Package: dirttee (via r-universe)

September 17, 2024

Type Package

Title Distributional Regression for Time to Event Data

Version 1.0.2

Date 2023-12-21

Author Alexander Seipp [aut, cre], Fabian Otto-Sobotka [aut]

Maintainer Alexander Seipp <alexander.seipp@uni-oldenburg.de>

Description Semiparametric distributional regression methods (expectile, quantile and mode regression) for time-to-event variables with right-censoring; uses inverse probability of censoring weights or accelerated failure time models with auxiliary likelihoods. Expectile regression using inverse probability of censoring weights has been introduced in Seipp et al. (2021) ``Weighted Expectile Regression for Right-Censored Data'' <doi:10.1002/sim.9137>, mode regression for time-to-event variables has been introduced in Seipp et al. (2022) ``Flexible Semiparametric Mode Regression for Time-to-Event Data'' <doi:10.1177/09622802221122406>.

License GPL (>= 2)

Depends expectreg(>= 0.5.0), R (>= 3.6.0)

Imports formula.tools, MASS, Matrix, mgcv, nloptr, provenance, rlang, splines, survival

Encoding UTF-8

RoxygenNote 7.2.3

NeedsCompilation no

Repository CRAN

Date/Publication 2023-12-21 21:20:02 UTC

Contents

`dirttee-package` *DIstributional Regression for Times To EvEnt*

Description

This package includes regession methods for right-censored response variables. It allows for the estimation of distributional regression methods with semiparametric predictors, including, for example, nonlinear, spatial or random effects. The distribution of the response can be estimated with expectiles, quantiles and mode regression. Censored observations can be included with accelerated failure time models or inverse probability of censoring weights.

Author(s)

Alexander Seipp, Fabian Otto-Sobotka
Carl von Ossietzky University Oldenburg
`https://uol.de/eub`

Maintainer: Alexander Seipp <alexander.seipp@uni-oldenburg.de>

Special thanks for their help go to Lisa Eilers and Florian Berger!

Partially funded by the German Research Foundation (DFG) grant SO1313/1-1, project 'Distributional Regression for Time-to-Event Data'.

References

Seipp A, Uslar V, Weyhe D, Timmer A, Otto-Sobotka F. Weighted expectile regression for right-censored data. *Statistics in Medicine*. 2021;40(25):5501–5520. doi: 10.1002/sim.9137

Seipp A, Uslar V, Weyhe D, Timmer A, Otto-Sobotka F. Flexible Semiparametric Mode Regression for Time-to-Event Data. *Statistical Methods in Medical Research*. 2022;31(12):2352-2367. doi: 10.1177/09622802221122406

Examples

```
data(colcancer)
c100 <- colcancer[1:100,]

#mode regression
reg <- modreg(Surv(logfollowup, death) ~ sex + LNE, data = c100)

#expectile regression
fit_exp <- expectreg.aft(Surv(logfollowup, death) ~ LNE, data = c100,smooth="f")
fit_expipc <- expectreg.ipc(Surv(logfollowup, death) ~ sex + LNE, data = c100)

#quantile regression
qu1 <- qureg.aft(Surv(logfollowup, death) ~ sex + LNE, data=c100, smooth="fixed")
```

`asynorm` *The asymmetric normal distribution.*

Description

Density, distribution function, quantile function and random generation for the asymmetric normal distribution with the parameters `mu`, `sigma` and `tau`.

Usage

```
dasynorm(x, mu = 0, sigma = 1, tau = 0.5)
pasynorm(q, mu = 0, sigma = 1, tau = 0.5)
qasynorm(p, mu = 0, sigma = 1, tau = 0.5)
rasynorm(n, mu = 0, sigma = 1, tau = 0.5)
```

Arguments

`q`	vector of quantiles.
`mu`	location parameter and mode of the distribution.
`sigma`	comparable to the standard deviation. Must be positive.
`tau`	asymmetry parameter.
`x`	vector of locations.
`p`	vector of probabilities.
`n`	number of observations. If $length(n) > 1$, the length is taken to be the number required.

