Package 'RobustAFT'

May 7, 2026

Type Package

Title Truncated Maximum Likelihood Fit and Robust Accelerated Failure Time Regression for Gaussian and Log-Weibull Case

Version 1.4-7

Date 2023-08-21

Depends R (>= 3.2.0), stats, graphics, survival, robustbase,DEoptimR

Author Alfio Marazzi <Alfio.Marazzi@unisante.ch>, Jean-Luc Muralti

Maintainer A. Randriamiharisoa <exelami@gmail.com>

Description R functions for the computation of the truncated maximum likelihood and the robust accelerated failure time regression for gaussian and log-Weibull case.

License GPL (>= 2)

LazyLoad yes

NeedsCompilation yes

Repository CRAN

Date/Publication 2023-08-21 16:40:02 UTC

Contents

`RobustAFT-package` *Robust Accelerated Failure Time Model Fitting*

Description

This package computes the truncated maximum likelihood regression estimates described in Marazzi and Yohai (2004) and Locatelli et al. (2010). The error distribution is assumed to follow approximately a Gaussian or a log-Weibull distribution. The cut-off values for outlier rejection are fixed or adaptive. The main functions of this package are `TML.noncensored` and `TML.censored`.

Details

Package:	RobustAFT
Type:	Package
#Version:	1.4-6
#Date:	2023-06-19
License:	GPL-2 or later
LazyLoad:	yes

Author(s)

Alfio Marazzi <Alfio.Marazzi@chuv.ch>, Jean-Luc Muralti

Maintainer: A. Randriamiharisoa <Alex.Randriamiharisoa@chuv.ch>

References

Marazzi A., Yohai V. (2004). Adaptively truncated maximum likelihood regression with asymmetric errors. *Journal of Statistical Planning and Inference*, 122, 271-291.

Locatelli I., Marazzi A., Yohai V. (2010). Robust accelerated failure time regression. *Computational Statistics and Data Analysis*, 55, 874-887.

Marazzi A. (1993) *Algorithm, Routines, and S functions for Robust Statistics*. Wadsworth & Brooks/cole, Pacific Grove, California.

