algorithm
- reldiff: the positive convergence tolerance that occured at the final iteration.
- theta.new: Vector of variance parameter estimates. Note that these are not yet transformed back to the appropriate scale
- mean: Vector of mean parameter estimates
- fittedmean: Vector of fitted mean estimates
- p.old: Vector of fitted variance estimates

---

criterion *Calculation of information criterion*

Description

criterion calculates various information criterion for the algorithms in this package

Usage

```
criterion(n, loglik, param)
```

Arguments

n	Number of observations
loglik	Loglikelihood from model
param	Number of parameters fit in model

Value

A list of the four IC

- aic.c: Akaike information criterion corrected for small samples
- aic: Akaike information criterion
- bic: Bayesian information criterion
- hqc: Hannan-Quinn information criterion

linVarReg *Linear mean and variance regression function*

Description

linVarReg performs multivariate mean and multivariate variance regression. This function is designed to be used by the semiVarReg function.

Usage

```
linVarReg(
  dat,
  var.ind = c(2),
  mean.ind = c(2),
  para.space = c("all", "positive", "negative"),
  control = list(...),
  ...
)
```

Arguments

dat	Dataframe containing outcome and covariate data. Outcome data must be in the first column. Covariates for mean and variance model in next columns.
var.ind	Vector containing the column numbers of the data in 'dat' to be fit as covariates in the variance model. FALSE indicates constant variance option.
mean.ind	Vector containing the column numbers of the data in 'dat' to be fit as covariates in the mean model. 0 indicates constant mean option. NULL indicates zero mean option.
para.space	Parameter space to search for variance parameter estimates. "positive" means only search positive parameter space, "negative" means search only negative parameter space and "all" means search all.
control	List of control parameters. See VarReg.control.
...	arguments to be used to form the default control argument if it is not supplied directly

Value

linVarReg returns a list of output including:

- converged: Logical argument indicating if convergence occurred.
- iterations: Total iterations performed of the EM algorithm.
- reldiff: the positive convergence tolerance that occured at the final iteration.
- loglik: Numeric variable of the maximised log-likelihood.
- boundary: Logical argument indicating if estimates are on the boundary.

- aic.c: Akaike information criterion corrected for small samples
- aic: Akaike information criterion
- bic: Bayesian information criterion
- hqc: Hannan-Quinn information criterion
- mean.ind: Vector of integer(s) indicating the column number(s) in the dataframe data that were fit in the mean model.
- mean: Vector of the maximum likelihood estimates of the mean parameters.
- var.ind: Vector of integer(s) indicating the column(s) in the dataframe data that were fit in the variance model.
- variance: Vector of the maximum likelihood estimates of the variance parameters.
- cens.ind: Integer indicating the column in the dataframe data that corresponds to the censoring indicator. Always NULL.
- data: Dataframe containing the variables included in the model.

---

loop_em *The EM loop for the main mean and variance function*

Description

loop_em is a basic EM loop function to be utilised by various other higher level functions.

Usage

```
loop_em(meanmodel, theta.old, p.old, x.0, X, maxit, eps)
```

Arguments

meanmodel	Dataframe containing only the covariates to be fit in the mean model. NULL for zero mean model and FALSE for constant mean model.
theta.old	Vector containing the initial variance parameter estimates to be fit in the variance model.
p.old	Vector of length n containing the containing the initial variance estimate.
x.0	Matrix of covariates (length n) to be fit in the variance model. All have been rescaled so zero is the minimum. If NULL, then its a constant variance model.
X	Vector of length n of the outcome variable.
maxit	Number of maximum iterations for the EM algorithm.
eps	Very small number for the convergence criteria.

Value

A list of the results from the EM algorithm, including

- `conv`: Logical argument indicating if convergence occurred
- `it`: Total iterations performed of the EM algorithm
- `reldiff`: the positive convergence tolerance that occured at the final iteration.
- `theta.new`: Vector of variance parameter estimates. Note that these are not yet transformed back to the appropriate scale
- `mean`: Vector of mean parameter estimates
- `fittedmean`: Vector of fitted mean estimates
- `p.old`: Vector of fitted variance estimates

---

loop_lss *The EM loop for the LSS model*

Description

`loop_lss` is the EM loop function for the LSS model to be utilised by various other higher level functions

Usage

```
loop_lss(
  alldat,
  xiold,
  omega2old,
  nuold,
  mean.ind,
  var.ind,
  nu.ind,
  para.space,
  maxit,
  eps,
  int.maxit,
  print.it
)
```

Arguments

`alldat`	Dataframe containing all the data for the models. Outcome in the first column.
`xiold`	Vector of initial location parameter estimates to be fit in the location model.
`omega2old`	Vector of initial scale2 parameter estimates to be fit in the scale2 model.
`nuold`	Vector of initial nu parameter estimates to be fit in the nu model.

mean.ind	Vector containing the column numbers of the data in 'alldat' to be fit as covariates in the location model.
var.ind	Vector containing the column numbers of the data in 'alldat' to be fit as covariates in the scale2 model. FALSE indicates a constant variance model.
nu.ind	Vector containing the column numbers of the data in 'alldat' to be fit as covariates in the nu model. NULL indicates constant model.
para.space	Parameter space to search for variance parameter estimates. "positive" means only search positive parameter space, "negative" means search only negative parameter space and "all" means search all.
maxit	Number of maximum iterations for the main EM algorithm.
eps	Very small number for the convergence criteria.
int.maxit	Number of maximum iterations for the internal EM algorithm for the location and scale.
print.it	Logical to indicate if the estimates for each iteration should be printed.