Details

The asymmetric normal distribution has the following density
$f(x) = (2\sqrt{\tau(1-\tau)/\pi}/\sigma)/(\sqrt{1-\tau}+\sqrt{\tau})\exp(-|(\tau-(x <= \mu))| * (x-\mu)^2/\sigma^2)$ The cdf is derived by integration of the distribution function by using the `integrate` function.

Value

`dasynorm` gives the density, `pasynorm` gives the distribution function, `qasynorm` gives the quantile function, and `rasynorm` generates random deviates.

Corresponds to the normal distribution for $\tau = 0.5$.

The length of the result is determined by `n` for `rasynorm`, and is the maximum of the lengths of the numerical arguments for the other functions.

The numerical arguments other than `n` are recycled to the length of the result.

Examples

```
hist(rasynorm(1000))

qg <- qasynorm(0.1, 1, 2, 0.5)

pasynorm(qg, 1, 2, 0.5)

ax <- c(1:1000)/100-5
plot(ax,dasynorm(ax), type = 'l')
```

`boot.modreg` *Estimate confidence intervals and standard errors for the mode regression fit*

Description

Performs bootstrap on the modreg object.

Usage

```
boot.modreg(
  reg,
  nboot,
  level = 0.95,
  newdata = NULL,
  bw = c("variable", "fixed"),
  quiet = FALSE,
  terms = NULL,
  seed = NULL
)
```

Arguments

- `reg` — an object of class modreg (output of the modreg function)
- `nboot` — number of bootstrap replications
- `level` — confidence level
- `newdata` — Should be a data frame containing all the variables needed for predictions. If supplied, confidence intervals are calculated for the corresponding predictions.
- `bw` — Either `"variable"` or `"fix"`, determining if the bandwidth of the original fit should be used for the bootstrap fits (`fix`) or if the bandwith should be recalculated (`variable`).
- `quiet` — if TRUE, printing of the status is suppressed
- `terms` — character scalar. If supplied, uses this term for confidence intervals of the prediction
- `seed` — the seed to use

Details

A nonparametric residual bootstrap is performed to calculate standard errors of parameters and confidence intervals. More details can be found in Seipp et al. (2022). `newdata` can be supplied to get confidence intervals for specific predictions. `terms` can be specified to calculate confidence interval for the contribution of one covariate (useful for P-splines). `variable` bandwidth is the default, which has higher coverage than `fix`, but is computationally much more demanding. A `seed` can be supplied to guarantee a reproducible result.

Value

a list with the following elements

- `confpredict` — data frame, the confidence intervals for the predictions.
- `confparams` — data frame, the confidence intervals and standard errors for the parametric regression coefficients.
- `level` — confidence level
- `na` — scalar, stating the number of NA bootstrap repetitions.
- `seed` — scalar, the used seed.

References

Seipp, A., Uslar, V., Weyhe, D., Timmer, A., & Otto-Sobotka, F. (2022). Flexible Semiparametric Mode Regression for Time-to-Event Data. Manuscript submitted for publication.

Examples

```
data(colcancer)
colcancer80 <- colcancer[1:80, ]

# linear mode regression
```

```
regL <- modreg(Surv(logfollowup, death) ~ sex + age, data = colcancer80)

# bootstrap with a fixed bandwidth and 3 iterations, chosen to speed up the function.
# Should in practice be much more than 3 iterations.
btL <- boot.modreg(regL, 3, bw = "fixed", level = 0.9, seed = 100)

# coefficients, SE and confidence intervals
cbind(coef(regL), btL$confparams)


## confidence inverval for smooth effect / predictions

reg <- modreg(Surv(logfollowup, death) ~ sex + s(age, bs = "ps"), data = colcancer80,
              control = modreg.control(tol_opt = 10^-2, tol_opt2 = 10^-2, tol = 10^-3))
ndat <- data.frame(sex = rep(colcancer80$sex[1], 200), age = seq(50, 90, length = 200))

# iterations should in practice be much more than 2!
bt <- boot.modreg(reg, 2, bw = "fixed", newdata = ndat, terms = "s(age)", seed = 100)

pr <- predict(reg, newdata = ndat, type = "terms", terms = "s(age)")[, 1]

plot(ndat$age, pr, ylim = c(-0.75, 1.5), type = "l", xlab = "age", ylab = "s(age)")
lines(ndat$age, bt$confpredict$lower, lty = 2)
lines(ndat$age, bt$confpredict$upper, lty = 2)
```

