# Package 'gets'

July 29, 2020

**Type** Package

**Title** General-to-Specific (GETS) Modelling and Indicator Saturation Methods

**Version** 0.24

**Date** 2020-07-28

**Author** Genaro Sucarrat [aut, cre], Felix Pretis [aut], James Reade [aut], Jonas Kurle [ctb], Moritz Schwarz [ctb]

**Maintainer** Genaro Sucarrat <genaro.sucarrat@bi.no>

**Description** Automated General-to-Specific (GETS) modelling of the mean and variance of a regression, and indicator saturation methods for detecting and testing for structural breaks in the mean, see Pretis, Reade and Sucarrat (2018) <doi:10.18637/jss.v086.i03>.

**License** GPL (>= 2)

**Depends** R (>= 3.3.0), zoo, parallel

**Suggests** lgarch, xtable

**BugReports** https://github.com/gsucarrat/gets/issues

**URL** https://CRAN.R-project.org/package=gets, http://www.sucarrat.net/R/gets

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2020-07-28 23:00:09 UTC

## R topics documented:

---

`gets-package` *General-to-Specific Modelling and Indicator Saturation Methods*

---

## Description

Multi-path General-to-Specific (GETS) modelling of the mean and/or variance of a regression, and Indicator Saturation (ISAT) methods for detecting structural breaks in the mean. The mean can be specified as an autoregressive model with covariates (an 'AR-X' model), and the variance can be specified as a dynamic log-variance model with covariates (a 'log-ARCH-X' model). For the statistical details of the model, see Section 4 in Pretis, Reade and Sucarrat (2018).

The main functions of the package are arx, getsm, getsv and isat. The first function, arx, estimates an AR-X model with (optionally) a log-ARCH-X specification on the log-variance. The second function, getsm, undertakes GETS model selection of the mean specification of an arx object. The third function, getsv, undertakes GETS model selection of the log-variance specification of an arx object. The fourth function, isat, undertakes GETS model selection of an indicator saturated mean specification. Extraction functions (mainly S3 methods) are also available, together with additional auxiliary functions used by the main functions.

For an introduction to the package, see Pretis, Reade and Sucarrat (2018): https://www.jstatsoft.org/article/view/v086i03. The package also provides facilities for user-defined GETS and ISAT methods. While this is to some extent available via the main functions, full flexibility is provided by getsFun and blocksFun, see Sucarrat (2019): https://mpra.ub.uni-muenchen.de/96653/.

## Details

| | |
|---|---|
| Package: | gets |
| Type: | Package |
| Version: | 0.24 |
| Date: | 2020-07-28 |
| License: | GPL-2 |

The code originated in relation with G. Sucarrat and A. Escribano (2012). Felix Pretis and James Reade joined for the development of the isat code and related functions. Subsequently, Moritz Schwarz and Jonas Kurle have made various contributions.

## Author(s)

Jonas Kurle, https://www.jonaskurle.com/
Felix Pretis, http://www.felixpretis.org/
James Reade, https://sites.google.com/site/jjamesreade/
Moritz Schwarz, https://www.inet.ox.ac.uk/people/moritz-schwarz/
Genaro Sucarrat, http://www.sucarrat.net/

Maintainer: Genaro Sucarrat

## References

Jurgen A. Doornik, David F. Hendry, and Felix Pretis (2013): 'Step Indicator Saturation', Oxford Economics Discussion Paper, 658.

Felix Pretis, James Reade and Genaro Sucarrat (2018): 'Automated General-to-Specific (GETS) Regression Modeling and Indicator Saturation for Outliers and Structural Breaks'. Journal of Statistical Software 86, Number 3, pp. 1-44. https://www.jstatsoft.org/article/view/v086i03

Carlos Santos, David F. Hendry and Soren Johansen (2007): 'Automatic selection of indicators in a fully saturated regression'. Computational Statistics, vol 23:1, pp.317-335

Genaro Sucarrat (2019): 'User-Specified General-to-Specific and Indicator Saturation Methods'. https://mpra.ub.uni-muenchen.de/96653/

Genaro Sucarrat and Alvaro Escribano (2012): 'Automated Financial Model Selection: General-to-Specific Modelling of the Mean and Volatility Specifications', Oxford Bulletin of Economics and Statistics 74, Issue 5 (October), pp. 716-735. http://doi.org/10.1111/j.1468-0084.2011.00669.x

## See Also

arx, getsm, getsv, isat, getsFun, blocksFun

## Examples

```
##Simulate from an AR(1):
set.seed(123)
y <- arima.sim(list(ar=0.4), 60)

##Estimate an AR(2) with intercept as mean specification
##and a log-ARCH(4) as log-volatility specification:
myModel <- arx(y, mc=TRUE, ar=1:2, arch=1:4)

##GETS modelling of the mean of myModel:
simpleMean <- getsm(myModel)

##GETS modelling of the log-variance of myModel:
simpleVar <- getsv(myModel)

##results:
print(simpleMean)
print(simpleVar)

##step indicator saturation of an iid normal series:
set.seed(123)
y <- rnorm(30)
isat(y)
```

---

# arx *Estimate an AR-X model with log-ARCH-X errors*

---

## Description

Estimation by OLS, two-step OLS if a variance specification is specified: In the first the mean specification (AR-X) is estimated, whereas in the second step the log-variance specification (log-ARCH-X) is estimated.

The AR-X mean specification can contain an intercept, AR-terms, lagged moving averages of the regressand and other conditioning covariates ('X'). The log-variance specification can contain log-ARCH terms, asymmetry or 'leverage' terms, log(EqWMA) where EqWMA is a lagged equally weighted moving average of past squared residuals (a volatility proxy) and other conditioning covariates ('X').

## Usage

```
arx(y, mc=FALSE, ar=NULL, ewma=NULL, mxreg=NULL, vc=FALSE,
  arch=NULL, asym=NULL, log.ewma=NULL, vxreg=NULL, zero.adj=0.1,
  vc.adj=TRUE, vcov.type=c("ordinary", "white", "newey-west"),
  qstat.options=NULL, normality.JarqueB=FALSE, user.estimator=NULL,
  user.diagnostics=NULL, tol=1e-07, LAPACK=FALSE, plot=NULL)
```

## Arguments

| | |
|---|---|
| y | numeric vector, time-series or zoo object. Missing values in the beginning and at the end of the series is allowed, as they are removed with the na.trim command |
| mc | logical. TRUE includes an intercept in the mean specification, whereas FALSE (default) does not |
| ar | either NULL (default) or an integer vector, say, c(2,4) or 1:4. The AR-lags to include in the mean specification. If NULL, then no lags are included |
| ewma | either NULL (default) or a list with arguments sent to the eqwma function. In the latter case a lagged moving average of y is included as a regressor |
| mxreg | either NULL (default) or a numeric vector or matrix, say, a zoo object, of conditioning variables. Note that, if both y and mxreg are zoo objects, then their samples are chosen to match |
| vc | logical. TRUE includes an intercept in the log-variance specification, whereas FALSE (default) does not. If the log-variance specification contains any other item but the log-variance intercept, then vc is set to TRUE |
| arch | either NULL (default) or an integer vector, say, c(1,3) or 2:5. The log-ARCH lags to include in the log-variance specification |
| asym | either NULL (default) or an integer vector, say, c(1) or 1:3. The asymmetry (i.e. 'leverage') terms to include in the log-variance specification |
| log.ewma | either NULL (default) or a vector of the lengths of the volatility proxies, see leqwma |
| vxreg | either NULL (default) or a numeric vector or matrix, say, a zoo object, of conditioning variables. If both y and mxreg are zoo objects, then their samples are chosen to match |
| zero.adj | numeric value between 0 and 1. The quantile adjustment for zero values. The default 0.1 means the zero residuals are replaced by the 10 percent quantile of the absolute residuals before taking the logarithm |
| vc.adj | logical. If TRUE (default), then the log-variance intercept is adjusted by the estimate of E[ln(z^2)], where z is the standardised error. This adjustment is needed for the conditional scale to be equal to the conditional standard deviation. If FALSE, then the log-variance intercept is not adjusted |
| vcov.type | character vector, "ordinary" (default), "white" or "newey-west". If "ordinary", then the ordinary variance-covariance matrix is used for inference. If "white", then the White (1980) heteroscedasticity-robust matrix is used. If "newey-west", then the Newey and West (1987) heteroscedasticity and autocorrelation-robust matrix is used |