Examples

```
# Example 1. This is the example described in Marazzi and Yohai (2004).
# ---------
# The two following auxiliary functions, not included in the library,
#must be loaded.
# ------------- Auxiliary functions -----------------------------------

SDmux.lw <- function(x,theta,sigma,COV){
# Standard deviation of the conditional mean estimate: log-Weibull case
np <- length(theta); nc <- ncol(COV); nr <- nrow(COV)
if (np!=length(x)) cat("length(x) must be the same as length(theta)")
if (nc!=nr)        cat("COV is not a square matrix")
if (nc!=(np+1))    cat("ncol(COV) must be the same as length(theta)+1")
log.mu.x <- t(x)%*%theta+lgamma(1+sigma) # log of conditional mean estimate
mu.x     <- exp(log.mu.x)                # conditional mean estimate
dg       <- digamma(1+sigma)
COV.TT   <- COV[1:np,1:np]
Var.S    <- COV[(np+1),(np+1)]
COV.TS   <- COV[1:np,(np+1)]
V.mu.x   <- t(x)%*%COV.TT%*%x + dg^2*Var.S + 2*dg*(t(x)%*%COV.TS)
SD.mu.x  <- as.numeric((sqrt(V.mu.x))*mu.x)
SD.mu.x}

plt <- function(LOS,Cost,Adm,theta.fr,sigma.fr,sd0.fr,sd1.fr,theta.ml,
                sigma.ml,sd0.ml,sd1.ml){
# Plot of the conditional mean and confidence intervals: log-Weibull case
par(mfrow=c(1,2),oma=c(0,0,2,0))
plot(LOS,Cost,type="n")
points(LOS[Adm==0],Cost[Adm==0],pch=1)
points(LOS[Adm==1],Cost[Adm==1],pch=16,col=2)
x0t   <- x0%*%theta.fr; x1t <- x1t <- x1%*%theta.fr
lines(l0,exp(x0t)*gamma(1+sigma.fr))
lines(l0,exp(x1t)*gamma(1+sigma.fr),col=2)
z0min <- exp(x0t)*gamma(1+sigma.fr)-2.576*sd0.fr
z0max <- exp(x0t)*gamma(1+sigma.fr)+2.576*sd0.fr
z1min <- exp(x1t)*gamma(1+sigma.fr)-2.576*sd1.fr
z1max <- exp(x1t)*gamma(1+sigma.fr)+2.576*sd1.fr
lines(l0,z0min,lty=2,col=1)
lines(l0,z0max,lty=2,col=1)
lines(l0,z1min,lty=2,col=1)
lines(l0,z1max,lty=2,col=1)
polygon(c(l0,rev(l0)), c(z0min,rev(z0max)), border=FALSE, density=10, angle=90)
polygon(c(l0,rev(l0)), c(z1min,rev(z1max)), border=FALSE, density=12, angle=90,col=2)
plot(LOS,Cost,type="n")
points(LOS[Adm==0],Cost[Adm==0],pch=1)
points(LOS[Adm==1],Cost[Adm==1],pch=16,col=2)
x0t   <- x0%*%theta.ml; x1t <- x1t <- x1%*%theta.ml
lines(l0,exp(x0t)*gamma(1+sigma.ml))
lines(l0,exp(x1t)*gamma(1+sigma.ml),col=2)
z0min <- exp(x0t)*gamma(1+sigma.ml)-2.576*sd0.ml
z0max <- exp(x0t)*gamma(1+sigma.ml)+2.576*sd0.ml
z1min <- exp(x1t)*gamma(1+sigma.ml)-2.576*sd1.ml
```

```
z1max <- exp(x1t)*gamma(1+sigma.ml)+2.576*sd1.ml
lines(l0,z0min,lty=2,col=1)
lines(l0,z0max,lty=2,col=1)
lines(l0,z1min,lty=2,col=1)
lines(l0,z1max,lty=2,col=1)
polygon(c(l0,rev(l0)), c(z0min,rev(z0max)), border=FALSE, density=10, angle=90)
polygon(c(l0,rev(l0)), c(z1min,rev(z1max)), border=FALSE, density=12, angle=90,col=2)}

#------ End of auxiliary functions --------------------------------------------------

library(RobustAFT)

data(D243)
Cost <- D243$Cost                              # Cost (Swiss francs)
LOS <- D243$LOS                                # Length of stay (days)
Adm <- D243$Typadm; Adm <- (Adm==" Urg")*1   # Type of admission
                                               # (0=on notification, 1=Emergency)
Ass <- D243$Typass; Ass <- (Ass=="P"   )*1   # Type of insurance (0=usual, 1=private)
Age <- D243$age                                # Age (years)
Dst <- D243$dest;   Dst <- (Dst=="DOMI")*1   # Destination (1=Home, 0=another hospital)
Sex <- D243$Sexe;   Sex <- (Sex=="M"   )*1   # Sex (1=Male, 0=Female)

## Not run:
# Plot data
par(mfrow=c(1,2))
plot(LOS,Cost); plot(log(LOS),log(Cost))

# log-Weibull fits
# ----------------
# Full robust model
zwff     <- TML.noncensored(log(Cost)~log(LOS)+Adm+Ass+Age+Sex+Dst,
            errors="logWeibull")
summary(zwff)

# Reduced model
zwfr     <- update(zwff,log(Cost)~log(LOS)+Adm)
summary(zwfr)

# Residual plots
par(mfrow=c(1,2))
plot(zwfr,which=c(1,3))

# Plot robust predictions on log-log scale
par(mfrow=c(1,1))
l0       <- seq(from=2,to=60,by=0.5)
x0       <- as.matrix(cbind(1,log(l0),0))
x1       <- as.matrix(cbind(1,log(l0),1))
plot(log(LOS),log(Cost),type="n")
points(log(LOS[Adm==1]),log(Cost[Adm==1]),pch=16,col=2)
points(log(LOS[Adm==0]),log(Cost[Adm==0]),pch=1)
lines(log(l0),predict(zwfr,x0))
lines(log(l0),predict(zwfr,x1),col=2)
```

```
# Maximum likelihood : full model
zmlf     <- TML.noncensored(log(Cost)~log(LOS)+Adm+Ass+Age+Sex+Dst,
            errors="logWeibull",cu=100)
summary(zmlf)

# Maximum likelihood : reduced model
zmlr     <- update(zmlf,log(Cost)~log(LOS)+Adm)
summary(zmlr)

# Plot conditional means and confidence intervals
l0       <- seq(from=2,to=62,by=0.5)
x0       <- as.matrix(cbind(1,log(l0),0))
x1       <- as.matrix(cbind(1,log(l0),1))
theta.fr <- coef(zwfr)
sigma.fr <- zwfr$v1
COV.fr   <- vcov(zwfr)
sd0.fr   <- apply(x0,1,SDmux.lw,theta.fr,sigma.fr,COV.fr)
sd1.fr   <- apply(x1,1,SDmux.lw,theta.fr,sigma.fr,COV.fr)
theta.ml <- coef(zmlr)
sigma.ml <- zmlr$v1
COV.ml   <- zmlr$COV
sd0.ml   <- apply(x0,1,SDmux.lw,theta.ml,sigma.ml,COV.ml)
sd1.ml   <- apply(x1,1,SDmux.lw,theta.ml,sigma.ml,COV.ml)
plt(LOS,Cost,Adm,theta.fr,sigma.fr,sd0.fr,sd1.fr,theta.ml,sigma.ml,sd0.ml,sd1.ml)

# Gaussian fits (for comparison)
# ------------------------------
# Reduced model
zgfr <- TML.noncensored(log(Cost)~log(LOS)+Adm,errors="Gaussian")
summary(zgfr)

# Residual plots
par(mfrow=c(1,2))
plot(zgfr,which=c(1,3))

# Classical Gaussian fit
lr <- lm(log(Cost)~log(LOS)+Adm)
summary(lr)

# Compare several fits
# --------------------
comp <- fits.compare(TML.logWeibull=zwfr,ML.logWeibull=zmlr,least.squares=lr)
comp
plot(comp,leg.position=c("topleft","topleft","bottomleft")) # click on graphics

## End(Not run)
#
#-------------------------------------------------------------------------------
#
# Example 2. This is the example described in Locatelli Marazzi and Yohai (2010).
# ---------
# This is the example described in Locatelli et al. (2010).
# The estimates are slighty different due to changes in the algorithm for the
```

```
# final estimate.
## Not run:
# Remove data of Example 1
rm(Cost,LOS,Adm,Ass,Age,Dst,Sex)

data(MCI)
attach(MCI)

# Exploratory Analysis

par(mfrow=c(1,1))

plot(Age,log(LOS),type= "n",cex=0.7)

# (1) filled square   : regular,   complete
# (2) empty  square   : regular,   censored
# (3) filled triangle : emergency, complete
# (4) empty  triangle : emergency, censored

points(Age[Dest==1 & TypAdm==0], log(LOS)[Dest==1 & TypAdm==0], pch=15,cex=0.7)  # (1)
points(Age[Dest==0 & TypAdm==0], log(LOS)[Dest==0 & TypAdm==0], pch=0, cex=0.7)  # (2)
points(Age[Dest==1 & TypAdm==1], log(LOS)[Dest==1 & TypAdm==1], pch=17,cex=0.7)  # (3)
points(Age[Dest==0 & TypAdm==1], log(LOS)[Dest==0 & TypAdm==1], pch=2, cex=0.7)  # (4)

# Maximum Likelihood
ML   <- survreg(Surv(log(LOS), Dest) ~ TypAdm*Age, dist="gaussian")
summary(ML)
B.ML <- ML$coef; S.ML <- ML$scale
abline(c(B.ML[1]        ,B.ML[3]        ),lwd=1,col="grey",lty=1)
abline(c(B.ML[1]+B.ML[2],B.ML[3]+B.ML[4]),lwd=1,col="grey",lty=1)

# Robust Accelerated Failure Time Regression with Gaussian errors
ctrol.S   <- list(N=150, q=5, sigma0=1, MAXIT=100, TOL=0.001,seed=123)

ctrol.ref <- list(maxit.sigma=2,tol.sigma=0.0001,maxit.Beta=2,tol.Beta=0.0001,
          Maxit.S=50, tol.S.sigma=0.001, tol.S.Beta=0.001,alg.sigma=1,nitmon=FALSE)

ctrol.tml <- list(maxit.sigma=50,tol.sigma=0.0001,maxit.Beta=50,tol.Beta=0.0001,
    Maxit.TML=50, tol.TML.sigma=0.001, tol.TML.Beta=0.001, alg.sigma=1,nitmon=FALSE)

WML       <- TML.censored(log(LOS)~TypAdm*Age,data=MCI,delta=Dest,otp="adaptive",
             control.S=ctrol.S,control.ref=ctrol.ref,control.tml=ctrol.tml)

summary(WML)

B.WML     <-coef(WML)
abline(c(B.WML[1]         ,B.WML[3]         ),lty=1, col="red")
abline(c(B.WML[1]+B.WML[2],B.WML[3]+B.WML[4]),lty=1, col="red")
#
detach(MCI)

## End(Not run)
```

D243 *Sample of 100 hospital stays for medical back problems*

Description

Sample of 100 patients hospitalized for medical back problems in Switzerland

Usage

```
data(D243)
```

Format

A data frame with 100 observations on the following 11 variables.