colcancer — *Colon Cancer Dataset*

Description

A dataset describing colon cancer patients. The data is based on real data from a hospital-based cancer registry but many values are changed to ensure anonymity. Each row is a single case, while the columns represent patients' health conditions and physical parameters.

Usage

```
data("colcancer")
```

Format

A data.frame with 546 observations with colon cancer cases. The 12 columns describe different parameters of patients' conditions.

Details

The columns of the data set are:

- followup. numeric. Follow-up time since surgery in days. The time the patient was observed.
- logfollowup. numeric. The follow-up time, but logarithmic.
- death. integer. Indicates whether the patient died. If death occured it is set to 1, otherwise 0.
- sex. factor. Level: "f", "m". The sex of the patient. In this case "f" stands for female, and "m" represents male patients.
- LNE. numeric. The number of examined lymph nodes.
- LNR. numeric, ranges from 0 to 1. The number of cancerous lymph nodes divided by the total number (LNE).
- pUICC. factor. Levels: "I", "II", "III", "IV". Pathological cancer stage. The UICC staging system was used.
- CTX. factor. Levels: "0", "1". Chemotherapy (no / yes)
- ASA.score. factor. Levels: "mild", "severe". An ASA score smaller than 3 is considered a mild general illness, 3 or greater is considered a severe general illness. The ASA scoring system of patients was originally proposed by the American Society of Anesthesiologists.
- R.status factor. Level: "0", "12". Residual tumor after surgery. 0 stands for no residual tumor. 12 stands either for microscopic (R1) or macroscopic residues (R2).
- preexisting.cancer. integer. If there was a history of cancer before the colon cancer. Set to 1 if there has been a cancer in the past and to 0 if not.
- age. numeric. The age of the patient in years.

`expectreg.aft` *Expectile regression for right censored event times using an auxiliary likelihood*

Description

Estimate a set of conditional expectiles or quantiles with semiparametric predictors in accelerated failure time models. For the estimation, the asymmetric loss functions are reformulated into auxiliary likelihoods.

Usage

```
expectreg.aft(
   formula,
   data = NULL,
   smooth = c("cvgrid", "aic", "bic", "lcurve", "fixed"),
   lambda = 1,
   expectiles = NA, ci = FALSE)

qureg.aft(
```

```
formula,
data = NULL,
smooth = c( "cvgrid", "aic", "bic", "lcurve", "fixed"),
lambda = 1,
quantiles = NA,
ci = FALSE)
```

Arguments

`formula`	An R formula object consisting of the response variable, '~' and the sum of all effects that should be taken into consideration. Each semiparametric effect has to be given through the function `rb`. The response needs to be a call of `Surv`.
`data`	Optional data frame containing the variables used in the model, if the data is not explicitely given in the formula.
`smooth`	There are different smoothing algorithms that tune `lambda` to prevent overfitting. Caution, the currently implemented smoothing algorithms can take a long time. Cross validation is done with a grid search ('`cvgrid`'). The function can also use a supplied fixed penalty ('`fixed`'). The numerical minimisation is also possible with AIC or BIC as score ('`aic`', '`bic`'). The L-curve ('`lcurve`') is a new experimental grid search by Frasso and Eilers.
`lambda`	The fixed penalty can be adjusted. Also serves as starting value for the smoothing algorithms.
`expectiles`	In default setting, the expectiles (0.01,0.02,0.05,0.1,0.2,0.5,0.8,0.9,0.95,0.98,0.99) are calculated. You may specify your own set of expectiles in a vector. The option may be set to 'density' for the calculation of a dense set of expectiles that enhances the use of `cdf.qp` and `cdf.bundle` afterwards.
`ci`	Whether a covariance matrix for confidence intervals and a `summary` is calculated.
`quantiles`	Quantiles for which the regression should be performed.