`qstat.options` NULL (default) or an integer vector of length two, say, `c(1,1)`. The first value sets the lag-order of the AR diagnostic test, whereas the second value sets the lag-order of the ARCH diagnostic test. If NULL, then the two values of the vector are set automatically

`normality.JarqueB` FALSE (default) or TRUE. If TRUE, then the results of the Jarque and Bera (1980) test for non-normality in the residuals are included in the estimation results.

`user.estimator` NULL (default) or a `list` with one entry, `name`, containing the name of the user-defined estimator. Additional items, if any, are passed on as arguments to the estimator in question

`user.diagnostics` NULL (default) or a `list` with two entries, `name` and `pval`, see the `user.fun` argument in `diagnostics`

`tol` numeric value (`default = 1e-07`). The tolerance for detecting linear dependencies in the columns of the regressors (see `qr` function). Only used if LAPACK is FALSE (default) and `user.estimator` is NULL.

`LAPACK` logical. If TRUE, then use LAPACK. If FALSE (default), then use LINPACK (see `qr` function). Only used if `user.estimator` is NULL.

`plot` NULL or logical. If TRUE, then the fitted values and the residuals are plotted. If NULL (default), then the value set by `options` determines whether a plot is produced or not.

## Details

For an overview of the AR-X model with log-ARCH-X errors, see Pretis, Reade and Sucarrat (2018): https://www.jstatsoft.org/article/view/v086i03.

The arguments `user.estimator` and `user.diagnostics` enables the specification of user-defined estimators and user-defined diagnostics. To this end, the principles of the same arguments in `getsFun` are followed, see its documentation under "Details", and Sucarrat (2019): https://mpra.ub.uni-muenchen.de/96653/.

## Value

A list of class 'arx'

## Author(s)

Genaro Sucarrat, http://www.sucarrat.net/

## References

C. Jarque and A. Bera (1980): 'Efficient Tests for Normality, Homoscedasticity and Serial Independence'. Economics Letters 6, pp. 255-259

Pretis, Felix, Reade, James and Sucarrat, Genaro (2018): 'Automated General-to-Specific (GETS) Regression Modeling and Indicator Saturation for Outliers and Structural Breaks'. Journal of Statistical Software 86, Number 3, pp. 1-44

Genaro Sucarrat (2019): 'User-Specified General-to-Specific and Indicator Saturation Methods'. https://mpra.ub.uni-muenchen.de/96653/

Halbert White (1980): 'A Heteroskedasticity-Consistent Covariance Matrix Estimator and a Direct Test for Heteroskedasticity', Econometrica 48, pp. 817-838

Whitney K. Newey and Kenned D. West (1987): 'A Simple, Positive Semi-Definite, Heteroskedasticity and Autocorrelation Consistent Covariance Matrix', Econometrica 55, pp. 703-708

## See Also

Extraction functions (mostly S3 methods): coef.arx, ES, fitted.arx, plot.arx, print.arx, recursive, residuals.arx, sigma.arx, rsquared, summary.arx, VaR and vcov.arx

Related functions: getsm, getsv, isat

## Examples

```
##Simulate from an AR(1):
set.seed(123)
y <- arima.sim(list(ar=0.4), 70)

##estimate an AR(2) with intercept:
arx(y, mc=TRUE, ar=1:2)

##Simulate four independent Gaussian regressors:
xregs <- matrix(rnorm(4*70), 70, 4)

##estimate an AR(2) with intercept and four conditioning
##regressors in the mean:
arx(y, mc=TRUE, ar=1:2, mxreg=xregs)

##estimate a log-variance specification with a log-ARCH(4)
##structure:
arx(y, arch=1:4)

##estimate a log-variance specification with a log-ARCH(4)
##structure and an asymmetry/leverage term:
arx(y, arch=1:4, asym=1)

##estimate a log-variance specification with a log-ARCH(4)
##structure, an asymmetry or leverage term, a 10-period log(EWMA) as
##volatility proxy, and the log of the squareds of the conditioning
##regressors in the log-variance specification:
arx(y, arch=1:4, asym=1, log.ewma=list(length=10), vxreg=log(xregs^2))

##estimate an AR(2) with intercept and four conditioning regressors
##in the mean, and a log-variance specification with a log-ARCH(4)
##structure, an asymmetry or leverage term, a 10-period log(EWMA) as
##volatility proxy, and the log of the squareds of the conditioning
##regressors in the log-variance specification:
arx(y, mc=TRUE, ar=1:2, mxreg=xregs, arch=1:4, asym=1,
```

```
log.ewma=list(length=10), vxreg=log(xregs^2))
```

---

`as.lm` *Convert to 'lm' object*

---

## Description

Convert 'arx'/'gets'/'isat' object to 'lm' object

## Usage

```
as.lm(object)
```

## Arguments

| | |
|---|---|
| `object` | object of class `arx`, `gets` or `isat` |

## Value

Object of class `lm`

## Author(s)

Moritz Schwarz Genaro Sucarrat

## See Also

`arx`, `gets`, `isat`, `lm`

## Examples

```
##generate data, estimate model of class 'arx':
set.seed(123)
y <- rnorm(30)
arxmod <- arx(y, mc=TRUE, ar=1:3)
as.lm(arxmod)

##from 'gets' to 'lm':
getsmod <- getsm(arxmod, keep=1)
as.lm(getsmod)

##from 'isat' to 'lm':
isatmod <- isat(y)
as.lm(isatmod)
```

`biascorr` *Bias-correction of coefficients following general-to-specific model selection*

## Description

Takes a vector of coefficients (valid for orthogonal variables), their standard errors, the significance level the variables were selected at, and the sample size, to return bias-corrected coefficient estimates to account for the bias induced by model selection.

## Usage

```
biascorr(b, b.se, p.alpha, T)
```

## Arguments

| | |
|---|---|
| `b` | a Kx1 vector of coefficients. |
| `b.se` | a Kx1 vector of standard errors of the coefficients in 'b'. |
| `p.alpha` | numeric value between 0 and 1, the significance level at which selection was conducted. |
| `T` | integer, the sample size of the original model selection regression. |

## Details

The function computes the bias-corrected estimates of coefficients in regression models post general-to-specific model selection using the approach by Hendry and Krolzig (2005). The results are valid for orthogonal regressors only. Bias correction can be applied to the coefficient path in `isat` models where the only additional covariate besides indicators is an intercept - see Pretis (2015).

## Value

Returns a Kx3 matrix, where the first column lists the original coefficients, the second column the one-step corrected coefficients, and the third column the two-step bias-corrected coefficients.

## Author(s)

Felix Pretis, http://www.felixpretis.org/

## References

Hendry, D.F. and Krolzig, H.M. (2005): 'The properties of automatic Gets modelling'. Economic Journal, 115, C32-C61.

Pretis, F. (2015): 'Testing for time-varying predictive accuracy using bias-corrected indicator saturation'. Oxford Department of Economics Discussion Paper.