`Sexe` Gender: M=Male, F=Female

`age` Age in years

`dest` Destination: DOMI=Home else=another hospital

`Typadm` Type of admission: Urg=Emergency else=on notification

`Typass` Type of insurance: P=Private else=usual

`LOS` Length of stay (days)

`APDRG` DRG code: Always 243

`Cost` Cost (Swiss francs)

`CSansInv` Intermediate cost

`BBDaggr` a numeric vector

`BBD` a numeric vector

Examples

```
data(D243)
```

dfcomn2 *Assigns values to the ROBETH parameters included in common blocks*

Description

See Marazzi A. (1993), p.405

Usage

```
dfcomn2(ipsi = -9, c = -1.345, h1 = -1.7, h2 = -3.4, h3 = -8.5,
       xk = -1.548, d = -1.345, beta = -0.5, bet0 = -1, iucv = -1,
       a2 = 0, b2 = -3, chk = -9, ckw = -2, bb = -1, bt = -1,
       cw = -1, em = -1.345, cr = -2, vk = -1, np = -2, nu = -1,
       v7 = -1, iwww = -1)
```

Arguments

ipsi	Option parameter for the choice of ψ. Set -4 <= ipsi <= 4
c	Parameter c of the Huber function
h1	Parameter $h1$ of the Hampel function
h2	Parameter $h2$ of the Hampel function
h3	Parameter $h3$ of the Hampel function
xk	Parameter k of the rescaled Tukey biweight
d	See reference
beta	Parameter β to makeσ estimate asymptotically unbiased
bet0	Parameter β_0 to makeσ estimate asymptotically unbiased
iucv	Option parameter for the choice of u(s), u'(s), v(s), v'(s), w(s) or w'(s)
a2	Parameter a^2 of Huber's mimimax u-function
b2	Parameter b^2 of Huber's mimimax u-function
chk	Parameter c of the Hampel-Krasker u-function
ckw	Parameter c of the Krasker-Welsch u-function
bb	Parameter b of the Mallows-unstandard u-function
bt	Option parameter for w(s) or w'(s)
cw	Option parameter for w(s) or w'(s)
em	Parameter em for unstandard u-function
cr	Parameter cr for unstandard u-function
vk	Parameter vk for unstandard u-function
np	Parameter np for unstandard u-function
nu	Parameter nu for unstandard u-function
v7	Parameter v for unstandard u-function
iwww	Option parameter for the choice of $\bar{\omega}$. Set 0 <= iwww <= 3

Value

See reference

References

Marazzi A. (1993) *Algorithm, Routines, and S functions for Robust Statistics*. Wadsworth & Brooks/cole, Pacific Grove, California. p.405

`fits.compare` *Numerical comparison of several fits*

Description

Creates a class "fits.compare" object allowing the user to display model summary statistics in a form allowing easy comparison of models.

Usage

```
fits.compare(...)
```

Arguments

`...`	one or more class "lm", class "lm.robust" or class "TML" objects. Names given to objects in the list are used as labeling information in the printed output.

Details

The `fits.compare` function processes its arguments one at a time to create a named list of objects. The object returned is a member of class "fits.compare". Because of differences in the computed summary statistics, the list of input objects is currently limited to class "lm", class "lm.robust" and class "TML" objects. The `print.fits.compare` function displays a textual comparison of the input models, and the `plot.fits.compare` function provides comparative plots.

Value

An object of class "fits.compare" containing the list of input models to be compared.

See Also

`TML.noncensored`, `plot.fits.compare`

Examples

```
## Not run:
     data(D243)
     Cost <- D243$Cost                              # Cost (Swiss francs)
     LOS  <- D243$LOS                               # Length of stay (days)
     Adm  <- D243$Typadm; Adm <- (Adm==" Urg")*1    # Type of admission
                                                    # (0=on notification, 1=Emergency)

     lwrob <- TML.noncensored(log(Cost)~log(LOS)+Adm, errors="logWeibull")
     grob  <- TML.noncensored(log(Cost)~log(LOS)+Adm)
     reg   <- lm(log(Cost)~log(LOS)+Adm)

     fits.compare(least.squares=reg, TML.logWeibull=lwrob, TML.Gaussian=grob)

## End(Not run)
```

MCI *Sample of 75 Hospital Stays*

Description

Sample of 75 hospital for major cardiovascular interventions

Usage

```
data(MCI)
```

Format

A data frame with 75 observations on the following 6 variables.

`Sex` Gender: 1=Female, 2=Male

`Age` Age in years

`LOS` Length of stay (days)

`TypAdm` Type of admission: 1=Emergency 0=on notification

`Dest` Destination: 1=Home 0=another hospital

`Cost` Cost (Swiss francs)

Examples

```
data(MCI)
```

plot.fits.compare *Plot Method for "fits.compare" objects*

Description

Comparative plots for objects of class "fits.compare".

Usage

```
## S3 method for class 'fits.compare'
plot(x, xplots = FALSE, ask = TRUE, which = 1:4,
             leg.position = c("topleft", "topleft", "topleft"), ...)
```

Arguments

x	An object of class "fits.compare", usually, a result of a call to fits.compare.
xplots	If xplots=TRUE, plots of the independent variables versus the residuals are produced.
ask	If ask=TRUE, plot.fits.compare() operates in interactive mode.
which	If a subset of the plots is required, specify a subset of the numbers 1:4.
leg.position	A vector of character string specifying the legend position of the second, third and fourth plots.
...	Optional arguments for par.

Details

For clarity reasons, at most three models should be compared. Four default plots (selectable by which) are produced: histograms of the residuals of each model, a residual Q-Q plot, response against fitted values and residuals against fitted values. Additional plots are produced if xplots=TRUE.

See Also

fits.compare, plot.default, plot.TML

Examples

```
## Not run:
     data(D243)
     Cost <- D243$Cost                                # Cost (Swiss francs)
     LOS  <- D243$LOS                                 # Length of stay (days)
     Adm  <- D243$Typadm; Adm <- (Adm==" Urg")*1       # Type of admission
                                                      # (0=on notification, 1=Emergency)

     lwrob <- TML.noncensored(log(Cost)~log(LOS)+Adm, errors="logWeibull")
     reg   <- lm(log(Cost)~log(LOS)+Adm)

     comp  <- fits.compare(least.squares=reg, TML.logWeibull=lwrob)
     plot(comp, leg.position=c("topleft", "topleft", "bottomleft"), xplots=TRUE)

## End(Not run)
```

---

plot.TML *Plot Method for "TML" objects*

Description

Diagnostic plots for elements of class "TML". Three plots (selectable by which) are currently available: a residual Q-Q plot, a plot of response against fitted values and a plot of standardized residuals against fitted values.