Details

For expectile regression, the LAWS loss function

$$S = \sum_{i=1}^{n} w_i(p)(y_i - \mu_i(p))^2$$

with

$w_i(p) = p1_{(y_i > \mu_i(p))} + (1-p)1_{(y_i < \mu_i(p))}$

is repackaged into the asymmetric normal distribution. Then, an accelerated failure time model is estimated. This function is based on the 'expectreg' package and uses the same functionality to include semiparametric predictors.

For quantile regression, the loss function is replaced with a likelihood from the asymmetric laplace distribution.

Value

An object of class 'expectreg', which is basically a list consisting of:

`lambda`	The final smoothing parameters for all expectiles and for all effects in a list.
`intercepts`	The intercept for each expectile.
`coefficients`	A matrix of all the coefficients, for each base element a row and for each expectile a column.
`values`	The fitted values for each observation and all expectiles, separately in a list for each effect in the model, sorted in order of ascending covariate values.
`response`	Vector of the response variable.
`covariates`	List with the values of the covariates.
`formula`	The formula object that was given to the function.
`asymmetries`	Vector of fitted expectile asymmetries as given by argument `expectiles`.
`effects`	List of characters giving the types of covariates.
`helper`	List of additional parameters like neighbourhood structure for spatial effects or ϕ for kriging.
`design`	Complete design matrix.
`bases`	Bases components of each covariate.
`fitted`	Fitted values $\hat{y}$.
`covmat`	Covariance matrix, estimated when `ci = TRUE`.
`diag.hatma`	Diagonal of the hat matrix. Used for model selection criteria.
`data`	Original data
`smooth_orig`	Unchanged original type of smoothing.

`plot`, `predict`, `resid`, `fitted`, `effects` and further convenient methods are available for class 'expectreg'.

Author(s)

Fabian Otto-Sobotka
Carl von Ossietzky University Oldenburg
https://uol.de

See Also

`expectreg.ipc`, `expectreg.ls`

Examples

```
data(colcancer)
ex <- c(0.05, 0.2, 0.5, 0.8, 0.95)
c100 <- colcancer[1:100,]
exfit <- expectreg.aft(Surv(logfollowup, death) ~ LNE, data = c100, expectiles = ex, smooth="f")
coef(exfit)
```

```
qu1 <- qureg.aft(Surv(logfollowup, death) ~ LNE + sex, data=c100, smooth="fixed")
coef(qu1)

## Not run:

# takes some time
qu2 <- qureg.aft(Surv(logfollowup, death) ~ rb(LNE) + sex, data=colcancer[1:200,])

## End(Not run)
```

`expectreg.ipc` *Expectile regression for right-censored data*

Description

This function extends expectile regression with inverse probability of censoring (IPC) weights to right-censored data.

Usage

```
expectreg.ipc(
  formula,
  data = NULL,
  smooth = c("schall", "ocv", "aic", "bic", "cvgrid", "lcurve", "fixed"),
  lambda = 1,
  expectiles = NA,
  LAWSmaxCores = 1,
  IPC_weights = c("IPCRR", "IPCKM"),
  KMweights = NULL,
  ci = FALSE,
  hat1 = FALSE
)
```

Arguments

`formula`	A formula object, with the response on the left of the '~' operator, and the terms on the right. The response must be a `Surv` object as returned by the `Surv` function. Only right censored data are allowed. Splines can be specified through the function `rb`.
`data`	Optional data frame containing the variables used in the model, if the data is not explicitly given in the formula.
`smooth`	The smoothing method that shall be used. There are different smoothing algorithms that should prevent overfitting. The '`schall`' algorithm balances variance of errors and contrasts. Ordinary cross- validation '`ocv`' minimizes a score-function using `nlminb` or with a grid search by '`cvgrid`' or the function uses a fixed penalty. The numerical minimizatioin is also possible with AIC or BIC as score. The L-curve is an experimental grid search by Frasso and Eilers.