Pretis, Felix, Reade, James and Sucarrat, Genaro (2018): 'Automated General-to-Specific (GETS) Regression Modeling and Indicator Saturation for Outliers and Structural Breaks'. Journal of Statistical Software 86, Number 3, pp. 1-44

## See Also

isat, coef.gets, plot.gets, isatvar, isattest

## Examples

```
###Bias-correction of the coefficient path of the Nile data
#nile <- as.zoo(Nile)
#isat.nile <- isat(nile, sis=TRUE, iis=FALSE, plot=TRUE, t.pval=0.005)
#var <- isatvar(isat.nile)
#biascorr(b=var$const.path, b.se=var$const.se, p.alpha=0.005, T=length(var$const.path))

##Bias-correction of the coefficient path on artificial data
#set.seed(123)
#d <- matrix(0,100,1)
#d[35:55] <- 1
#e <- rnorm(100, 0, 1)
#y <- d*1  +e

#ys <- isat(y, sis=TRUE, iis=FALSE, t.pval=0.01)
#var <- isatvar(ys)
#biascorr(b=var$const.path, b.se=var$const.se, p.alpha=0.01, T=length(var$const.path))
```

---

# blocksFun *Block-based General-to-Specific (GETS) modelling*

---

## Description

Auxiliary function (i.e. not intended for the average user) that enables block-based GETS-modelling with user-specified estimators, diagnostics and goodness-of-fit criteria.

## Usage

```
blocksFun(y, x, untransformed.residuals=NULL, blocks=NULL,
  no.of.blocks=NULL, max.block.size=30, ratio.threshold=0.8,
  gets.of.union=TRUE, force.invertibility=FALSE,
  user.estimator=list(name="ols"), t.pval=0.001, wald.pval=t.pval,
  do.pet=FALSE, ar.LjungB=NULL, arch.LjungB=NULL, normality.JarqueB=NULL,
  user.diagnostics=NULL, gof.function=list(name="infocrit"),
  gof.method=c("min", "max"), keep=NULL, include.gum=FALSE,
  include.1cut=FALSE, include.empty=FALSE, max.paths=NULL,
  turbo=FALSE, parallel.options=NULL, tol=1e-07, LAPACK=FALSE,
  max.regs=NULL, print.searchinfo=TRUE, alarm=FALSE)
```

## Arguments

| | |
|---|---|
| y | a numeric vector (with no missing values, i.e. no non-numeric 'holes') |
| x | a matrix, or a list of matrices |

untransformed.residuals
: NULL (default) or, when ols is used with method=6 in user.estimator, a numeric vector containing the untransformed residuals

blocks
: NULL (default) or a list of lists with vectors of integers that indicate how blocks should be put together. If NULL, then the block composition is undertaken automatically by an internal algorithm that depends on no.of.blocks, max.block.size and ratio.threshold

no.of.blocks
: NULL (default) or integer. If NULL, then the number of blocks is determined automatically by an internal algorithm

max.block.size
: integer that controls the size of blocks

ratio.threshold
: numeric between 0 and 1 that controls the minimum ratio of variables in each block to total observations

gets.of.union
: logical. If TRUE, then GETS modelling is undertaken of the union of retained variables. Otherwise it is not

force.invertibility
: logical. If TRUE, then the x-matrix is ensured to have full row-rank before it is passed on to getsFun

user.estimator
: list, see getsFun for the details

t.pval
: numeric value between 0 and 1. The significance level used for the two-sided coefficient significance t-tests

wald.pval
: numeric value between 0 and 1. The significance level used for the Parsimonious Encompassing Tests (PETs)

do.pet
: logical. If TRUE, then a Parsimonious Encompassing Test (PET) against the GUM is undertaken at each variable removal for the joint significance of all the deleted regressors along the current GETS path. If FALSE, then a PET is not undertaken at each removal

ar.LjungB
: a two element vector, or NULL. In the former case, the first element contains the AR-order, the second element the significance level. If NULL, then a test for autocorrelation in the residuals is not conducted

arch.LjungB
: a two element vector, or NULL. In the former case, the first element contains the ARCH-order, the second element the significance level. If NULL, then a test for ARCH in the residuals is not conducted

normality.JarqueB
: NULL or a numeric value between 0 and 1. In the latter case, a test for non-normality in the residuals is conducted using a significance level equal to normality.JarqueB. If NULL, then no test for non-normality is conducted

user.diagnostics
: NULL (default) or a list with two entries, name and pval. See getsFun for the details

gof.function
: list. The first item should be named name and contain the name (a character) of the Goodness-of-Fit (GOF) function used. Additional items in the list gof.function are passed on as arguments to the GOF-function. . See getsFun for the details

| | |
|---|---|
| gof.method | character. Determines whether the best Goodness-of-Fit is a minimum (default) or maximum |
| keep | NULL (default), vector of integers or a list of vectors of integers. In the latter case, the number of vectors should be equal to the number of matrices in x |
| include.gum | logical. If TRUE, then the GUM (i.e. the starting model) is included among the terminal models |
| include.1cut | logical. If TRUE, then the 1-cut model is added to the list of terminal models |
| include.empty | logical. If TRUE, then the empty model is added to the list of terminal models |
| max.paths | NULL (default) or integer greater than 0. If NULL, then there is no limit to the number of paths. If integer (e.g. 1), then this integer constitutes the maximum number of paths searched (e.g. a single path) |
| turbo | logical. If TRUE, then (parts of) paths are not searched twice (or more) unnecessarily in each GETS modelling. Setting turbo to TRUE entails a small additional computational costs, but may be outweighed substantially if estimation is slow, or if the number of variables to delete in each path is large |
| parallel.options | NULL or integer that indicates the number of cores/threads to use for parallel computing (implemented w/makeCluster and parLapply) |
| tol | numeric value, the tolerance for detecting linear dependencies in the columns of the variance-covariance matrix when computing the Wald-statistic used in the Parsimonious Encompassing Tests (PETs), see the qr.solve function |
| LAPACK | currently not used |
| max.regs | integer. The maximum number of regressions along a deletion path. Do not alter unless you know what you are doing! |
| print.searchinfo | logical. If TRUE (default), then a print is returned whenever simiplification along a new path is started |
| alarm | logical. If TRUE, then a sound or beep is emitted (in order to alert the user) when the model selection ends |

## Details

blocksFun undertakes block-based GETS modelling by a repeated but structured call to getsFun. For the details of how to user-specify an estimator via user.estimator, diagnostics via user.diagnostics and a goodness-of-fit function via gof.function, see documentation of getsFun under "Details".

The algorithm of blocksFun is similar to that of isat, but more flexible. The main use of blocksFun is the creation of user-specified methods that employs block-based GETS modelling, e.g. indicator saturation techniques.

## Value

A list with the results of the block-based GETS-modelling.