Value

A list of the results from the algorithm, including conv, reldiff, it, mean, xi.new, omega2.new, nu.new, fitted.xi

- conv: Logical argument indicating if convergence occurred
- it: Total iterations performed of the EM algorithm
- reldiff: the positive convergence tolerance that occured at the final iteration
- xinew: Vector of location parameter estimates
- omega2new: Vector of scale squared parameter estimates
- nunew: Vector of shape parameter estimates
- fitted.xi: Vector of fitted location estimates

---

lssVarReg *Semi parametric location, shape and scale regression*

Description

lssVarReg performs a semiparametric location (ξ or xi), shape (ν or nu) and scale (ω or omega) regression model. Currently, this is only designed for a single covariate that is fit in the location, scale and shape models.

Usage

```
lssVarReg(
  y,
  x,
  locationmodel = c("constant", "linear", "semi"),
  scale2model = c("constant", "linear", "semi"),
  shapemodel = c("constant", "linear"),
  knots.l = 2,
  knots.sc = 2,
  knots.sh = 2,
  degree = 2,
  mono.scale = c("none", "inc", "dec"),
  para.space = c("all", "positive", "negative"),
  location.init = NULL,
  scale2.init = NULL,
  shape.init = NULL,
  int.maxit = 1000,
  print.it = FALSE,
  control = list(...),
  ...
)
```

Arguments

y	Vector containing outcome data. Must be no missing data.
x	Vector containing the covariate data, same length as y. Must be no missing data.
locationmodel	Text to specify the location model to be fit. Options: "constant" = constant model (intercept only), "linear" = linear term with x covariate, "semi" = semi-parametric spline (specify with knots.l).
scale2model	Text to specify the scale^2 model to be fit. Options: "constant" = constant term only, "linear" = linear term with x covariate, "semi" = semiparametric spline (specify with knots.sc)
shapemodel	Text to specify the shape model to be fit. Options: "constant" = constant shape model, "linear" = linear term with x covariate, "semi" = semiparametric spline (specify with knots.sh).
knots.l	Integer indicating the number of internal knots to be fit in the location model. Default is '2'. (Note that the knots are placed equidistantly over x.)
knots.sc	Integer indicating the number of internal knots to be fit in the scale^2 model. Default is '2'. (Note that the knots are placed equidistantly over x.)
knots.sh	Integer indicating the number of internal knots to be fit in the shape model. Default is '2'. (Note that the knots are placed equidistantly over x.)
degree	Integer to indicate the degree of the splines fit in the location and scale. Default is '2'.
mono.scale	Text to indicate whether the scale2 model is monotonic. Default is "none" (no monotonic constraints). Options are "inc" for increasing or "dec" for decreasing. If this is chosen, the appropriate para.space is set autopmatically ("positive" for inc, "negative" for dec).

para.space	Text to indicate the parameter space to search for scale2 parameter estimates. "positive" means only search positive parameter space, "negative" means search only negative parameter space and "all" means search all parameter spaces. Default is all.
location.init	Vector of initial parameter estimates for the location model. Defaults to vector of 1's of appropriate length.
scale2.init	Vector of initial parameter estimates for the scale^2 model. Defaults to vector of 1's of appropriate length.
shape.init	Vector of initial parameter estimates for the shape model. Defaults to vector of 1's of appropriate length.
int.maxit	Integer of maximum iterations for the internal location and scale EM algorithm. Default is 1000 iterations.
print.it	Logical for printing progress of estimates through each iteration. Default is FALSE.
control	List of control parameters for the algorithm. See VarReg.control.
...	arguments to be used to form the default control argument if it is not supplied directly

Value

lssVarReg returns an object of class "lssVarReg", which inherits most from class "VarReg". This object of class lssVarReg is a list of the following components:

- modeltype: Text indicating the model that was fit, always "LSS model".
- locationmodel, scale2model, shapemodel, knots.l, knots.sc, knots.sh, degree,mono.scale : Returning the input variables as described above
- converged: Logical argument indicating if convergence occurred.
- iterations: Total iterations performed of the main algorithm (not including the internal EM algorithm).
- reldiff: the positive convergence tolerance that occured at the final iteration.
- loglik: Numeric variable of the maximised log-likelihood.
- aic.c: Akaike information criterion corrected for small samples
- aic: Akaike information criterion
- bic: Bayesian information criterion
- hqc: Hannan-Quinn information criterion
- location: Vector of the maximum likelihood estimates of the location parameters.
- scale2: Vector of the maximum likelihood estimates of the scale (squared) parameters.
- shape: Vector of the maximum likelihood estimates of the shape parameters.
- data: Dataframe containing the variables included in the model.