Usage

```
## S3 method for class 'TML'
plot(x, which = 1:3, caption = c("Residual QQ-plot",
  "Response vs. Fitted Values", "Standardized Residuals vs. Fitted Values"),
  panel = points, sub.caption = deparse(x$call$formula), main = "",
  ask = prod(par("mfcol")) < length(which) && dev.interactive(), ...)
```

Arguments

x	An object of class "TML", usually, a result of a call to TML.noncensored or TML.censored.
which	If a subset of the plots is required, specify a subset of the numbers 1:3.
caption	Caption for the different plots.
panel	Panel.
sub.caption	Sub titles.
main	Main title.
ask	If ask=TRUE, plot.TML() operates in interactive mode.
...	Optional arguments for par.

Details

The residual Q-Q plot is build with respect to the errors argument of the object. This means that the expected order statistics are calculated either for a Gaussian or a log-Weibull distribution. The two horizontal dotted lines on the first and the third plots represent the upper and lower cut-off values for outlier rejection. Observations that were not retained for the estimation (outliers) are identified on the third plot.

See Also

TML.noncensored, TML.censored, plot.default

Examples

```
## Not run:
     data(D243)
     Cost <- D243$Cost                              # Cost (Swiss francs)
     LOS  <- D243$LOS                               # Length of stay (days)
     Adm  <- D243$Typadm; Adm <- (Adm==" Urg")*1    # Type of admission
                                                    # (0=on notification, 1=Emergency)

     # Truncated maximum likelihood regression with log-Weibull errors
     w  <- TML.noncensored(log(Cost)~log(LOS)+Adm, errors="logWeibull",
           otp="adaptive", control=list(fastS=TRUE))

     plot(w)
     plot(w, which = 1)
     plot(w, which = 2)
     plot(w, which = 3)
```

```
## End(Not run)
```

`predict.TML` *Predict method for "TML" objects*

Description

Obtains predictions from a fitted Truncated Maximum Likelihood (TML) object.

Usage

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

Arguments

`object`	An object of class "TML", usually, a result of a call to `TML.noncensored` or `TML.censored`.
`newdata`	Optionally, a vector, a matrix or a data frame containing the variables with which to predict. If omitted, the fitted values of `object` are returned.
`...`	Additional arguments affecting the predictions produced.

Details

`newdata` must have the same number of variables (that is of columns) as the model. If `object` is a model with an intercept, `newdata` must have a first column of 1.

Value

Returns a vector of predictions.

See Also

`TML.noncensored`, `TML.censored`, `predict`

Examples

```
## Not run:
     data(D243)
     Cost <- D243$Cost                              # Cost (Swiss francs)
     LOS  <- D243$LOS                               # Length of stay (days)
     Adm  <- D243$Typadm; Adm <- (Adm==" Urg")*1    # Type of admission
                                                    # (0=on notification, 1=Emergency)

     # Fitting the model
     z    <- TML.noncensored(log(Cost)~log(LOS)+Adm, errors="logWeibull")
```

```
# With a vector of data
vec  <- c(1, 2.4, 1)
predict(object = z, newdata = vec)
# With a matrix of data
mat  <- matrix(c(1,1,2.4,2.7,1,0), ncol=3)
predict(z, mat)
# With a data frame
dat  <- as.data.frame(cbind("intercept"=c(1,1,1), "log(LOS)"=c(2.4,2.7,2.2),
        "Adm"=c(1,0,1)))
predict(z, dat)

## End(Not run)
```

---

summary.TML *Summarizing Truncated Maximum Likelihood regression*

Description

Summary and print methods for R object of class "TML" and print method for the summary object. Further, methods fitted(), residuals(), weights() or update() work (via the default methods), and coef(), vcov() have explicitly defined TML methods.

Usage

```
## S3 method for class 'TML'
summary(object, ...)
## S3 method for class 'TML'
print(x, digits = max(3, getOption("digits") - 3), ...)
## S3 method for class 'TML'
coef(object, ...)
## S3 method for class 'TML'
vcov(object, ...)

## S3 method for class 'summary.TML'
print(x, digits = max(3, getOption("digits") - 3),
  signif.stars = getOption("show.signif.stars"), ...)
```

Arguments

object	An object of class "TML", usually, a result of a call to TML.noncensored or TML.censored.
...	Potentially more arguments passed to methods.
digits	Number of digits for printing, see digits in options.
x	An object of class "TML" or "summary.TML".
signif.stars	Logical indicating if the P-values should be visualized by so called "significance stars".

Details

summary.TML returns an object of class "summary.TML".

print.TML returns a printed summary of object of class "TML".

print.summary.TML tries to be smart about formatting the coefficients, standard errors, etc, and gives "significance stars" if signif.stars is TRUE (as per default when options where not changed).

coef.TML returns the final coefficient estimates (value th1 of a "TML" object), and vcov.TML returns the covariance matrix of the final estimates (value CV1 of a "TML" object).

Value

An object of class "summary.TML" is a list with the following components:

call	The component from object.
terms	The component from object.
residuals	The component from object.
fitted.values	The component from object.
tn	The component from object.
coefficients	The matrix of coefficients, standard errors, t-values and p-values. Aliased coefficients are omitted.
aliased	Named logical vector showing if the original coefficients are aliased.
df	Degrees of freedom, a 3-vector (p, n-p, p*), the last being the number of non-aliased coefficients.
sigma	The final scale estimate from object.
cutoff.values	A vector of the final lower and upper cut-off values from object.

See Also

TML.noncensored, TML.censored, summary, print

Examples

```
## Not run:
     data(D243)
     Cost <- D243$Cost                              # Cost (Swiss francs)
     LOS  <- D243$LOS                               # Length of stay (days)
     Adm  <- D243$Typadm; Adm <- (Adm==" Urg")*1    # Type of admission
                                                    # (0=on notification, 1=Emergency)
     Ass  <- D243$Typass; Ass <- (Ass=="P"   )*1    # Type of insurance
                                                    # (0=usual, 1=private)
     Age  <- D243$age                               # Age (years)
     Dst  <- D243$dest;   Dst <- (Dst=="DOMI")*1    # Destination
                                                    # (1=Home, 0=another hospital)
     Sex  <- D243$Sexe;   Sex <- (Sex=="M"   )*1    # Sex (1=Male, 0=Female)

     # Truncated maximum likelihood regression with Gaussian errors
     z    <- TML.noncensored(log(Cost)~log(LOS)+Adm+Ass+Age+Dst+Sex, otp="adaptive",
```

```
            cov="nonparametric", control=list(fastS=TRUE))

     z                 # -> print.TML(....)
     sumz <- summary(z) # -> summary.TML(....)
     sumz              # -> print.summary.TML(....)
     coef(z)           # -> coef.TML(....)
     vcov(z)           # -> vcov.TML(....)

## End(Not run)
```

`TML.censored` *Truncated Maximum Likelihood Regression With Censored Observations*

Description

This function computes the truncated maximum likelihood estimates of accelerated failure time regression described in Locatelli et al. (2010). The error distribution is assumed to follow approximately a Gaussian or a log-Weibull distribution. The cut-off values for outlier rejection are fixed or adaptive.