## Author(s)

Genaro Sucarrat, with contributions from Jonas kurle, Felix Pretis and James Reade

## References

F. Pretis, J. Reade and G. Sucarrat (2018): 'Automated General-to-Specific (GETS) Regression Modeling and Indicator Saturation for Outliers and Structural Breaks'. Journal of Statistical Software 86, Number 3, pp. 1-44

G. sucarrat (2019): 'User-Specified General-to-Specific and Indicator Saturation Methods', Munich Personal RePEc Archive: https://mpra.ub.uni-muenchen.de/96653/

## See Also

ols, diagnostics, infocrit and isat

## Examples

```
## more variables than observations:
y <- rnorm(20)
x <- matrix(rnorm(length(y)*40), length(y), 40)
blocksFun(y, x)

## 'x' as list of matrices:
z <- matrix(rnorm(length(y)*40), length(y), 40)
blocksFun(y, list(x,z))

## ensure regressor no. 3 in matrix no. 2 is not removed:
blocksFun(y, list(x,z), keep=list(integer(0), 3))
```

---

# coef.arx *Extraction functions for 'arx' objects*

---

## Description

Extraction functions for objects of class 'arx'

## Usage

```
  ## S3 method for class 'arx'
coef(object, spec=NULL, ...)
  ## S3 method for class 'arx'
fitted(object, spec=NULL, ...)
  ## S3 method for class 'arx'
logLik(object, ...)
  ## S3 method for class 'arx'
plot(x, spec=NULL, col=c("red","blue"),
    lty=c("solid","solid"), lwd=c(1,1), ...)
  ## S3 method for class 'arx'
print(x, signif.stars=TRUE, ...)
  ## S3 method for class 'arx'
```

```
residuals(object, std=FALSE, ...)
  ## S3 method for class 'arx'
sigma(object, ...)
  ## S3 method for class 'arx'
summary(object, ...)
  ## S3 method for class 'arx'
vcov(object, spec=NULL, ...)
```

## Arguments

| | |
|---|---|
| object | an object of class 'arx' |
| x | an object of class 'arx' |
| spec | NULL, "mean", "variance" or, in some instances, "both". When NULL is a valid value, then it is automatically determined whether information pertaining to the mean or variance specification should be returned |
| signif.stars | logical. If TRUE, then p-values are additionally encoded visually, see printCoefmat |
| std | logical. If FALSE (default), then the mean residuals are returned. If TRUE, then the standardised residuals are returned |
| col | colours of actual (default=blue) and fitted (default=red) lines |
| lty | types of actual (default=solid) and fitted (default=solid) lines |
| lwd | widths of actual (default=1) and fitted (default=1) lines |
| ... | additional arguments |

## Value

| | |
|---|---|
| coef: | a numeric vector containing parameter estimates |
| fitted: | a zoo object with fitted values |
| logLik: | log-likelihood (normal density) |
| plot: | a plot of the fitted values and the residuals |
| print: | a print of the estimation results |
| residuals: | a zoo object with the residuals |
| sigma: | the regression standard error ('SE of regression') |
| summary: | a print of the items in the arx object |
| vcov: | variance-covariance matrix |

## Author(s)

Felix Pretis, http://www.felixpretis.org/
James Reade, https://sites.google.com/site/jjamesreade/
Genaro Sucarrat, http://www.sucarrat.net/

## See Also

arx

## Examples

```
##simulate from an AR(1):
set.seed(123)
y <- arima.sim(list(ar=0.4), 40)

##simulate four independent Gaussian regressors:
xregs <- matrix(rnorm(4*40), 40, 4)

##estimate an 'arx' model: An AR(2) with intercept and four conditioning
##regressors in the mean, and log-ARCH(3) in the variance:
mymod <- arx(y, mc=TRUE, ar=1:2, mxreg=xregs, arch=1:3)

##print results:
print(mymod)

##plot the fitted vs. actual values, and the residuals:
plot(mymod)

##print the entries of object 'mymod':
summary(mymod)

##extract coefficient estimates (automatically determined):
coef(mymod)

##extract mean coefficients only:
coef(mymod, spec="mean")

##extract log-variance coefficients only:
coef(mymod, spec="variance")

##extract all coefficient estimates:
coef(mymod, spec="both")

##extract regression standard error:
sigma(mymod)

##extract log-likelihood:
logLik(mymod)

##extract variance-covariance matrix of mean equation:
vcov(mymod)

##extract variance-covariance matrix of log-variance equation:
vcov(mymod, spec="variance")

##extract and plot the fitted mean values (automatically determined):
mfit <- fitted(mymod)
plot(mfit)

##extract and plot the fitted variance values:
vfit <- fitted(mymod, spec="variance")
plot(vfit)
```

```
##extract and plot both the fitted mean and variance values:
vfit <- fitted(mymod, spec="both")
plot(vfit)

##extract and plot the fitted mean values:
vfit <- fitted(mymod, spec="mean")
plot(vfit)

##extract and plot residuals:
epshat <- residuals(mymod)
plot(epshat)

##extract and plot standardised residuals:
zhat <- residuals(mymod, std=TRUE)
plot(zhat)
```

---

coef.gets *Extraction functions for 'gets' objects*

---

## Description

Extraction functions for objects of class 'gets'

## Usage

```
  ## S3 method for class 'gets'
coef(object, spec=NULL, ...)
  ## S3 method for class 'gets'
fitted(object, spec=NULL, ...)
  ## S3 method for class 'gets'
logLik(object, ...)
  ## S3 method for class 'gets'
plot(x, spec=NULL, col=c("red","blue"),
    lty=c("solid","solid"), lwd=c(1,1), ...)
  ## S3 method for class 'gets'
predict(object, spec=NULL, n.ahead=12, newmxreg=NULL,
    newvxreg=NULL, newindex=NULL, n.sim=5000, innov=NULL, probs=NULL,
    ci.levels=NULL, quantile.type=7, return=TRUE, verbose=FALSE, plot=NULL,
    plot.options=list(), ...)
  ## S3 method for class 'gets'
print(x, signif.stars=TRUE, ...)
  ## S3 method for class 'gets'
residuals(object, std=NULL, ...)
  ## S3 method for class 'gets'
sigma(object, ...)
  ## S3 method for class 'gets'
```

```
summary(object, ...)
  ## S3 method for class 'gets'
vcov(object, spec=NULL, ...)
```

## Arguments

| | |
|---|---|
| object | an object of class 'gets' |
| x | an object of class 'gets' |
| spec | NULL, "mean", "variance" or, in some instances, "both". When NULL is a valid value, then it is automatically determined whether information pertaining to the mean or variance specification should be returned |
| signif.stars | logical. If TRUE, then p-values are additionally encoded visually, see printCoefmat |
| std | logical. If FALSE (default), then the mean residuals are returned. If TRUE, then the standardised residuals are returned |
| n.ahead | integer that determines how many steps ahead predictions should be generated (the default is 12) |
| newmxreg | a matrix of n.ahead rows and NCOL(mxreg) columns with the out-of-sample values of the mxreg regressors |
| newvxreg | a matrix of n.ahead rows and NCOL(vxreg) columns with the out-of-sample values of the vxreg regressors |
| newindex | NULL (default) or the date-index for the zoo object returned by predict.arx. If NULL, then the function uses the in-sample index to generate the out-of-sample index |
| n.sim | integer, the number of replications used for the generation of the forecasts |
| innov | NULL (default) or a vector of length n.ahead * n.sim containing the standardised errors (that is, zero mean and unit variance) used for the forecast simulations. If NULL, then a classica bootstrap procedure is used to draw from the standardised in-sample residuals |
| probs | NULL (default) or a vector with the quantile-levels (values strictly between 0 and 1) of the forecast distribution. If NULL, then no quantiles are returned unless ci.levels is non-NULL |
| ci.levels | NULL (default) or a vector with the confidence levels (expressed as values strictly between 0 and 1) of the forecast distribution. The upper and lower values of the confidence interval(s) are returned as quantiles |
| quantile.type | an integer between 1 and 9 that selects which algorithm to be used in computing the quantiles, see the argument type in quantile |
| return | logical. If TRUE (default), then the out-of-sample predictions are returned. The value FALSE, which does not return the predictions, may be of interest if only a prediction plot is of interest |
| verbose | logical with default FALSE. If TRUE, then additional information (typically the quantiles and/or the simulated series) used in the generation of forecasts is returned. If FALSE, then only the forecasts are returned |
| plot | NULL (default) or logical. If NULL, then the value set by options$plot (see options) determines whether a plot is produced or not. If TRUE, then the out-of-sample forecasts are plotted. |

| | |
|---|---|
| plot.options | a list of options related to the plotting of forecasts, see 'Details' |
| col | colours of fitted (default=red) and actual (default=blue) lines |
| lty | types of fitted (default=solid) and actual (default=solid) lines |
| lwd | widths of fitted (default=1) and actual (default=1) lines |
| ... | additional arguments |