See Also

VarReg.control plotlssVarReg

Examples

```
## run a model with linear mean, linear variance and constant shape (not run):
## lssmodel<-lssVarReg(mcycle$accel, mcycle$times, locationmodel="linear", scale2model="linear",
## shapemodel="constant",  maxit=10000)
```

`lssVarReg.multi` *Semi parametric location, shape and scale regression*

Description

`lssVarReg.multi` performs a semiparametric location (ξ or xi), shape (ν or nu) and scale (ω or omega) regression model. This is designed for multiple covariates that are fit in the location, scale and shape models.

Usage

```
lssVarReg.multi(
  y,
  x,
  locationmodel = c("constant", "linear", "semi"),
  location.vars = c(1),
  scale2model = c("constant", "linear", "semi"),
  scale2.vars = c(1),
  shapemodel = c("constant", "linear", "semi"),
  shape.vars = c(1),
  knots.l = NULL,
  knots.sc = NULL,
  knots.sh = NULL,
  degree = 2,
  location.init = NULL,
  scale2.init = NULL,
  shape.init = NULL,
  int.maxit = 1000,
  print.it = FALSE,
  control = list(...),
  ...
)
```

Arguments

`y`	Vector containing outcome data. Must be no missing data.
`x`	Matrix containing the covariate data, same length as `y`. Must be no missing data.
`locationmodel`	Vector to specify the location model to be fit for each covariate. Options: `"constant"` = constant model (intercept only), `"linear"` = linear term with x covariate, `"semi"` = semiparametric spline (specify with `knots.l`).

location.vars	Vector to specify the column(s) in x referring to covariates to be fit in the location model, eg c(1,2) indicates columns 1 and 2 in x. Must be the same length as locationmodel which specifies if they are fit as linear/semi. If semi, use knots.l to specify knots.
scale2model	Vector to specify the scale^2 model to be fit for each covariate. Options: "constant" = constant term only, "linear" = linear term with x covariate, "semi" = semi-parametric spline (specify with knots.sc)
scale2.vars	Vector to specify the column(s) in x referring to covariates to be fit in the scale^2 model, eg c(1,2) indicates columns 1 and 2 in x. Must be the same length as scale2model which specifies if they are fit as linear/semi. If semi, use knots.sc to specify knots.
shapemodel	Vector to specify the shape model to be fit for each covariate. Options: "constant" = constant shape model, "linear" = linear term with x covariate, "semi" = semiparametric spline (specify with knots.sh).
shape.vars	Vector to specify the column(s) in x referring to covariates to be fit in the shape model, eg c(1,2) indicates columns 1 and 2 in x. Must be the same length as shapemodel which specifies if they are fit as linear/semi. If semi, use knots.sh to specify knots.
knots.l	Vector indicating the number of internal knots to be fit in the location model for each covariate. Default is '2'. (Note that the knots are placed equidistantly over x.)
knots.sc	Vector indicating the number of internal knots to be fit in the scale^2 model for each covariate. Default is '2'. (Note that the knots are placed equidistantly over x.)
knots.sh	Vector indicating the number of internal knots to be fit in the shape model for each covariate. Default is '2'. (Note that the knots are placed equidistantly over x.)
degree	Integer to indicate the degree of the splines fit in the location, scale and shape. Default is '2'.
location.init	Vector of initial parameter estimates for the location model. Defaults to vector of 1's of appropriate length.
scale2.init	Vector of initial parameter estimates for the scale^2 model. Defaults to vector of 1's of appropriate length.
shape.init	Vector of initial parameter estimates for the shape model. Defaults to vector of 1's of appropriate length.
int.maxit	Integer of maximum iterations for the internal location and scale EM algorithm. Default is 1000 iterations.
print.it	Logical for printing progress of estimates through each iteration. Default is FALSE.
control	List of control parameters for the algorithm. See VarReg.control.
...	arguments to be used to form the default control argument if it is not supplied directly

Value

`lssVarReg` returns an object of class `"lssVarReg"`, which inherits most from class `"VarReg"`. This object of class `lssVarReg` is a list of the following components:

- `modeltype`: Text indicating the model that was fit, always "LSS model" for this model.
- `locationmodel`, `scale2model`, `shapemodel`, `knots.l`, `knots.sc`, `knots.sh`, `degree`,`mono.scale` : Returning the input variables as described above
- `converged`: Logical argument indicating if convergence occurred.
- `iterations`: Total iterations performed of the main algorithm (not including the internal EM algorithm).
- `reldiff`: the positive convergence tolerance that occured at the final iteration.
- `loglik`: Numeric variable of the maximised log-likelihood.
- `aic.c`: Akaike information criterion corrected for small samples
- `aic`: Akaike information criterion
- `bic`: Bayesian information criterion
- `hqc`: Hannan-Quinn information criterion
- `location`: Vector of the maximum likelihood estimates of the location parameters.
- `scale2`: Vector of the maximum likelihood estimates of the scale (squared) parameters.
- `shape`: Vector of the maximum likelihood estimates of the shape parameters.
- `data`: Dataframe containing the variables included in the model.