Usage

```
TML.censored(formula, delta, data, errors = "Gaussian",  initial = "S",
             input = NULL, otp = "fixed", cov=TRUE, cu = NULL, control.S=list(),
              control.ref=list(), control.tml=list())
```

Arguments

`formula`	A `formula`, i.e., a symbolic description of the model to be adjusted (cf. `glm` or `lm`).
`data`	An optional data frame containing the variables in the model. If not found in `data`, the variables are taken from `environment(formula)`, typically the environment from which `robaft` is called.
`delta`	Vector of 0 and 1. • 0: censored observation. • 1: complete observation.
`errors`	• "Gaussian": the error distribution is assumed to be Gaussian. • "logWeibull" : the error distribution is assumed to be log-Weibull.
`initial`	• "S": initial S-estimate. • "input": the initial estimate is given on input.
`input`	A list(theta=c(...),sigma=...): initial input estimates where theta is a vector of p coefficients and sigma a scalar scale. Required when initial="input".
`otp`	• "adaptive": adaptive cut-off.

- "fixed": non adaptive cut-off.

cov	If TRUE the covariance matrix is computed.
cu	Preliminary minimal upper cut-off. The default is 2.5 in the Gaussian case and 1.855356 in the log-Weibull case.
control.S	A list of control parameters for the computation of the initial S estimates. See the function TML.censored.control.S for the default values.
control.ref	A list of control parameters for the refinement algorithm of the initial S estimates. See the function TML.censored.control.ref for the default values.
control.tml	AA list of control parameters for the computation of the final estimates. See the function TML.censored.control.tml for the default values.

Value

TML.censored returns an object of class "TML". The function summary can be used to obtain or print a summary of the results. The generic extractor functions fitted, residuals and weights can be used to extract various elements of the object returned by TML.censored. The function update can be used to update the model.

An object of class "TML" is a list with at least the following components:

th0	Initial coefficient estimates.
v0	Initial scale estimate.
nit.ref	Reached number of iteration in the refinement step for the initial estimates.
th1	Final coefficient estimates.
v1	Final scale estimate.
nit.tml	Number of iterations reached in IRLS algorithm for the final estimates.
tu, tl	Final cut-off values.
alpha	Estimated proportion of retained observations.
tn	Number of retained observations.
weights	Vector of weights (0 for rejected observations, 1 for retained observations).
COV	Covariance matrix of the final estimates (th1[1],...,th1[p],v1) (where p=ncol(X)).
residuals	Residuals of noncensored observations are calculated as response minus fitted values. For censored observations, the the expected residuals given that the response is larger than the recorded censored value are provided.
fitted.values	The fitted mean values.
call	The matched call.
formula	The formula supplied.
terms	The terms object used.
data	The data argument.

References

Locatelli I., Marazzi A., Yohai V. (2010). Robust accelerated failure time regression. *Computational Statistics and Data Analysis*, 55, 874-887.

See Also

TML.censored.control.ref, TML.censored.control.tml, TML.censored.control.S, TML.noncensored

Examples

```
     # This is the example described in Locatelli et al. (2010).
     # The estimates are slighty different than those of the paper due to changes
     # in the algorithm for the final estimate.
     #
## Not run:
     data(MCI)
     attach(MCI)

     # Exploratory Analysis
     plot(Age,log(LOS),type= "n",cex=0.7)

     # (1) filled square : regular,   complete
     # (2) empty  square : regular,   censored
     # (3) filled triangle : emergency, complete
     # (4) empty  triangle : emergency, censored

    points(Age[Dest==1 & TypAdm==0], log(LOS)[Dest==1 & TypAdm==0], pch=15,cex=0.7) # (1)
    points(Age[Dest==0 & TypAdm==0], log(LOS)[Dest==0 & TypAdm==0], pch=0, cex=0.7) # (2)
    points(Age[Dest==1 & TypAdm==1], log(LOS)[Dest==1 & TypAdm==1], pch=17,cex=0.7) # (3)
    points(Age[Dest==0 & TypAdm==1], log(LOS)[Dest==0 & TypAdm==1], pch=2, cex=0.7) # (4)

     # Maximum Likelihood
     ML   <- survreg(Surv(log(LOS), Dest) ~ TypAdm*Age, dist="gaussian")
     summary(ML)
     B.ML <- ML$coef
     S.ML <- ML$scale

     abline(c(B.ML[1]        ,B.ML[3]        ),lwd=1,col="grey",lty=1)
     abline(c(B.ML[1]+B.ML[2],B.ML[3]+B.ML[4]),lwd=1,col="grey",lty=1)

     # Robust Accelerated Failure Time Regression with Gaussian errors
     ctrol.S   <- list(N=150, q=5, sigma0=1, MAXIT=100, TOL=0.001,seed=123)

     ctrol.ref <- list(maxit.sigma=2,tol.sigma=0.0001,maxit.Beta=2,tol.Beta=0.0001,
           Maxit.S=50, tol.S.sigma=0.001, tol.S.Beta=0.001,alg.sigma=1,nitmon=FALSE)

     ctrol.tml <- list(maxit.sigma=50,tol.sigma=0.0001,maxit.Beta=50,tol.Beta=0.0001,
      Maxit.TML=50, tol.TML.sigma=0.001, tol.TML.Beta=0.001, alg.sigma=1,nitmon=FALSE)

     WML<-TML.censored(log(LOS)~TypAdm*Age,data=MCI,delta=Dest,otp="adaptive",
          control.S=ctrol.S,control.ref=ctrol.ref,control.tml=ctrol.tml)

     summary(WML)

     B.WML<-coef(WML)
     abline(c(B.WML[1]         ,B.WML[3]         ),lty=1, col="red")
     abline(c(B.WML[1]+B.WML[2],B.WML[3]+B.WML[4]),lty=1, col="red")
```

```
## End(Not run)
```

`TML.censored.control.ref` *Control parameters for the refinement IRLS algorithm of the TML.censored initial S-estimates*

Description

Auxiliary function for `TML.censored`. Typically only used internally by `TML.censored`, but may be used to provide a control argument. This function provides default values.