## Details

The plot.options argument is a list that controls the prediction plot, see 'Details' in predict.arx

## Value

| | |
|---|---|
| coef: | a numeric vector containing parameter estimates |
| fitted: | a zoo object with fitted values |
| logLik: | a numeric, the log-likelihood (normal density) |
| plot: | a plot of the fitted values and the residuals |
| predict: | a vector of class zoo containing the out-of-sample forecasts, or a matrix of class zoo containing the out-of-sample forecasts together with prediction-quantiles, or - if return=FALSE - NULL |
| print: | a print of the estimation results |
| residuals: | a zoo object with the residuals |
| sigma: | the regression standard error ('SE of regression') |
| summary: | a print of the items in the gets object |
| vcov: | a variance-covariance matrix |

## Author(s)

Felix Pretis, http://www.felixpretis.org/
James Reade, https://sites.google.com/site/jjamesreade/
Genaro Sucarrat, http://www.sucarrat.net/

## See Also

getsm, getsv, isat

## Examples

```
##Simulate from an AR(1):
set.seed(123)
y <- arima.sim(list(ar=0.4), 100)

##Simulate four independent Gaussian regressors:
xregs <- matrix(rnorm(4*100), 100, 4)

##estimate an AR(2) with intercept and four conditioning
##regressors in the mean, and a log-ARCH(3) in the variance:
```

```
mymod <- arx(y, mc=TRUE, ar=1:2, mxreg=xregs, arch=1:3)

##General-to-Specific (GETS) model selection of the mean:
meanmod <- getsm(mymod)

##General-to-Specific (GETS) model selection of the variance:
varmod <- getsv(mymod)

##print results:
print(meanmod)
print(varmod)

##plot the fitted vs. actual values, and the residuals:
plot(meanmod)
plot(varmod)

##generate and plot predictions of the mean:
predict(meanmod, plot=TRUE)

##print the entries of object 'gets':
summary(meanmod)
summary(varmod)

##extract coefficients of the simplified (specific) model:
coef(meanmod) #mean spec
coef(varmod) #variance spec

##extract log-likelihood:
logLik(mymod)

##extract coefficient-covariance matrix of simplified
##(specific) model:
vcov(meanmod) #mean spec
vcov(varmod) #variance spec

##extract and plot the fitted values:
mfit <- fitted(meanmod) #mean fit
plot(mfit)
vfit <- fitted(varmod) #variance fit
plot(vfit)

##extract and plot residuals:
epshat <- residuals(meanmod)
plot(epshat)

##extract and plot standardised residuals:
zhat <- residuals(varmod)
plot(zhat)
```

coef.isat *Extraction functions for 'isat' objects*

## Description

Extraction functions for objects of class 'isat'

## Usage

```
  ## S3 method for class 'isat'
coef(object, ...)
  ## S3 method for class 'isat'
fitted(object, ...)
  ## S3 method for class 'isat'
logLik(object, ...)
  ## S3 method for class 'isat'
plot(x, col=c("red","blue"), lty=c("solid","solid"),
    lwd=c(1,1), coef.path=TRUE, ...)
  ## S3 method for class 'isat'
predict(object, n.ahead=12, newmxreg=NULL, newindex=NULL,
    n.sim=2000, probs=NULL, ci.levels=NULL, quantile.type=7,
    return=TRUE, verbose=FALSE, plot=NULL, plot.options=list(), ...)
  ## S3 method for class 'isat'
print(x, ...)
  ## S3 method for class 'isat'
residuals(object, std=FALSE, ...)
  ## S3 method for class 'isat'
sigma(object, ...)
  ## S3 method for class 'isat'
summary(object, ...)
  ## S3 method for class 'isat'
vcov(object, ...)
```

## Arguments

| | |
|---|---|
| object | an object of class 'isat' |
| x | an object of class 'isat' |
| std | logical. If FALSE (default), then the mean residuals are returned. If TRUE, then the standardised residuals are returned |
| n.ahead | integer that determines how many steps ahead predictions should be generated (the default is 12) |
| newmxreg | a matrix of n.ahead rows and NCOL(mxreg) columns with the out-of-sample values of the mxreg regressors |
| newindex | NULL (default) or the date-index for the zoo object returned by predict.arx. If NULL, then the function uses the in-sample index to generate the out-of-sample index |
| n.sim | integer, the number of replications used for the generation of the forecasts |

| | |
|---|---|
| probs | NULL (default) or a vector with the quantile-levels (values strictly between 0 and 1) of the forecast distribution. If NULL, then no quantiles are returned unless ci.levels is non-NULL |
| ci.levels | NULL (default) or a vector with the confidence levels (expressed as values strictly between 0 and 1) of the forecast distribution. The upper and lower values of the confidence interval(s) are returned as quantiles |
| quantile.type | an integer between 1 and 9 that selects which algorithm to be used in computing the quantiles, see the argument type in quantile |
| return | logical. If TRUE (default), then the out-of-sample predictions are returned. The value FALSE, which does not return the predictions, may be of interest if only a prediction plot is of interest |
| verbose | logical with default FALSE. If TRUE, then additional information (typically the quantiles and/or the simulated series) used in the generation of forecasts is returned. If FALSE, then only the forecasts are returned |
| plot | NULL (default) or logical. If NULL, then the value set by options$plot (see options) determines whether a plot is produced or not. If TRUE, then the out-of-sample forecasts are plotted. |
| plot.options | a list of options related to the plotting of forecasts, see 'Details' |
| col | colours of fitted (default=red) and actual (default=blue) lines |
| lty | types of fitted (default=solid) and actual (default=solid) lines |
| lwd | widths of fitted (default=1) and actual (default=1) lines |
| coef.path | logical. Only applicable if there are retained indicators after the application of isat |
| ... | additional arguments |

## Details

The plot.options argument is a list that controls the prediction plot, see 'Details' in predict.arx

## Value

| | |
|---|---|
| coef: | numeric vector containing parameter estimates |
| fitted: | a zoo object with fitted values |
| logLik: | a numeric, the log-likelihood (normal density) |
| plot: | plot of the fitted values and the residuals |
| predict: | a vector of class zoo containing the out-of-sample forecasts, or a matrix of class zoo containing the out-of-sample forecasts together with prediction-quantiles, or - if return=FALSE - NULL |
| print: | a print of the estimation results |
| residuals: | a zoo object with the residuals |
| sigma: | the regression standard error ('SE of regression') |
| summary: | a print of the items in the isat object |
| vcov: | variance-covariance matrix |

## Author(s)

Felix Pretis, http://www.felixpretis.org/
James Reade, https://sites.google.com/site/jjamesreade/
Genaro Sucarrat, http://www.sucarrat.net/

## See Also

paths, terminals, coef.gets, getsm, arx

## Examples

```
##step indicator saturation:
set.seed(123)
y <- rnorm(30)
isatmod <- isat(y)

##print results:
print(isatmod)

##plot the fitted vs. actual values, and the residuals:
plot(isatmod)

##print the entries of object 'isatmod':
summary(isatmod)

##extract coefficients of the simplified (specific) model:
coef(isatmod)

##extract log-likelihood:
logLik(isatmod)

##extract the coefficient-covariance matrix of simplified
##(specific) model:
vcov(isatmod)

##extract and plot the fitted values:
mfit <- fitted(isatmod)
plot(mfit)

##extract and plot (mean) residuals:
epshat <- residuals(isatmod)
plot(epshat)

##extract and plot standardised residuals:
zhat <- residuals(isatmod, std=TRUE)
plot(zhat)

##generate forecasts of the simplified (specific) model:
predict(isatmod, newmxreg=matrix(1,12,1), plot=TRUE)
```

diagnostics *Diagnostics tests*

## Description

Auxiliary function (i.e. not intended for the average user) called by the arx, getsm, getsv, isat, getsFun and blocksFun functions. The diagnostics function undertakes tests for autocorrelation, ARCH and non-normality in a residual series, and user-defined diagnostics provided via the user.fun argument (see details). The autocorrelation and ARCH tests are conducted as Ljung and Box (1979) tests for autocorrelation in the residuals and squared residuals, respectively, whereas the test for non-normality is that of Jarque and Bera (1980).