See Also

`VarReg.control` `plotlssVarReg`

Examples

```
## not run
## library(palmerpenguins)
## cc<-na.omit(penguins)
## y<-cc$body_mass_g
## x<-as.data.frame(cbind(cc$bill_length_mm, cc$flipper_length_mm,cc$bill_depth_mm))
## colnames(x) <-c("bill length mm", "flipper length mm","bill depth mm")
## model1<-lssVarReg.multi(y, x,
##                         locationmodel="linear", location.vars = 2,
##                         scale2model="constant",
##                         shapemodel=c("linear", "semi"), shape.vars = c(2,3),
##                         knots.sh = 1, int.maxit=10 )
## model1[-21] ## print model
```

lss_calc *Calculations for SN*

Description

lss_calc performs calculations for transforming SN data (location, scale and shape) to mean, variance and skew. This function is utilised by other, higher level functions.

Usage

```
lss_calc(x)
```

Arguments

x	Object of class lssVarReg (output from lssVarReg).

Value

dataframe containing:

- y: y variable
- x: x variable
- eta: η or fitted location estimates
- omega: ω or fitted scale estimates
- shape: α or fitted shape estimates
- predicted mean: fitted mean estimates
- predicted variance: fitted variance estimates
- Predicted skewness: fitted skewness estimates
- stand.res2: Squared standardised residuals

mcycle *mcycle dataset.*

Description

A dataset containing 133 observations from a simulated motorcycle accident, used to test crash helmets.

Usage

```
mcycle
```

Format

A data frame with 133 rows and 2 variables:

times in milliseconds from time of impact

accel in g, acceleration of the head ...

Source

Silverman, B. W. (1985) Some aspects of the spline smoothing approach to non-parametric curve fitting. Journal of the Royal Statistical Society series B 47, 1-52.

References

Venables, W. N. and Ripley, B. D. (1999) Modern Applied Statistics with S-PLUS. Third Edition. Springer.

Examples

```
library(VarReg)
data(mcycle)
attach(mcycle)
plot(times,accel)
```

---

plotlssVarReg *Plots graphics for a location, scale and shape regression model*

Description

`plotlssVarReg` is used to produce graphics for models fit in the `VarReg` package with the `lssVarReg` function. As the skew-normal distribution is used to fit this type of model, the data needs to be transformed from the SN parameters (location, scale and shape) to the typical mean, variance and skew parameters.

Usage

```
plotlssVarReg(x, knot.lines = FALSE, xlab = "x", ylab = "y")
```

Arguments

`x`	Object of class lssVarReg (output from `lssVarReg`).
`knot.lines`	Logical to show the knot lines on the graphics (if model is type "semi"). Default is `TRUE`
`xlab`	Label to be placed on the x axis of graphics (covariate)
`ylab`	Label to be placed on the y axis of graphics (outcome)

Value

A graphic is returned, as well as a dataframe. The graphic returned is a 2 by 2 plot of:

- the mean function over the x-variable, with or without the knot lines indicated
- the variance function over the x-variable, with or without the knot lines indicated
- the skew function over the x-variable, with or without the knot lines indicated
- a Q-Q plot of the squared residuals from the model, plotted against the Chi-squared (df=1) distribution. For data from a skew-normal distribution, these residuals should follow a Chi-squared (df=1) distribution, regardless of skew.

The dataframe returned contains the following columns:

- `x`: x variable
- `y`: y variable
- `eta`: (η), the location parameter
- `omega`: (ω), the scale parameter
- `shape`: (ν), the shape parameter
- `predicted~mean`: (μ), the mean
- `predicted~variance`: (σ^2), the variance
- `predicted~skewness`: (γ), the skew
- `stand.res2`: the standardised residuals squared.

See Also

`lssVarReg`

Examples

```
data(mcycle)
## not run. LSS model followed by the basic plot command
##lssmodel<-lssVarReg(mcycle$accel, mcycle$times, locationmodel="linear", scale2model="linear",
##shapemodel="constant", maxit=10000)
##lssplot_out<-plotlssVarReg(lssmodel, xlab="Time in seconds", ylab="Acceleration")
```

`plotVarReg` *Plots graphics for a mean and variance regression model*

Description

`plotVarReg` to produce graphics for models fit in this package.