`lambda`	The fixed penalty can be adjusted. Also serves as starting value for the smoothing algorithms.
`expectiles`	In default setting, the expectiles (0.01,0.02,0.05,0.1,0.2,0.5,0.8,0.9,0.95,0.98,0.99) are calculated. You may specify your own set of expectiles in a vector.
`LAWSmaxCores`	How many cores should maximally be used by parallelization. Currently only implemented for Unix-like OS.
`IPC_weights`	Denotes the kind of IPC weights to use. `IPCRR` weights differ from `IPCKM` weights by modifying the weights for the last observation if it is censored.
`KMweights`	Custom IPC weights can be supplied here. This argument is used by `modreg`.
`ci`	If TRUE, calculates the covariance matrix
`hat1`	If TRUE, the hat matrix for the last asymetry level is calculated. This argument is mainly used by `modreg`.

Details

Fits least asymmetrically weighted squares (LAWS) for each expectile. This function is intended for right-censored data. For uncensored data, `expectreg.ls` should be used instead. This function modifies `expectreg.ls` by adding IPC weights. See Seipp et al. (2021) for details on the IPC weights. P-splines can be used with `rb`. The Schall algorithm is used for choosing the penalty.

Value

A list with the following elements.

`lambda`	The final smoothing parameters for all expectiles and for all effects in a list.
`intercepts`	The intercept for each expectile.
`coefficients`	A matrix of all the coefficients, for each base element a row and for each expectile a column.
`values`	The fitted values for each observation and all expectiles, separately in a list for each effect in the model, sorted in order of ascending covariate values.
`response`	Vector of the response variable.
`covariates`	List with the values of the covariates.
`formula`	The formula object that was given to the function.
`asymmetries`	Vector of fitted expectile asymmetries as given by argument `expectiles`.
`effects`	List of characters giving the types of covariates.
`helper`	List of additional parameters like neighbourhood structure for spatial effects or ϕ for kriging.
`design`	Complete design matrix.
`bases`	Bases components of each covariate.
`fitted`	Fitted values.
`covmat`	Covariance matrix.
`diag.hatma`	Diagonal of the hat matrix. Used for model selection criteria.
`data`	Original data.

smooth_orig	Unchanged original type of smoothing.
KMweights	Vector with IPC weights used in fitting.
aic	Area under the AIC, approximated with a Riemannian sum.
hat	The hat matrix for the last asymmetry level. This is used by modreg.

References

Seipp, A, Uslar, V, Weyhe, D, Timmer, A, Otto-Sobotka, F. Weighted expectile regression for right-censored data. Statistics in Medicine. 2021; 40(25): 5501- 5520. https://doi.org/10.1002/sim.9137

Examples

```
data(colcancer)

# linear effect
expreg <- expectreg.ipc(Surv(logfollowup, death) ~ sex + age, data = colcancer,
                        expectiles = c(0.05, 0.2, 0.5, 0.8, 0.95))
coef(expreg)


# with p-splines, smoothing parameter selection with schall algorithm
expreg2 <- expectreg.ipc(Surv(logfollowup, death) ~ sex + rb(age), data = colcancer)
# smoothing parameter selection with AIC
expreg3 <- expectreg.ipc(Surv(logfollowup, death) ~ sex + rb(age), data = colcancer, smooth = "aic")
# manually selected smoothing parameter
expreg4 <- expectreg.ipc(Surv(logfollowup, death) ~ sex + rb(age), data = colcancer,
                         smooth = "fixed", lambda = 2)

plot(expreg2)
plot(expreg3)
plot(expreg4)
```

---

gumbel *The Gumbel Distribution.*

Description

Density, distribution function, quantile function and random generation for the gumbel distribution with the two parameters location and scale.

Usage

```
dgumbel(x, location = 0, scale = 1)
pgumbel(q, location = 0, scale = 1)
qgumbel(p, location = 0, scale = 1)
rgumbel(n, location = 0, scale = 1)
```

Arguments

q	vector of quantiles.
location	location parameter and mode of the distribution.
scale	scaling parameter, has to be positive.
x	vector of locations.
p	vector of probabilities.
n	number of observations. If $length(n) > 1$, the length is taken to be the number required.