## Usage

```
diagnostics(x, ar.LjungB=c(1, 0.025), arch.LjungB=c(1, 0.025),
  normality.JarqueB=NULL, verbose=TRUE, user.fun=NULL, ...)
```

## Arguments

| | |
|---|---|
| x | a list, for example the estimation result of ols. The tests for serial correlation, ARCH and normality look for an entry in the list named std.residuals or residuals |
| ar.LjungB | a two element vector or NULL. In the former case, the first element contains the AR-order, the second element the significance level. If NULL, then a test for autocorrelation is not conducted |
| arch.LjungB | a two element vector or NULL. In the former case, the first element contains the ARCH-order, the second element the significance level. If NULL, then a test for ARCH is not conducted |
| normality.JarqueB | NULL (the default) or a value between 0 and 1. In the latter case, a test for non-normality is conducted using a significance level equal to normality.JarqueB. If NULL, then no test for non-normality is conducted |
| verbose | logical. If TRUE, then a data.frame with the results of the diagnostics is returned. If FALSE, then the return-value is a logical that indicates whether the model passes the diagnostics (TRUE if it does, otherwise FALSE) |
| user.fun | NULL or a list with at least one entry, name (must be of class character), which should contain the name of the user-defined function. See details |
| ... | further arguments (ignored) to accommodate deleted arguments from past versions of the functions |

## Details

The argument user.fun enables the user to specify additional diagnostics. To do this, the argument should be a list with at least one entry, name (of class character), that contains the name of the user-defined function. The call to this function is executed with do.call, whose default value on

envir is parent.frame(). Usually, this will be the global environment (.GlobalEnv), but it can be changed by adding an entry named envir to the list that indicates where the user-defined function resides. If the verbose argument is set to FALSE, then an entry named pval must be provided. This entry should contain the chosen significance level or levels, i.e. either a scalar or a vector of length equal to the number of *p*-values returned by the user-defined diagnostics function (see examples). Additional entries in the list are passed on as arguments to the user-defined function.

The user-defined function should refer to the named items of the estimation result x (see examples), and the value returned by the user-defined function should be a matrix of dimension *m* x 3. Here, *m* is the number of diagnostic tests performed by the user-defined function. For example, if only a single test is performed, then *m* = 1 and so the returned value should be a 1 x 3 matrix (or a vector of length 3). The three columns of the *m* x 3 matrix should contain, in the following order, 1) the value(s) of the test-statistic(s) (or NA), 2) the degree(s) of freedom(s) (or NA) of the tests, and 3) the *p*-value(s) of the test(s). When checking whether the model passes the diagnostics or not, the *p*-value(s) is(are) checked against the value(s) in the entry named pval in the list provided to user.fun.

## Value

If verbose=TRUE:

a data.frame that contains the diagnostics results

If verbose=FALSE:

a logical indicating whether the residuals and/or model passes ALL the diagnostics (TRUE if it does, FALSE otherwise)

## Author(s)

Genaro Sucarrat, http://www.sucarrat.net/

## References

C. Jarque and A. Bera (1980): 'Efficient Tests for Normality, Homoscedasticity and Serial Independence'. Economics Letters 6, pp. 255-259

G. Ljung and G. Box (1979): 'On a Measure of Lack of Fit in Time Series Models'. Biometrika 66, pp. 265-270

## See Also

arx, getsm, getsv, isat, getsFun, blocksFun

## Examples

```
##generate some data:
set.seed(123)
vY <- rnorm(20) #the regressand
mX <- matrix(rnorm(3*20), 20, 3) #the regressors
est <- ols(vY,mX)

##return a data-frame with autocorrelation and ARCH diagnostics (default),
##and check whether they pass (the default p-value is 0.025):
diagnostics(est)
```

```
diagnostics(est, verbose=FALSE)

##add the Jarque-Bera normality test to the diagnostics (w/p-value=0.05):
diagnostics(est, normality.JarqueB=0.05)
diagnostics(est, normality.JarqueB=0.05, verbose=FALSE)

##user-defined Shapiro-Wilks test for non-normality of the residuals:
SWtest <- function(x, ...){
  tmp <- shapiro.test(x$residuals) #do test on est$residuals
  return( c(tmp$statistic, NA, tmp$p.value) )
}
diagnostics(est, user.fun=list(name="SWtest", pval=0.05))
diagnostics(est, user.fun=list(name="SWtest", pval=0.05), verbose=FALSE)
```

---

dropvar *Drop variable*

---

## Description

Drops columns in a matrix to avoid perfect multicollinearity.

## Usage

```
dropvar(x, tol=1e-07, LAPACK=FALSE, silent=FALSE)
```

## Arguments

| | |
|---|---|
| x | a matrix, possibly less than full column rank. |
| tol | numeric value. The tolerance for detecting linear dependencies among regressors, see qr function. Only used if LAPACK is FALSE |
| LAPACK | logical, TRUE or FALSE (default). If true use LAPACK otherwise use LINPACK, see qr function |
| silent | logical, TRUE (default) or FALSE. Whether to print a notification whenever a regressor is removed |

## Details

Original function drop.coef developed by Rune Haubo B. Christensen in package ordinal, https://cran.r-project.org/package=ordinal.

## Value

a matrix whose regressors linearly independent

## Author(s)

Rune Haubo B. Christensen, with modifications by Genaro Sucarrat, http://www.sucarrat.net/

## References

Rune H.B. Christensen (2014): 'ordinal: Regression Models for Ordinal Data'. https://cran.r-project.org/package=ordinal

## See Also

isat

## Examples

```
set.seed(1)
x <- matrix(rnorm(20), 5)
dropvar(x) #full rank, none are dropped

x[,4] <- x[,1]*2
dropvar(x) #less than full rank, last column dropped
```

---

eqwma *Equally Weighted Moving Average (EqWMA) of the pth. exponentiated values*

---

## Description

The function eqwma returns an Equally Weighted Moving Average (EqWMA) of the pth. exponentiated values lagged k times (the default of k is 1). Optionally, the absolute values are computed before averaging if abs=TRUE, and the natural log of the values is returned if log=TRUE. The function leqwma is a wrapper to eqwma with abs=TRUE and log=TRUE.

If x is financial return (possibly mean-corrected) and p=2, then this gives the socalled 'historical' model, also known as an integrated ARCH model where the ARCH coefficients all have the same value with sum equal to one. In the log-variance specification the lag of log(EqWMA) is thus a financial volatility proxy. It may be an imperfect proxy compared with high-frequency data (which can also be included as regressors), but - in contrast to high-frequency data - is always available and easy to compute.