Usage

```
plotVarReg(
  x,
  knot.lines = FALSE,
  ci = FALSE,
  ci.type = c("im", "boot"),
  bootreps = 1000,
  xlab = "x",
  ylab = "y",
  control = list(...),
  ...
)
```

Arguments

x	Object of class VarReg (see semiVarReg).
knot.lines	Logical to indicate if knot lines should be shown on graphics (if model is type "semi"). Default is FALSE
ci	Logical indicate if 95% CI should be shown on the plots. Default is FALSE and ci.type="im".
ci.type	Text to indicate the type of CI to plot. Either "im" (information matrix) or "boot" (bootstrapped). Default is "im".
bootreps	Integer to indicate the number of bootstrap replications to be performed if ci.type="boot". Default is 1000.
xlab	Text for the label to be placed on the x axis of graphics (covariate)
ylab	Text for the label to be placed on the y axis of graphics (outcome)
control	list of control parameters to be used in bootstrapping. See VarReg.control.
...	arguments to be used to form the default control argument if it is not supplied directly

Value

This function returns a 2x2 plot, with slightly different plots given, depending on the outcome data. For uncensored data, the plots are:

- the mean function over the x-variable, with or without 95% CI, and with or without the knot lines indicated
- the variance function over the x-variable, with or without 95% CI and with or without the knot lines indicated
- a Q-Q plot of the residuals from the model
- a histogram of the residuals from the model

If the outcome data is censored, the last two plots are no longer appropriate. Given the censored residuals from the model, we can compare the squared standardised residuals (given in black) with their censoring indicator to the chi-squared distribution with one degree of freedom (given in red). This is one of the plots given for censored data, and the other is a plot of the data, coloured by the censoring status. The triangles with the point at the top are bottom censored and the triangles with the point at the bottom are top censored.

See Also

semiVarReg, VarReg.control

Examples

```
data(mcycle)
linmodel<-semiVarReg(mcycle$accel, mcycle$times, meanmodel="linear", varmodel="linear",
maxit=10000)
plotVarReg(linmodel)
plotVarReg(linmodel, ci=TRUE, ci.type="im", ylab="Range", xlab="Time in seconds")
##not run
##plotVarReg(linmodel, ci=TRUE, ci.type="boot", bootreps=10,ylab="Acceleration",
##xlab="Time in seconds")

##not run
##semimodel<-semiVarReg(mcycle$accel, mcycle$times, meanmodel="semi", varmodel="semi",
##knots.m=4, knots.v=2, maxit=10000)
##plotVarReg(semimodel, ci=TRUE, ci.type="boot",bootreps=10,ylab="Acceleration",
##xlab="Time in seconds", maxit=10000)
```

---

searchVarReg *Searches for best semi parametric mean and variance regression model*

Description

searchVarReg performs multiple semi-parametric mean and variance regression models for a covariate of interest, in order to search for the optimal number of knots. The best model is chosen based on the information criterion of preference ("selection"). At the moment, this is only designed for a single covariate that is fit in both the mean and variance models.

Usage

```
searchVarReg(
  y,
  x,
  cens.ind = NULL,
  maxknots.m = 3,
  maxknots.v = 3,
  degree = 2,
  mono.var = c("none", "inc", "dec"),
  selection = c("AIC", "AICc", "HQC", "BIC"),
  print.it = FALSE,
  control = list(...),
  ...
)
```

Arguments

y	Vector containing outcome data. Must be no missing data and any censored values must be set to the limits of detection.
x	Vector containing the covariate data. Must be no missing data and same length as y.
cens.ind	Vector containing the censoring indicator, if applicable. There must be no missing data contained in the vector and this vector should be the same length as y. "0" values indicate uncensored data, "1" indicates right, or upper, censoring and "-1" indicates left, or lower, censoring. The default is NULL which indicates there is no censored data.
maxknots.m	Integer indicating the maximum number of internal knots to be fit in the mean model. Default is 3. (Note that the knots are placed equidistantly over x.)
maxknots.v	Integer indicating the maximum number of internal knots to be fit in the variance model. Default is 3. (Note that the knots are placed equidistantly over x.)
degree	The degree of the splines fit in the mean and variance. Default is 2.
mono.var	Text to indicate whether the variance model is monotonic (only applied to 'linear' or semi-parametric variance models). Default is "none" (no monotonic constraints). Options are "inc" for increasing or "dec" for decreasing. If the variance model is linear, the parameter space is constrained (positive for increasing and negative for decreasing). For semi-parametric variance models, the appropriate monotonic B splines are fit in the semi-parametric variance model.
selection	Text to indicate which information criteria is to be used for the selection of the best model. Choices are "AIC", "AICc", "BIC" and "HQC". Default is "AIC".
print.it	Logical to indicate whether to print progress from each model as the models are performed. Default is FALSE.
control	list of control parameters. See VarReg.control.
...	arguments to be used to form the default control argument if it is not supplied directly

Details

A matrix of models are performed, of increasing complexity. Mean models start at a zero mean model, then constant mean, linear, 0 internal knots, etc, up to a maximum internal knots as specified in maxknots.m. Variance models start at constant variance, linear variance, 0 internal knots, etc, up to max internal knots as specified in maxknots.v.

Note that this function can take some time to run, due to the number of models to be fit. A window will appear on windows based systems to show a progress bar for the function.

Value

searchVarReg returns an list, with the following components:

- ll: a dataframe of the log-likelihoods from each of the models that have been fit.
- AIC: a dataframe of the AIC from each of the models that have been fit. The parameters fit in the mean model are given in the columns, and the parameters in the variance are given in the rows.