Usage

```
TML.censored.control.ref(maxit.sigma=2, tol.sigma=0.0001, maxit.Beta=2,
      tol.Beta=0.0001, Maxit.S=50, tol.S.sigma=0.001, tol.S.Beta=0.001,
      alg.sigma=1, nitmon = FALSE)
```

Arguments

`maxit.sigma`	Maximum number of iterations in scale step.
`tol.sigma`	Tolerance for sigma in scale step.
`maxit.Beta`	Maximum number of iterations in coefficient step.
`tol.Beta`	Tolerance for coefficients in coefficient step.
`Maxit.S`	Maximum number of iterations in global cycle.
`tol.S.sigma`	Tolerance for sigma in global cycle.
`tol.S.Beta`	Tolerance for coefficients in global cycle.
`alg.sigma`	Type of algorithm in scale step: • 1: fixed point algorithm. • 2: regula falsi.
`nitmon`	Set to TRUE if iteration monitoring is desired. Default=FALSE.

Value

A list with components named as the arguments.

See Also

`TML.censored`, `TML.censored.control.S`, `TML.censored.control.tml`

Examples

```
      ### In the example(TML.censored), the control argument for the refinement
      ### algorithm can be built using this function:
## Not run:
      data(MCI)
      attach(MCI)

      # Robust Accelerated Failure Time Regression with Gaussian errors

      ctrol.ref <- TML.censored.control.ref(maxit.sigma=2,tol.sigma=0.0001,
                   maxit.Beta=2,tol.Beta=0.0001, Maxit.S=50, tol.S.sigma=0.001,
                   tol.S.Beta=0.001,alg.sigma=1,nitmon=FALSE)

      ctrol.tml <- list(maxit.sigma=50,tol.sigma=0.0001,maxit.Beta=50,
                   tol.Beta=0.0001, Maxit.TML=50, tol.TML.sigma=0.001,
                   tol.TML.Beta=0.001, alg.sigma=1,nitmon=FALSE)

      WML<-TML.censored(log(LOS)~TypAdm*Age,data=MCI,delta=Dest,otp="adaptive",
                   control.ref=ctrol.ref,control.tml=ctrol.tml)

      summary(WML)

## End(Not run)
```

`TML.censored.control.S` *Control parameters for the computation of the initial S estimates in TML.censored*

Description

Auxiliary function for `TML.censored`. Typically only used internally by `TML.censored`, but may be used to provide a control argument. This function provides default values.

Usage

```
TML.censored.control.S(N=100, q=6, sigma0=1, MAXIT=100, TOL=0.01, seed=153)
```

Arguments

`N`	Number of subsamples.
`q`	Subsample size.
`sigma0`	Initial value of scale.
`MAXIT`	Maximum number of iterations for solving the equation for scale.
`TOL`	Relative tolerance for scale.
`seed`	Seed for the random number generator.

Value

A list with components named as the arguments.

See Also

TML.censored, TML.censored.control.ref, TML.censored.control.tml

Examples

```
    ### In the example(TML.censored), the control argument for the refinement
    ### algorithm can be built using this function:

 ## Not run:
    data(MCI)
    attach(MCI)

    # Robust Accelerated Failure Time Regression with Gaussian errors
    ctrol.S   <- list(N=150, q=5, sigma0=1, MAXIT=100, TOL=0.001,seed=123)

    ctrol.ref <- TML.censored.control.ref(maxit.sigma=2,tol.sigma=0.0001,
              maxit.Beta=2,tol.Beta=0.0001, Maxit.S=50, tol.S.sigma=0.001,
              tol.S.Beta=0.001,alg.sigma=1,nitmon=FALSE)

    ctrol.tml <- list(maxit.sigma=50,tol.sigma=0.0001,maxit.Beta=50,
              tol.Beta=0.0001, Maxit.TML=50, tol.TML.sigma=0.001,
              tol.TML.Beta=0.001, alg.sigma=1,nitmon=FALSE)

    WML       <- TML.censored(log(LOS)~TypAdm*Age,data=MCI,delta=Dest,
              otp="adaptive",control.S=ctrol.S,control.ref=ctrol.ref,
              control.tml=ctrol.tml)

    summary(WML)

## End(Not run)
```

TML.censored.control.tml

Control parameters for the IRLS algorithm of the final TML.censored estimates

---

Description

Auxiliary function for TML.censored. Typically only used internally by TML.censored, but may be used to provide a control argument. This function provides default values.

Usage

```
TML.censored.control.tml(maxit.sigma=20, tol.sigma=0.0001, maxit.Beta=20,
    tol.Beta=0.0001,Maxit.TML=50, tol.TML.sigma=0.001, tol.TML.Beta=0.001,
    alg.sigma=1, nitmon = FALSE)
```

Arguments

maxit.sigma	Maximum number of iterations in scale step.
tol.sigma	Tolerance for sigma in scale step.
maxit.Beta	Maximum number of iterations in coefficient step.
tol.Beta	Tolerance for coefficients in coefficient step.
Maxit.TML	Maximum number of iterations for global cycle.
tol.TML.sigma	Tolerance for sigma in global cycle.
tol.TML.Beta	Tolerance for coefficients in global cycle.
alg.sigma	Type of algorithm in scale step: • 1: fixed point algorithm. • 2: regula falsi.
nitmon	Set to TRUE if iteration monitoring is desired. Default=FALSE.

Value

A list with components named as the arguments.

See Also

TML.censored, TML.censored.control.S, TML.censored.control.ref

Examples

```
    ### In the example(TML.censored), the control argument for the final estimates
### can be built using this function:

## Not run:
    data(MCI)
    attach(MCI)

    # Robust Accelerated Failure Time Regression with Gaussian errors
    ctrol.ref <- list(maxit.sigma=2,tol.sigma=0.0001,maxit.Beta=2,tol.Beta=0.0001,
          Maxit.S=50, tol.S.sigma=0.001, tol.S.Beta=0.001,alg.sigma=1,nitmon=FALSE)

    ctrol.tml <- TML.censored.control.tml(maxit.sigma=50,tol.sigma=0.0001,
          maxit.Beta=50,tol.Beta=0.0001, Maxit.TML=50, tol.TML.sigma=0.001,
          tol.TML.Beta=0.001, alg.sigma=1,nitmon=FALSE)

    WML   <- TML.censored(log(LOS)~TypAdm*Age,data=MCI,delta=Dest,otp="adaptive",
          control.ref=ctrol.ref,control.tml=ctrol.tml)

    summary(WML)

## End(Not run)
```

`TML.noncensored` *Truncated Maximum Likelihood Regression Without Censored Observations*

Description

This function computes the truncated maximum likelihood regression estimate described in Marazzi and Yohai (2004). The error distribution is assumed to follow approximately a Gaussian or a log-Weibull distribution. The cut-off values for outlier rejection are fixed or adaptive.