## Usage

```
eqwma(x, length=5, k=1, p=1, abs=FALSE, log=FALSE,  as.vector=FALSE,
  lag=NULL, start=NULL)
leqwma(x, length=5, k=1, p=2, as.vector=FALSE, lag=NULL, start=NULL)
```

## Arguments

| | |
|---|---|
| x | numeric vector, time-series or zoo object |
| length | integer or vector of integers each equal to or greater than 1. The length or lengths of the moving window or windows of averages |
| k | integer that determines how many periods the term(s) should be lagged. If 0 (or smaller), then the moving averages are not lagged |

| | |
|---|---|
| p | numeric value. The exponent p in x^p when abs=FALSE, and in abs(x)^p when abs=TRUE |
| log | logical with default FALSE. If TRUE, then the logarithm of the moving average is returned |
| abs | logical with default FALSE. If TRUE, then x is transformed to absolute values before x is exponentiated |
| as.vector | logical with default FALSE. If TRUE, and if length(length)==1, then the result is returned as a vector. Otherwise the returned value is always a matrix |
| lag | deprecated |
| start | deprecated |

## Details

The intended primary use of eqwma is to construct mixed frequency regressors for the mean specification of an arx model.

The intended primary use of leqwma is to construct volatility proxies for the log-variance specification in an arx model. In the latter case, the default is the lagged log of an equally weighted moving average of the squared residuals, where each average is made up of m observations. This is equivalent to an integrated ARCH(p) model where the p coefficients are all equal. For further details on the use of log(EqWMA) as a volatility proxy, see Sucarrat and Escribano (2012).

## Value

numeric matrix, vector or zoo object

## Author(s)

Genaro Sucarrat, http://www.sucarrat.net/

## References

Genaro Sucarrat and Alvaro Escribano (2012): 'Automated Financial Model Selection: General-to-Specific Modelling of the Mean and Volatility Specifications', Oxford Bulletin of Economics and Statistics 74, Issue no. 5 (October), pp. 716-735

Pretis, Felix, Reade, James and Sucarrat, Genaro (2018): 'Automated General-to-Specific (GETS) Regression Modeling and Indicator Saturation for Outliers and Structural Breaks'. Journal of Statistical Software 86, Number 3, pp. 1-44

## See Also

zoo, arx, getsm, getsv

## Examples

```
##generate an iid normal series:
set.seed(123)
x <- rnorm(100)
```

```
##compute lag of EqWMA(20) for x^2:
eqwma(x, p=2)

##compute lag of EqWMA(5) and lag of EqWMA(10) for x:
eqwma(x, length=c(5,10))

##compute lag of log(EqWMA(20)) for x^2:
leqwma(x)

#compute lag of log(EqWMA(5)) and lag of log(EqWMA(8))
#for abs(x)^2:
leqwma(x, length=c(4,8))
```

---

ES *Conditional Value-at-Risk (VaR) and Expected Shortfall (ES)*

---

## Description

Extract the in-sample conditional Value-at-Risk, or the in-sample conditional Expected Shortfall for the chosen risk level(s).

## Usage

```
ES(object, level=0.99, type=7, ...)
VaR(object, level=0.99, type=7, ...)
```

## Arguments

| | |
|---|---|
| `object` | an `arx` or `gets` object |
| `level` | the risk level(s), must be between 0 and 1 |
| `type` | the method used to compute the empirical quantiles of the standardised residuals |
| `...` | arguments passed on (currently not used) |

## Value

A vector or matrix containing either the conditional Value-at-Risk (VaR) or the conditional Expected Shortfall (ES) for the chosen risk level(s).

## Author(s)

Genaro Sucarrat, http://www.sucarrat.net/

## See Also

`arx`, `getsm`, `getsv`

### Examples

```
##generate random variates, estimate model:
y <- rnorm(50)
mymodel <- arx(y, arch=1)

##extract 99% expected shortfall:
ES(mymodel)

##extract 99%, 95% and 90% expected shortfalls:
ES(mymodel, level=c(0.99, 0.95, 0.9))

##extract 99% value-at-risk:
VaR(mymodel)

##extract 99%, 95% and 90% values-at-risk:
VaR(mymodel, level=c(0.99, 0.95, 0.9))
```

---

## eviews *Exporting results to EViews and STATA*

---

### Description

Functions that facilitate the export of results to the commercial econometric softwares EViews and STATA, respectively.

### Usage

```
eviews(object, file=NULL, print=TRUE, return=FALSE)
stata(object, file=NULL, print=TRUE, return=FALSE)
```

### Arguments

| | |
|---|---|
| object | an arx, gets or isat object |
| file | filename, i.e. the destination of the exported data |
| print | logical. If TRUE, then the estimation code in EViews (or STATA) is printed |
| return | logical. If TRUE, then a list is returned |

### Value

Either printed text or a list (if return=TRUE)

### Author(s)

Genaro Sucarrat, http://www.sucarrat.net/

### See Also

arx, getsm, getsv, isat

## Examples

```
##simulate random variates, estimate model:
y <- rnorm(30)
mX <- matrix(rnorm(30*2), 30, 2)
mymod <- arx(y, mc=TRUE, mxreg=mX)

##print EViews code:
eviews(mymod)

##print Stata code:
stata(mymod)
```

---

# gets *General-to-Specific (GETS) Modelling*

---

## Description

Generic function that enables new GETS and ISAT methods for new classes

## Usage

```
gets(x, ...)

##wrapper to getsm() and getsv():
## S3 method for class 'arx'
gets(x, spec=NULL, ...)
```

## Arguments

| | |
|---|---|
| x | an object of class 'arx' used to select a method |
| ... | further arguments passed to or from other methods |
| spec | NULL (default), "mean" or "variance". If "mean", then getsm is called. If "variance", then getsv is called. If NULL, then it is automatically determined whether GETS-modelling of the mean or log-variance specification should be undertaken. |

## Details

gets.arx is a convenience wrapper to getsm and getsv.

## Author(s)

Genaro Sucarrat, http://www.sucarrat.net/

## See Also

getsm, getsv, getsFun

getsFun *General-to-Specific (GETS) modelling function*

## Description

Auxiliary function (i.e. not intended for the average user) that enables fast and efficient GETS-modelling with user-specified estimators and models, and user-specified diagnostics and goodness-of-fit criteria. The function is called by and relied upon by getsm, getsv, isat and blocksFun.

## Usage

```
getsFun(y, x, untransformed.residuals=NULL,
  user.estimator=list(name="ols"), gum.result=NULL, t.pval=0.05,
  wald.pval=t.pval, do.pet=TRUE, ar.LjungB=NULL, arch.LjungB=NULL,
  normality.JarqueB=NULL, user.diagnostics=NULL,
  gof.function=list(name="infocrit"), gof.method=c("min", "max"),
  keep=NULL, include.gum=FALSE, include.1cut=FALSE,
  include.empty=FALSE, max.paths=NULL, turbo=FALSE, tol=1e-07,
  LAPACK=FALSE, max.regs=NULL, print.searchinfo=TRUE, alarm=FALSE)
```