- AICc: a dataframe of the AIC-c from each of the models that have been fit.
- BIC: a dataframe of the BIC from each of the models that have been fit.
- HQC: a dataframe of the HQC from each of the models that have been fit.
- best.model: an object of class VarReg (see semiVarReg) containing the output from the optimal model (that model within the specified models in the mean and variance with the lowest information criterion according to the criterion selected).

See Also

semiVarReg, VarReg.control

Examples

```
data(mcycle)
### not run
### find<-searchVarReg(mcycle$accel, mcycle$times, maxknots.v=3, maxknots.m=3,
### selection="HQC", maxit=10000)
```

---

semiVarReg *Semi parametric mean and variance regression*

Description

semiVarReg performs semi-parametric mean and variance regression models. Currently, this is only designed for a single covariate that is fit in the mean and variance models.

Usage

```
semiVarReg(
  y,
  x,
  cens.ind = NULL,
  meanmodel = c("zero", "constant", "linear", "semi"),
  mean.intercept = TRUE,
  varmodel = c("constant", "linear", "semi"),
  knots.m = 2,
  knots.v = 2,
  degree = 2,
  mono.var = c("none", "inc", "dec"),
  para.space = c("all", "positive", "negative"),
  control = list(...),
  ...
)
```

Arguments

y	Vector containing outcome data. Must be no missing data and any censored values must be set to the limits of detection.
x	Vector containing the covariate data. Must be no missing data and same length as y.
cens.ind	Vector containing the censoring indicator, if applicable. There must be no missing data contained in the vector and this vector should be the same length as y. "0" values indicate uncensored data, "1" indicates right, or upper, censoring and "-1" indicates left, or lower, censoring. The default is NULL which indicates there is no censored data.
meanmodel	Text to specify the mean model to be fit to the data. The possible inputs are "zero", "constant", "linear" or "semi". "semi" indicates a semi-parametric spline model, with the number of internal knots specified in knots.m.
mean.intercept	Logical argument to indicate if the mean model is to include an intercept term. This option is only available in the censored mean model, and the default=TRUE.
varmodel	Text to specify the variance model to be fit to the data. The possible inputs are "constant", "linear" or "semi". "semi" indicates a semi-parametric B-spline model, with the number of internal knots specified in knots.v.
knots.m	Integer indicating the number of internal knots to be fit in the semi-parametric mean model. Knots are placed equidistantly over the covariate. The default value is 2.
knots.v	Integer indicating the number of internal knots to be fit in the semi-parametric variance model. Knots are placed equidistantly over the covariate. The default value is 2.
degree	Integer indicating the degree of the splines fit in the mean and the variance models. The default value is 2.
mono.var	Text to indicate whether the variance model is monotonic. Note that this is not available for the "constant" variance model. Options are "none", "inc" or "dec", with the default="none". "Inc" indicates increasing monotonic and "dec" indicates decreasing monotonic. If the variance model is linear, the parameter space is constrained (positive for increasing and negative for decreasing). For semi-parametric variance models, the appropriate monotonic B-splines are fit in the semi-parametric variance model.
para.space	Text to indicate the parameter space to search for scale2 parameter estimates. "positive" means only search positive parameter space, "negative" means search only negative parameter space and "all" means search all parameter spaces. Default is all.
control	list of control parameters. See VarReg.control.
...	arguments to be used to form the default control argument if it is not supplied directly

Value

semiVarReg returns an object of class "VarReg" which inherits some components from the class "glm". This object of class "VarReg" is a list containing the following components:

- modeltype: Text indicating the model that was fit, indicating if a censored approach or an uncensored approach was performed.
- knots.m, knots.v, degree, meanmodel, varmodel: Returning the input variables as described above
- converged: Logical argument indicating if convergence occurred.
- iterations: Total iterations performed.
- reldiff: the positive convergence tolerance that occurred at the final iteration.
- loglik: Numeric variable of the maximised log-likelihood.
- boundary: Logical argument indicating if the MLE is on the boundary of the parameter space.
- aic.c: Akaike information criterion corrected for small samples
- aic: Akaike information criterion
- bic: Bayesian information criterion
- hqc: Hannan-Quinn information criterion
- mean.ind: Vector of integer(s) indicating the column number(s) in the dataframe data that were fit in the mean model.
- mean: Vector of the maximum likelihood estimates of the mean parameters.
- var.ind: Vector of integer(s) indicating the column(s) in the dataframe data that were fit in the variance model.
- variance: Vector of the maximum likelihood estimates of the variance parameters.
- cens.ind: Integer indicating the column in the dataframe data that corresponds to the censoring indicator.
- data: Dataframe containing the variables included in the model.

Examples

```
data(mcycle)
## run a model with linear mean and linear variance:
linmodel<-semiVarReg(mcycle$accel, mcycle$times, meanmodel="linear", varmodel="linear",
 maxit=10000)
## run a model with semi-parametric mean (4 internal knots) and semi-parametric variance (2 knots):
##not run
##semimodel<-semiVarReg(mcycle$accel, mcycle$times, meanmodel="semi", varmodel="semi",
##knots.m=4, knots.v=2, maxit=10000)
## run a model with semi-parametric mean (4 internal knots) and semi-parametric monotonic
## variance (2 knots):
## not run
##semimodel_inc<-semiVarReg(mcycle$accel, mcycle$times, meanmodel="semi", varmodel="semi",
##knots.m=4, knots.v=2, mono.var="inc")
```

`semiVarReg.multi` *Semi parametric mean and variance regression (multivariate)*

Description

`semiVarReg.multi` performs semi-parametric mean and variance regression models. This is designed for multiple covariates fit in the mean and variance models.