Usage

```
TML.noncensored(formula, data, errors = "Gaussian", cu = NULL,
                initial = "S",otp = "fixed", cov = "parametric",
                input = NULL, control = list(), ...)
```

Arguments

`formula`	A `formula`, i.e., a symbolic description of the model to be fit (cf. `glm` or `lm`).
`data`	An optional data frame containing the variables in the model. If not found in `data`, the variables are taken from `environment(formula)`, typically the environment from which `TML.noncensored` is called.
`errors`	• "Gaussian": the error distribution is assumed to be Gaussian. • "logWeibull" : the error distribution is assumed to be log-Weibull.
`cu`	Preliminary minimal upper cut-off. The default is 2.5 in the Gaussian case and 1.855356 in the log-Weibull case.
`initial`	• "S" : initial S-estimate. • "input" : the initial estimate is given on input.
`otp`	• "adaptive": adaptive cut-off. • "fixed" : non adaptive cut-off.
`cov`	• "no": no estimate of the covariance matrix of the coefficients is provided on output. • "parametric": a parametric estimate of the covariance matrix of the coefficients is provided (to be used when n is small). • "nonparametric": a nonparametric estimate of the covariance matrix of the coefficients is provided.
`input`	Initial input estimates of location and scale. Required when initial="input". • "Gaussian case" : list(theta=...,sigma=...) initial input estimates. theta: location; sigma: scale. • "logWeibull case" : list(tau=...,v=...) initial input estimates of location (tau) and scale (v).
`control`	Control parameters. For the default values, see the function `TML.noncensored.control`.
`...`	If fastS=TRUE, parameters for `lmrob.S`. See the function `lmrob.control` (from the robustbase package) for the default values.

Value

TML.noncensored returns an object of class "TML". The function summary can be used to obtain or print a summary of the results. The generic extractor functions fitted, residuals and weights can be used to extract various elements of the value returned by TML.noncensored. The function update can be used to update the model.

An object of class "TML" is a list with the following components:

th0	Initial coefficient estimates (S or input).
v0	Initial scale (S or input).
nit0	Reached number of iteration in lmrob.S (available only if fastS is TRUE).
th1	Final coefficient estimates.
v1	Final scale (S or input).
nit1	Number of iterations reached by the IRLS algorithm for the final estimates.
tu, tl	Final cut-off values.
alpha	Estimated proportion of retained observations.
tn	Number of retained observations.
beta	Consistency constant for scale.
weights	Vector of weights (0 for rejected observations, 1 for retained observations).
COV	Covariance matrix of the final estimates (th1[1],...,th1[p],v1) (where p=ncol(X)).
residuals	The residuals, that is response minus fitted values.
fitted.values	The fitted mean values.
call	The matched call.
formula	The formula supplied.
terms	The terms object used.
data	The data argument.

References

Marazzi A., Yohai V. (2004). Adaptively truncated maximum likelihood regression with asymmetric errors. *Journal of Statistical Planning and Inference*, 122, 271-291.

See Also

TML.noncensored.control, TML1.noncensored, TML1.noncensored.control, TML.censored

Examples

```
## Not run:
     data(D243)
     Cost <- D243$Cost                              # Cost (Swiss francs)
     LOS  <- D243$LOS                               # Length of stay (days)
     Adm  <- D243$Typadm; Adm <- (Adm==" Urg")*1    # Type of admission
                                                    # (0=on notification, 1=Emergency)
     Ass  <- D243$Typass; Ass <- (Ass=="P"   )*1    # Type of insurance
```

```
                                                  # (0=usual, 1=private)
Age  <- D243$age                                  # Age (years)
Dst  <- D243$dest;   Dst <- (Dst=="DOMI")*1       # Destination
                                                  # (1=Home, 0=another hospital)
Sex  <- D243$Sexe;   Sex <- (Sex=="M"   )*1       # Sex (1=Male, 0=Female)

# Truncated maximum likelihood regression with Gaussian errors

z    <- TML.noncensored(log(Cost)~log(LOS)+Adm+Ass+Age+Dst+Sex,
         otp="adaptive",control=list(fastS=TRUE))

summary(z)

# Truncated maximum likelihood regression with log-Weibull errors

w    <- TML.noncensored(log(Cost)~log(LOS)+Adm+Ass+Age+Dst+Sex,
        errors="logWeibull",otp="adaptive",control=list(fastS=TRUE))

summary(w)

## End(Not run)
```

`TML.noncensored.control` *Control Parameters for Truncated Maximum Likelihood Regression Without Censored Observations*

Description

Control parameters for `TML.noncensored`. Typically only used internally by `TML.noncensored`, but may be used to construct a control argument. This function provides default values.

Usage

```
TML.noncensored.control(iv = 1, nrep = 0, gam = 0.1, nitmon = FALSE,
                maxit = 200, tol = 1e-04, fastS = FALSE, seed=1313)
```

Arguments

`iv`	• 0: use and do not change the initial estimate of scale. • 1: compute a truncated maximum likelihood estimate of scale.
`nrep`	• Number of subsamples to be used in the computation of the S-estimate. • 0: exhaustive sampling if the observation number is not too large.
`gam`	Relaxation factor for the IRLS algorithm of final estimate. Set 0 < gam <= 1.
`nitmon`	Set to TRUE if iteration monitoring in IRLS algorithm for the final estimate is desired. Default=FALSE.
`maxit`	Maximum number of iterations in IRLS algorithm for the final estimate.

tol	Relative tolerance in IRLS algorithm.
fastS	• "TRUE" : the initial S-estimate is computed using lmrob.S from the **robustbase** package. The control parameters are taken from lmrob.control. • "FALSE" : the initial S-estimate is computed using hysest from the **robeth** package.
seed	Seed for the random number generator in the resampling algorithm for the initial S-estimate.

Value

A list with components named as the arguments.

