Package 'pyinit'

October 20, 2021

Type Package

Title Pena-Yohai Initial Estimator for Robust S-Regression

Version 1.1.2

Date 2021-10-19

Encoding UTF-8

Maintainer David Kepplinger <david.kepplinger@gmail.com>

Description Deterministic Pena-Yohai initial estimator for robust S estimators
of regression. The procedure is described in detail in
Pena, D., & Yohai, V. (1999) <doi:10.2307/2670164>.

Imports robustbase

Suggests testthat

License GPL (>= 2)

URL https://github.com/dakep/pyinit

BugReports https://github.com/dakep/pyinit/issues

NeedsCompilation yes

Biarch true

Copyright See the file COPYRIGHTS for copyright details on some of the
functions.

RoxygenNote 7.1.1

Author David Kepplinger [aut, cre],
Matias Salibian-Barrera [aut],
Gabriela Cohen Freue [aut]

Repository CRAN

Date/Publication 2021-10-20 07:20:02 UTC

R topics documented:

`mscale` *Robust M-estimate of Scale*

Description

Compute the M-estimate of scale using the MAD as initial estimate.

Usage

```
mscale(
  x,
  delta = 0.5,
  rho = c("bisquare", "huber", "gauss"),
  cc,
  eps = 1e-08,
  maxit = 200
)
```

Arguments

`x`	numeric vector.
`delta`	desired value for the right-hand side of the M-estimation equation.
`rho`	rho function to use in the M-estimation equation. Valid options are `bisquare`, `huber` and `gauss`.
`cc`	non-negative constant for the chosen rho function. If missing, it will be chosen such that the expected value of the rho function under the normal model is equal to `delta`.
`eps`	threshold for convergence. Defaults to `1e-8`.
`maxit`	maximum number of iterations. Defaults to `200`.

Details

This solves the M-estimation equation given by

$$\sum_{i=1}^{n} \rho(x_i/s_n; cc) = n delta$$

All `NA` values in `x` are removed before calculating the scale.

Value

Numeric vector of length one containing the solution `s_n` to the equation above.

pyinit *PY (Pena-Yohai) initial estimates for S-estimates of regression*

Description

Computes the PY initial estimates for S-estimates of regression.

Usage

```
pyinit(
  x,
  y,
  intercept = TRUE,
  delta = 0.5,
  cc,
  maxit = 10,
  psc_keep,
  resid_keep_method = c("threshold", "proportion"),
  resid_keep_prop,
  resid_keep_thresh,
  eps = 1e-08,
  mscale_maxit = 200,
  mscale_tol = eps,
  mscale_rho_fun = c("bisquare", "huber", "gauss")
)
```

Arguments

`x` a matrix with the data, each observation in a row.

`y` the response vector.

`intercept` logical, should an intercept be included in the model? Defaults to `TRUE`.

`delta, cc` parameters for the M-scale estimator equation. If `cc` is missing it will be set to yield consistency under the Normal model for the given `delta` (right-hand side of the M-scale equation).

`maxit` the maximum number of iterations to perform.

`psc_keep` proportion of observations to keep based on PSCs.

`resid_keep_method`
how to clean the data based on large residuals. If `"threshold"`, all observations with scaled residuals larger than `resid_keep_thresh` will be removed (`resid_keep_thresh` corresponds to the constant C_1 from equation (21) in Pena & Yohai (1999). If `"proportion"`, observations with the largest `resid_keep_prop` residuals will be removed.

`resid_keep_prop, resid_keep_thresh`
see parameter `resid_keep_method` for details.

`eps` the relative tolerance for convergence. Defaults to `1e-8`.

`mscale_maxit` maximum number of iterations allowed for the M-scale algorithm. Defaults to `200`.

`mscale_tol` convergence threshold for the m-scale

`mscale_rho_fun` A string containing the name of the rho function to use for the M-scale. Valid options are `bisquare`, `huber` and `gauss`.

Value

`coefficients` numeric matrix with coefficient vectors in columns. These are regression estimators based on "cleaned" subsets of the data. The M-scales of the corresponding residuals are returned in the entry `objective`. The regression coefficients with smallest estimated residual scale is in the first column, but the others need not be ordered.

`objective` vector of values of the M-scale estimate of the residuals associated with each vector of regression coefficients in the columns of `coefficients`.

References

Pena, D., & Yohai, V.. (1999). A Fast Procedure for Outlier Diagnostics in Large Regression Problems. *Journal of the American Statistical Association*, 94(446), 434-445. <doi:10.2307/2670164>

Examples

```
# generate a simple synthetic data set for a linear regression model
# with true regression coefficients all equal to one "(1, 1, 1, 1, 1)"
set.seed(123)
x <- matrix(rnorm(100*4), 100, 4)
y <- rnorm(100) + rowSums(x) + 1
# add masked outliers
a <- svd(var(x))$v[,4]
x <- rbind(x, t(outer(a, rnorm(20, mean=4, sd=1))))
y <- c(y, rnorm(20, mean=-2, sd=.2))

# these outliers are difficult to find
plot(lm(y~x), ask=FALSE)

# use pyinit to obtain estimated regression coefficients
tmp <- pyinit(x=x, y=y, resid_keep_method='proportion', psc_keep = .5, resid_keep_prop=.5)
# the vector of regression coefficients with smallest residuals scale
# is returned in the first column of the "coefficients" element
tmp$coefficients[,1]
# compare that with the LS estimator on the clean data
coef(lm(y~x, subset=1:100))
# compare it with the LS estimator on the full data
coef(lm(y~x))
```

Index