Usage

```
semiVarReg.multi(
  y,
  x,
  mean.model = c("zero", "constant", "linear", "semi"),
  mean.vars = c(1),
  knots.m = NULL,
  var.model = c("constant", "linear", "semi"),
  var.vars = c(1),
  knots.v = NULL,
  degree = 2,
  control = list(...),
  ...
)
```

Arguments

`y`	Vector containing outcome data. Must be no missing data and any censored values must be set to the limits of detection.
`x`	Matrix containing the covariate data. Must be no missing data and same length as `y`.
`mean.model`	Vector to specify the mean model to be fit to the data. The possible inputs are `"zero"`, `"constant"`, or a vector to indicate if covariates are to be `"linear"` or `"semi"`. `"semi"` indicates a semi-parametric spline model, with the number of internal knots specified in `knots.m`. If covariates are fit, each covariate needs an indicator of `"linear"` or `"semi"`, where `mean.vars` specifies each covariate.
`mean.vars`	Vector to specify column(s) in `x` referring to covariates to be fit in the mean model, eg c(1,2) indicates columns 1 and 2 in `x`. Must be the same length as `mean.model` which specifies if they are fit as linear/semi. If semi, use `knots.m` to specify knots.
`knots.m`	Vector indicating the number of internal knots to be fit in each of covariate(s) fit in the semi-parametric mean model. Must be one entry per `"semi"` covariate in `mean.model`. Knots are placed equidistantly over each covariate.
`var.model`	Vector to specify the variance model to be fit to the data. The possible inputs are `"constant"`, or a vector to indicate if each covariate is to be `"linear"` or `"semi"`. `"semi"` indicates a semi-parametric B-spline model, with the number of internal knots specified in `knots.v`.

var.vars	Vector to specify column(s) in x referring to covariates to be fit in the variance model, eg c(1,2) indicates columns 1 and 2 in x. Must be the same length as var.model which specifies if they are fit as linear/semi. If semi, use knots.v to specify knots.
knots.v	Vector indicating the number of internal knots to be fit in the semi-parametric variance model. Knots are placed equidistantly over the covariate.
degree	Integer indicating the degree of the splines fit in the mean and the variance models. The default value is 2.
control	list of control parameters. See VarReg.control.
...	arguments to be used to form the default control argument if it is not supplied directly

Value

semiVarReg.multi returns an object of class "VarReg" which inherits some components from the class "glm". This object of class "VarReg" is a list containing the following components:

- modeltype: Text indicating the model that was fit, indicating an uncensored approach was performed.
- knots.m, knots.v, degree, meanmodel, varmodel: Returning the input variables as described above
- converged: Logical argument indicating if convergence occurred.
- iterations: Total iterations performed.
- reldiff: the positive convergence tolerance that occurred at the final iteration.
- loglik: Numeric variable of the maximised log-likelihood.
- boundary: Logical argument indicating if the MLE is on the boundary of the parameter space.
- aic.c: Akaike information criterion corrected for small samples
- aic: Akaike information criterion
- bic: Bayesian information criterion
- hqc: Hannan-Quinn information criterion
- mean.ind: Vector of integer(s) indicating the column number(s) in the dataframe data that were fit in the mean model.
- mean: Vector of the maximum likelihood estimates of the mean parameters.
- var.ind: Vector of integer(s) indicating the column(s) in the dataframe data that were fit in the variance model.
- variance: Vector of the maximum likelihood estimates of the variance parameters.
- data: Dataframe containing the variables included in the model.

Examples

```
data(mcycle)
## run a model with linear mean and linear variance:
linmodel<-semiVarReg.multi(mcycle$accel, x=mcycle, mean.model="linear",mean.vars=2,
var.model="linear", var.vars=2,  maxit=10000)
```

```
## run a model with semi-parametric mean (4 internal knots) and semi-parametric variance (2 knots):
##not run
##semimodel<-semiVarReg.multi(mcycle$accel, x=mcycle, meanmodel="semi",mean.vars=2, varmodel="semi",
##var.vars=2,knots.m=4, knots.v=2, maxit=10000)
```

---

seVarReg *SE calculations for mean and variance regression models*

Description

seVarReg calculates SE for an object of class VarReg. If the result is not on a boundary, the Fishers Information matrix SE are given. The bootstrapped 95% CI can also be calculated. Designed to be called by the plot function plotVarReg, rather than run by a user.