See Also

TML.noncensored

Examples

```
     ### In the example(TML.noncensored), the control argument can be built
     ### using this function:
## Not run:
     data(D243)
     Cost <- D243$Cost                              # Cost (Swiss francs)
     LOS  <- D243$LOS                               # Length of stay (days)
     Adm  <- D243$Typadm; Adm <- (Adm==" Urg")*1    # Type of admission
                                                    # (0=on notification, 1=Emergency)
     Ass  <- D243$Typass; Ass <- (Ass=="P"   )*1    # Type of insurance
                                                    # (0=usual, 1=private)
     Age  <- D243$age                               # Age (years)
     Dst  <- D243$dest;   Dst <- (Dst=="DOMI")*1    # Destination
                                                    # (1=Home, 0=another hospital)
     Sex  <- D243$Sexe;   Sex <- (Sex=="M"   )*1    # Sex (1=Male, 0=Female)

     # Truncated maximum likelihood regression with Gaussian errors

     ctrol <- TML.noncensored.control(iv=1, nrep=0, gam=0.2, fastS=TRUE, nitmon=FALSE)
     z     <- TML.noncensored(log(Cost)~log(LOS)+Adm+Ass+Age+Dst+Sex, otp="adaptive")
     summary(z)

## End(Not run)
```

TML1.noncensored *Truncated Maximum Likelihood Estimates of Location and Scale*

Description

This functions computes the truncated maximum likelihood estimates of location and scale described in Marazzi and Yohai (2004). It assumes that the error distribution is approximately Gaussian or log-Weibull. The cut-off values for outlier rejection are fixed or adaptive. This function is a simplified version of TML.noncensored for the case without covariates.

Usage

```
TML1.noncensored(y, errors= c("Gaussian", "logWeibull"), cu = NULL,
     initial = c("S", "input"), otp = c("adaptive", "fixed"),
     cov = c("no", "parametric", "nonparametric"), input = NULL,
     control = list(), ...)
```

Arguments

y	Observation vector
errors	• "Gaussian": the error distribution is assumed to be approximately Gaussian. • "logWeibull" : the error distribution is assumed to be approximately log-Weibull.
cu	Preliminary minimal upper cut-off. The default is 2.5 in the Gaussian case and 1.855356 in the log-Weibull case.
initial	• "S" : initial S-estimate. • "input" : the initial estimate is given on input.
otp	• "adaptive": adaptive cut-off. • "fixed" : non adaptive cut-off.
cov	• "no": no estimate of the covariance matrix of the estimates is provided on output. • "parametric": a parametric estimate of the covariance matrix of the location-scale estimates is provided (to be used when n is small). • "nonparametric": a nonparametric estimate of the covariance matrix of the location-scale estimates is provided.
input	Initial input estimates of location and scale. Required when initial="input". • "Gaussian case" : list(theta=...,sigma=...) initial input estimates. theta: location; sigma: scale. • "logWeibull case" : list(tau=...,v=...) initial input estimates of location (tau) and scale (v).
control	Control parameters. For the default values, see the function TML1.noncensored.control.
...	If initial="S", parameters for the computation of the initial S estimates. See the function TML1.noncensored.control.S for the default values.

Value

A list with the following components:

th0	Initial location estimate (S or input).
v0	Initial scale estimate (S or input).
nit0	Reached number of iteration if initial="S"
th1	Final location estimate.
v1	Final scale estimate.

nit1	Reached iteration number in IRLS algorithm for final estimate (only for the log_Weibull case).
tu, tl	Final cut-off values.
alpha	Estimated proportion of retained observations.
tn	Number of retained observations.
beta	Consistency constant for scale.
wi	Vector of weights (0 for rejected observations, 1 for retained observations).
CV0	Covariance matrix of the initial estimates (th0,v0).
CV1	Covariance matrix of the final estimates (th1,v1).

References

Marazzi A., Yohai V. (2004). Adaptively truncated maximum likelihood regression with asymmetric errors. *Journal of Statistical Planning and Inference*, 122, 271-291.

See Also

TML.noncensored, TML1.noncensored.control, TML1.noncensored.control.S

Examples

```
## Not run:
      data(Z243)
      Cost <- Z243$CouTot
      y    <- log(Cost)
      ctrl <- TML1.noncensored.control(iv=1,tol=1e-3)
      z    <- TML1.noncensored(y,errors="logWeibull", initial="S",otp="adaptive",
              cov="no",control=ctrl)

## End(Not run)
```

TML1.noncensored.control

Control Parameters forTruncated Maximum Likelihood Estimation of Location and Scale

---

Description

Auxiliary function for TML1.noncensored. Typically only used internally by TML1.noncensored, but may be used to construct a control argument. This function provides default values.

Usage

```
TML1.noncensored.control(iv = 1, gam = 0.1, maxit = 200, tol = 1e-04)
```

Arguments

iv	• 0: use and do not change the initial estimate of scale. • 1: compute a truncated maximum likelihood estimate of scale.
gam	Relaxation factor for the IRLS algorithm for the final estimate. Set 0 < gam <= 1.
maxit	Maximum number of iterations in the IRLS algorithm for the final estimate.
tol	Relative tolerance in the IRLS algorithm for the final estimate.

Value

A list with components named as the arguments.

See Also

TML1.noncensored

TML1.noncensored.control.S

Control parameters for S-estimate of location and scale

Description

Auxiliary function for TML1.noncensored. Typically only used internally by TML1.noncensored, but may be used to construct a control argument. This function provides default values.

Usage

```
TML1.noncensored.control.S(tlo = 1e-04, mxf = 50, mxs = 50, ntm = 50,
                           tls = 1e-06, h = 100)
```

Arguments

tlo	Relative tolerance in the iterative algorithms.
mxf	Maximum number of iterations in computing the location estimate.
mxs	Maximum number of iterations in computing the scale estimate.
ntm	Parameter used in iteration monitoring. When the number of iterations is a multiple of ntm, the current parameter values are printed.
tls	Tolerance for denominators. If a scale estimate is less than tls, the scale estimate is set equal to tls.
h	The number of subdivisions of the interval (min(yi), max(yi)) used is the computation of the estimate $\lambda^(0)$.

Value

List containing the desired values for each of the control parameters, plus the value Beta0 of β.

Z243 *Sample of 100 hospital stays for medical back problems*

Description

Sample of 100 patients hospitalized for medical back problems in Switzerland

Usage

```
data(Z243)
```

Format

A data frame with 100 observations on the following 14 variables.

`NoAdm` Admission number

`APDRG` DRG: Always 243

`Sex` Gender: 1=Male, 0=Female

`Age` Age in years

`LOS` Length of stay (days)

`CouTot` Total Cost (Swiss francs)

`CsansInv` Cost (Swiss francs)

`Adm` Type of admission (0=on notification, 1=Emergency)

`Ass` Type of insurance (0=usual, 1=private)

`Death` 0=No, 1=Yes

`BBD` A numeric vector

`BBDaggr` A numeric vector

`Charls` A numeric vector

`LOSF` Adjusted length of stay

Examples

```
data(Z243)
```

Index