Details

The gumbel distribution has the following density and cdf
$f(x) = (1/scale) * exp((x - location)/scale - exp((x - location)/scale))$, $F(x) = 1 - exp(-exp((x - location)/scale))$. The mode of the distribution is `location`, the variance is $\pi^{2/6}$ $*$ `scale`.

Value

`dgumbel` gives the density, `pgumbel` gives the distribution function, `qgumbel` gives the quantile function, and `rgumbel` generates random deviates.

The length of the result is determined by `n` for `rgumbel`, and is the maximum of the lengths of the numerical arguments for the other functions.

The numerical arguments other than `n` are recycled to the length of the result.

References

Collett, D. (2015). Modelling survival data in medical research, chapter 6. CRC press.

Examples

```
hist(rgumbel(1000))

qg <- qgumbel(0.1, 1, 2)

pgumbel(qg, 1, 2)

ax <- c(1:1000)/100-5
plot(ax,dgumbel(ax), type = 'l')
```

methods — *Methods for fitted modreg model*

Description

Methods for `modreg` objects returned by the mode regression function.

Usage

```
## S3 method for class 'modreg'
coefficients(object, ...)

## S3 method for class 'modreg'
coef(object, ...)

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

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

Arguments

`...`	further arguments passed to or from other methods
`x, object`	A modreg object

Value

`coef` returns a named numerical vector with coefficients

modreg — *Mode-regression for right-censored data*

Description

This function implements semiparametric kernel-based mode regression for right-censored or full data.

Usage

```
modreg(
  formula,
  data = NULL,
  bw = c("Pseudo", "Plugin"),
  lambda = NULL,
```

```
  KMweights = NULL,
  control = NULL
)
```

Arguments

- `formula` — A formula object, with the response on the left of the '~' operator, and the terms on the right. The response must be a `Surv` object as returned by the `Surv` function. Only right censored data are allowed.
- `data` — A data set on which the regression should be performed on. It should consist of columns that have the names of the specific variables defined in `formula`. If `NULL`, the function will look for the data in the environment given by the `formula` argument.
- `bw` — String, either `"Pseudo"`, `"Plugin"` or a fixed numerical value. This determines how bandwidth should be estimated. `"Plugin"` only recommended for uncensored linear mode regression.
- `lambda` — Penalty term for penalized splines. Will be estimated if `NULL`.
- `KMweights` — numerical vector, should be the same length as the response. Inverse probability of censoring weights can be provided here. They will be calculated if `NULL`.
- `control` — A call to `control`. Various control parameters can be supplied here.

Details

Fits mode regression in an iteratively weighted least squares approach. A detailed description of the approach and algorithm can be found in Seipp et al. (2022). In short, kernel-based mode regression leads to minimization of weighted least squares, if the normal kernel is assumed. We use gam for estimation in each iteration. Mode regression is extended to right-censored time-to event data with inverse probability of censoring weights. Hyperparameters (bandwidth, penalty) are determined with a pseudo-likelihood approach for `bw = "Pseudo"`. For "Plugin", plug-in bandwidth selection is performed, as described in Yao and Li (2014). However, this is only justified for uncensored data and mode regression with linear covariate trends or known transformations.

The event time has to be supplied using the `Surv` function. Positive event times with multiplicative relationships should be logarithmized beforehand. Nonlinear trends can be estimated with P-splines, indicated by using `s(covariate, bs = "ps")`. This will be passed down to gam, which is why the same notation is used. Other smooth terms are not tested yet. The whole gam object will be returned but standard errors and other information are not valid. `boot.modreg` can be used for calculation of standard errors and confidence intervals.

Value

This function returns a list with the following properties:

- `reg` — object of class gam. Should be interpreted with care.
- `bw` — The used bandwidth.
- `converged` — logical. Whether or not the iteratively weighted least squares algorithm converged.
- `iterations` — the number of iterations of the final weighted least squares fit

cova	Covariance matrix. Only supplied in case of linear terms and plug-in bandwidth.
KMweights	double vector. Weights used.
called	list. The arguments that were provided.
aic	Pseudo AIC.
pseudologlik	Pseudo log-likelihood.
edf	Effective degrees of freedom
delta	vector. Indicating whether an event has occured (1) or not (0) in the input data.
response	vector with response values
hp_opt	Summary of hyperparameter estimation.

References

Seipp, A., Uslar, V., Weyhe, D., Timmer, A., & Otto-Sobotka, F. (2022). Flexible Semiparametric Mode Regression for Time-to-Event Data. Manuscript submitted for publication.
Yao, W., & Li, L. (2014). A new regression model: modal linear regression. Scandinavian Journal of Statistics, 41(3), 656-671.

Examples