Usage

```
seVarReg(
  x,
  boot = FALSE,
  bootreps = 1000,
  vector.mean = x$data[, 2],
  vector.variance = x$data[, 2],
  control = list(...),
  ...
)
```

Arguments

x	Object of class VarReg to determin the SE (eg. result from semiVarReg).
boot	Logical to indicate if bootstrapped CI should be calculated. Default is FALSE.
bootreps	Number of bootstraps to be performed if boot=TRUE. Default is 1000.
vector.mean	Vector of x values for which the SE of the mean is to be calculated. Default is the x covariate from the model.
vector.variance	Vector of x values for which the SE of the variance is to be calculated. Default is the actual x covariate from the model.
control	List of control parameters for the bootstrapped models. See VarReg.control.
...	arguments to be used to form the default control argument if it is not supplied directly

Value

The result is a list of results. This includes:

- mean.est: dataframe of overall results from the mean model, including parameter estimates from the model, SEs from information matrix (if boundary=FALSE) and if specified, the SE from bootstrapping with the bootstrapped 95% CI.

- variance.est: dataframe of overall results from the variance model, including parameter estimates from the model, SEs from information matrix (if boundary=FALSE) and if specified, the SE from bootstrapping with the bootstrapped 95% CI.
- mean.im: dataframe of the expected information matrices for the mean (as appropriate)
- variance.im: dataframe of the expected information matrices for the variance (as appropriate)
- mean.outputs: dataframe with complete output for mean graphics. Includes the vector.mean as input, and the mean vector (mean.mean) and the SE vector mean.se.im, and bootstrapping outputs as appropriate.
- variance.outputs: dataframe with complete output for variance graphics. Includes the vector.variance as input, and the mean vector (var.mean) and the SE vector var.se.im, and bootstrapping outputs as appropriate.

See Also

semiVarReg, VarReg.control

Examples

```
data(mcycle)
##Fit model with range as a covariate in the mean and the variance model
semimodel<-semiVarReg(mcycle$accel, mcycle$times, meanmodel="semi", varmodel="linear",
knots.m=4, maxit=10000)
##Calculate SE
se1<-seVarReg(semimodel, boot=FALSE)
##not run: with bootstrapping
##se2<-seVarReg(semimodel, boot=TRUE, bootreps=10)
##not run: calculate mean and SE for a given sequence
##test.seq<-seq(min(mcycle$times), max(mcycle$times),
##by=((max(mcycle$times)-min(mcycle$times))/999))
##se2<-seVarReg(semimodel, boot=TRUE, bootreps=10, vector.mean=test.seq)
```

VarReg *VarReg: Semi-parametric mean and variance regression*

Description

Methods for fitting semi-parametric mean and variance models, with normal or censored data. Also extended to allow a regression in the location, scale and shape parameters.

Details

This package provides functions to fit semi-parametric mean and variance regression models. These models are based upon EM-type algorithms, which can have more stable convergence properties than other algorithms for additive variance regression models.

The primary function to use for linear and semi-parametric mean and variance models is semiVarReg. This function also is able to fit models to censored outcome data. There is also a plot function for

these models called plotVarReg. A search function has also been produced in order to assist users to find the optimal number of knots in the model (searchVarReg).

The other functions that are of particular use are lssVarReg and its plot function plotlssVarReg. This uses the skew-normal distribution and combines the EM algorithm with a coordinate-ascent type algorithm in order to fit a regression model in the location, scale and shape, therefore extending the semi-parametric models to non-normal data.

Multivariate models can be fit with semiVarReg.multi and lssVarReg.multi

Author(s)

Kristy Robledo <robledo.kristy@gmail.com>

`VarReg.control` *Auxillary for controlling VarReg fitting*

Description

Use `VarReg.control` to determine parameters for the fitting of semiVarReg. Typically only used internally within functions.

Usage

```
VarReg.control(bound.tol = 1e-05, epsilon = 1e-06, maxit = 1000)
```

Arguments

`bound.tol`	Positive tolerance for specifying the interior of the parameter space. This allows the algorithm to terminate early if an interior maximum is found. If set to `bound.tol=Inf`, no early termination is attempted.
`epsilon`	Positive convergence tolerance. If θ is a vector of estimates, convergence is declared when $\sqrt{(\sum(\theta_{old}-\theta_{new})^2)}/\sqrt{\sum(\theta_{old})^2}$. This should be smaller than `bound.tol`.
`maxit`	integer giving the maximum number of EM algorithm iterations for a given parameterisation.

Details

This is used similarly to glm.control. If required, it may be internally passed to another function.

Value

A list of the three components: `bound.tol`, `epsilon` and `maxit` .

vcf *vcf dataset.*

Description

A dataset containing 100 observations of mean velocity of circumferential fibre shortening (vcf), made by long axis and short axis echocardiography.

Usage

```
vcf
```

Format

A data frame with 133 rows and 3 variables:

pid patient identifier

vcflong vcf measurement from long axis

vcfshort vcf measurement from short axis ...

Source

Data from Bland JM, Altman DG. (1986) Statistical methods for assessing agreement between two methods of clinical measurement. Lancet i, 307-310. (Supplied by Paul D'Arbela)

Examples

```
library(VarReg)
data(vcf)
attach(vcf)
plot(rowMeans(vcf[-1]),vcf$vcflong-vcf$vcfshort)
```

Index