```
data(colcancer)
colcancer80 <- colcancer[1:80, ]

# linear trend
regL <- modreg(Surv(logfollowup, death) ~ sex + age, data = colcancer80)
summary(regL)

# mode regression with P-splines. Convergence criteria are changed to speed up the function
reg <- modreg(Surv(logfollowup, death) ~ sex + s(age, bs = "ps"), data = colcancer80,
control = modreg.control(tol_opt = 10^-2, tol_opt2 = 10^-2, tol = 10^-3))
summary(reg)
plot(reg)

# with a fixed penalty
reg2 <- modreg(Surv(logfollowup, death) ~ sex + s(age, bs = "ps"), data = colcancer80, lambda = 0.1)

# for linear effects and uncensored data, we can use the plug-in bandwidth
regP <- modreg(age ~ sex, data = colcancer, bw = "Plugin")
```

modreg.control *Setting fitting values for* modreg.

Description

This is an internal function of package dirttee which allows control of the numerical options for fitting mode regression. Typically, users will want to modify the defaults if model fitting is slow or fails to converge.

Usage

```
modreg.control(
  StartInterval = sqrt(3),
  nStart = 11,
  nInterim = NULL,
  maxit = 100,
  itInterim = 10,
  tol = 10^-4,
  tol_bw_plugin = 10^-3,
  maxit_bw_plugin = 10,
  maxit_penalty_plugin = 10,
  tol_penalty_plugin = 10^-3,
  tol_regopt = tol * 100,
  tol_opt = 10^-3,
  maxit_opt = 200,
  tol_opt2 = 10^-3,
  maxit_opt2 = 200
)
```

Arguments

StartInterval	Starting values are based on an estimate for the mean and an interval around it. The interval is $+-$ StartInterval $* \sigma$. Default is $\sqrt{3}$.
nStart	Number of starting values, considered in the first iteration. Default is 11.
nInterim	Probably has little impact on speed and result. After itInterim weighted least squares iterations, the number of estimates is reduced from nStart to nInterim estimates. Default is 5.
maxit	Maximum number of iterations for the weighted least squares algorithm. Default is 100.
itInterim	Probably has little impact on speed and result. After itInterim weighted least squares iterations, the number of estimates is reduced from nStart to nInterim estimates. Default is 10.
tol	Convergence criterion for the weighted least squares algorithm. Default is 10^-4.
tol_bw_plugin	Convergence criterion for bandwidth selection in the "Plugin" method. Default is 10^-3.

`maxit_bw_plugin`
: Maximum number of iterations for bandwidth selection in the `"Plugin"` method. Default is 10.

`maxit_penalty_plugin`
: Maximum number of iterations for penalty selection in the `"Plugin"` method. Default is 10.

`tol_penalty_plugin`
: Convergence criterion for penalty selection in the `"Plugin"` method. Default is 10^-3.

`tol_regopt`
: Weighted least squares are recalculated for hyperparameter optimization. This is the convergence criterion within this optimization. Default is `tol` * 100.

`tol_opt`
: Convergence criterion for the first hyperparameter optimizion. Can be increased to reduce compuation time. Default is 10^-3.

`maxit_opt`
: Maximum number of iterations for the first hyperparameter optimizion. Can be lowered to reduce compuation time. Default is 200.

`tol_opt2`
: Convergence criterion for the second hyperparameter optimizion. Default is 10^-3.

`maxit_opt2`
: Maximum number of iterations for the second hyperparameter optimizion. Default is 200.

Details

The algorithm is described in Seipp et al. (2022). To increase the speed of the algorithm, adapting `tol` and `maxit_opt`/`maxit_opt2` and other penalty / hyperparameter optimization parameters are a good starting point.

Value

A list with the arguments as components

References

Seipp, A., Uslar, V., Weyhe, D., Timmer, A., & Otto-Sobotka, F. (2022). Flexible Semiparametric Mode Regression for Time-to-Event Data. Manuscript submitted for publication.
Yao, W., & Li, L. (2014). A new regression model: modal linear regression. Scandinavian Journal of Statistics, 41(3), 656-671.

`plot.modreg` *Plot regression terms for modreg objects*

Description

Plots smooth components of a fitted modreg object.

Usage

```
## S3 method for class 'modreg'
plot(x, ...)
```

Arguments

x	The object to plot, must be of class modreg.
...	Additional arguments to pass to plot.gam.

Details

This function is a wrapper for plot.gam. It displays term plots of smoothed variables. Optionally produces term plots for parametric model components as well. Standard errors will not be displayed but can be estimated by boot_modreg.

Value

The functions main purpose is its side effect of generating plots. It also silently returns a list of the data used to produce the plots, which can be used to generate customized plots.

Examples

```
data(colcancer)
# mode regression with P-splines. Convergence criteria are changed to speed up the function
reg <- modreg(Surv(logfollowup, death) ~ sex + s(age, bs = "ps"), data = colcancer[1:70, ],
control = modreg.control(tol_opt = 10^-2, tol_opt2 = 10^-2, tol = 10^-3))
plot(reg)
```

---

predict.modreg   *Prediction from a fitted modreg model*

Description

Takes a fitted modreg object produced by modreg and produces predictions. New sets of covariates can by supplied through newdata.

Usage

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

Arguments

object	The object to plot, must be of class modreg.
...	Additional arguments to pass to predict.gam.

Details

This function is a wrapper for predict.gam.

Value

A vector or matrix of predictions. For type = "terms" this is a matrix with a column per term.

Examples

```
data(colcancer)
colcancer70 <- colcancer[1:70, ]

mc <- modreg.control(tol_opt = 10^-2, tol_opt2 = 10^-2,
tol = 10^-3)
reg <- modreg(Surv(logfollowup, death) ~ sex + s(age, bs = "ps"), data =
colcancer70, control = mc)
ndat <- data.frame(sex = rep(colcancer70$sex[1], 200), age = seq(50, 90, length = 200))
pr <- predict(reg, newdata = ndat)
```

---

weightsKM — *Inverse probability of censoring weights*

Description

Computes inverse probability of censoring weights.

Usage

```
weightsKM(y, delta)
```

Arguments

y	numerical vector with right-censored follow-up times
delta	numerical vector, same length as y, 1 indicates an event while 0 indicates censoring

Details

Inverse probability of censoring weights are calculated by dividing the event indicator by the Kaplan-Meier estimator of the censoring time. This leads to zero weights for censored observations, while every uncensored event receives a weight larger than 1, representing several censored observations. In the redistribute-to-the-right approach, the last observation always receives a positive weight such that no weight will be lost. Further details can be found in Seipp et al. (2021).

Value

A data frame with 2 coloumns. The first column consists of usual inverse probability of censoring weights. For the second column, IPC weights modified in a redistribute-to-the-right approach are given.

References

Seipp, A., Uslar, V., Weyhe, D., Timmer, A., & Otto-Sobotka, F. (2021). Weighted expectile regression for right-censored data. Statistics in Medicine, 40(25), 5501-5520.

Examples

```
data(colcancer)
kw <- weightsKM(colcancer$logfollowup, colcancer$death)
```

Index