## Arguments

| | |
|---|---|
| y | a numeric vector (with no missing values, i.e. no non-numeric 'holes') |
| x | a matrix or NULL |
| untransformed.residuals | NULL (default) or, when ols is used with method=6 in user.estimator, a numeric vector containing the untransformed residuals |
| user.estimator | a list. The first item should be named name and contain the name (a character) of the estimation function (the default is "ols"). Additional items, if any, in the list user.estimator are passed on as arguments to the estimator in question. Optionally, the list can also contain an item named envir, a character, which indicates the environment in which the user-specified estimator resides. The value returned by the user-specified estimator should be a list, see details |
| gum.result | a list with the estimation results of the General Unrestricted Model (GUM), or NULL (default). If the estimation results of the GUM are already available, then re-estimation of the GUM is skipped if the estimation results are provided via this argument |
| t.pval | numeric value between 0 and 1. The significance level used for the two-sided regressor significance t-tests |
| wald.pval | numeric value between 0 and 1. The significance level used for the Parsimonious Encompassing Tests (PETs) |
| do.pet | logical. If TRUE (default), then a Parsimonious Encompassing Test (PET) against the GUM is undertaken at each regressor removal for the joint significance of all the deleted regressors along the current path. If FALSE, then a PET is not undertaken at each regressor removal |

ar.LjungB — a two element vector or NULL (default). In the former case, the first element contains the AR-order, the second element the significance level. If NULL, then a test for autocorrelation is not conducted

arch.LjungB — a two element vector or NULL (default). In the former case, the first element contains the ARCH-order, the second element the significance level. If NULL, then a test for ARCH is not conducted

normality.JarqueB — NULL or a numeric value between 0 and 1. In the latter case, a test for non-normality is conducted using a significance level equal to normality.JarqueB. If NULL, then no test for non-normality is conducted

user.diagnostics — NULL (default) or a list with two entries, name and pval. The first item (name) should contain the name of the user-defined function, and must be of class character. The second item should contain the chosen significance level or levels, i.e. either a scalar or a vector of length equal to the number of p-values returned by the user-defined diagnostics function, see details. Optionally, the list user.diagnostics can also contain a third item named envir, a character, which indicates the environment in which the user-defined function resides

gof.function — a list. The first item should be named name and contain the name (a character) of the Goodness-of-Fit (GOF) function used. Additional items in the list gof.function are passed on as arguments to the GOF-function. The value returned by the GOF-function should be a numeric value (of length 1). Optionally, the list gof.function can also contain an item named envir, a character, which indicates the environment in which the user-defined function resides

gof.method — a character. Determines whether the best Goodness-of-Fit is a minimum or maximum

keep — NULL or an integer vector that indicates which regressors to be excluded from removal in the search

include.gum — logical. If TRUE, then the GUM (i.e. the starting model) is included among the terminal models. If FALSE (default), then the GUM is not included

include.1cut — logical. If TRUE, then the 1-cut model is added to the list of terminal models. If FALSE (default), then the 1-cut is not added, unless it is a terminal model in one of the paths

include.empty — logical. If TRUE, then the empty model is added to the list of terminal models. If FALSE (default), then the empty model is not added, unless it is a terminal model in one of the paths

max.paths — NULL (default) or an integer greater than 0. If NULL, then there is no limit to the number of paths. If an integer (e.g. 1), then this integer constitutes the maximum number of paths searched (e.g. a single path)

turbo — logical. If TRUE, then (parts of) paths are not searched twice (or more) unnecessarily, thus yielding a significant potential for speed-gain. However, the checking of whether the search has arrived at a point it has already been comes with a slight computational overhead. Accordingly, if turbo=TRUE, then the total search time might in fact be higher than if turbo=FALSE. This happens if estimation is very fast, say, less than quarter of a second. Hence the default is FALSE

`tol` numeric value (`default = 1e-07`). The tolerance for detecting linear dependencies in the columns of the variance-covariance matrix when computing the Wald-statistic used in the Parsimonious Encompassing Tests (PETs), see the `qr.solve` function

`LAPACK` currently not used

`max.regs` `integer`. The maximum number of regressions along a deletion path. Do not alter unless you know what you are doing!

`print.searchinfo`
`logical`. If `TRUE` (default), then a print is returned whenever simiplification along a new path is started

`alarm` `logical`. If `TRUE`, then a sound or beep is emitted (in order to alert the user) when the model selection ends

## Details

The value returned by the estimator specified in `user.estimator` should be a `list` containing at least six items: "coefficients", "df", "vcov", "logl", "n" and "k". The item "coefficients" should be a vector of length `NCOL(x)` containing the estimated coefficients. The item named "df" is used to compute the *p*-values associated with the *t*-statistics, i.e. coef/std.err. The item named "vcov" contains the (symmetric) coefficient-covariance matrix of the estimated coefficients. The items "logl" (the log-likelihood), "n" (the number of observations) and "k" (the number of estimated parameters; not necessarily equal to the number of coefficients) are used to compute the information criterion. Finally, the estimator MUST be able to handle empty regressor-matrices (i.e. `is.null(x)=TRUE` or `NCOL(x)=0`). In this case, then the first three items (i.e. "coefficients", "df" and "vcov") can - and should - be `NULL`.

The argument `user.estimator` enables the user to specify an estimator that differs from the default (`ols`). To do this, the argument should be a list with at least one entry, name (of class character), that contains the name of the user-defined function. The call to this function is executed with `do.call`, whose default value on `envir` is `parent.frame()`. Usually, this will be the global environment (`.GlobalEnv`), but it can be changed by adding an entry named `envir` to the list that indicates where the user-defined function resides.

The argument `user.diagnostics` enables the user to specify additional - or alternative - diagnostics, see `diagnostics`.

The argument `gof.function` enables the user to specify a goodness-of-fit function that differs from the default (`infocrit`). The principles to follow are the same as that of `user.estimator`: The argument should be a `list` with at least one entry, name, that contains the name of the user-defined function, additional entries in the list are passed on to the user-specified goodness-of-fit function, and optionally an entry named `envir` may indicate where the user-defined function resides.

## Value

A `list` with the results of the specification search.

## Author(s)

Genaro Sucarrat, http://www.sucarrat.net/

## References

C. Jarque and A. Bera (1980): 'Efficient Tests for Normality, Homoscedasticity and Serial Independence'. Economics Letters 6, pp. 255-259

G. Ljung and G. Box (1979): 'On a Measure of Lack of Fit in Time Series Models'. Biometrika 66, pp. 265-270

F. Pretis, J. Reade and G. Sucarrat (2018): 'Automated General-to-Specific (GETS) Regression Modeling and Indicator Saturation for Outliers and Structural Breaks'. Journal of Statistical Software 86, Number 3, pp. 1-44

G. sucarrat (2019): 'User-Specified General-to-Specific and Indicator Saturation Methods', Munich Personal RePEc Archive: https://mpra.ub.uni-muenchen.de/96653/

## See Also

ols, diagnostics, infocrit, getsv

## Examples

```
##aim: do gets on the x-part (i.e. the covariates) of an arma-x model.
##create the user-defined estimator (essentially adding, renaming
##and re-organising the items returned by the estimator):
myEstimator <- function(y, x)
{
  tmp <- arima(y, order=c(1,0,1), xreg=x)

  #rename and re-organise:
  result <- list()
  result$coefficients <- tmp$coef[-c(1:3)]
  result$vcov <- tmp$var.coef
  result$vcov <- result$vcov[-c(1:3),-c(1:3)]
  result$logl <- tmp$loglik
  result$n <- tmp$nobs
  result$k <- NCOL(x)
  result$df <- result$n - result$k

  return(result)
}

##generate some data:
##a series w/structural break and eleven step-dummies near the break
set.seed(123)
eps <- arima.sim(list(ar=0.4, ma=0.1), 60)
x <- coredata(sim(eps, which.ones=25:35)) #eleven step-dummies
y <- 4*x[,"sis30"] + eps #create shift upwards at observation 30
plot(y)

##estimate the gum and then do gets in a single step:
getsFun(y, x, user.estimator=list(name="myEstimator"))

##estimate the gum and then do gets in two steps:
mygum <- myEstimator(y, x)
```

```
getsFun(y, x, user.estimator=list(name="myEstimator"), gum.result=mygum)
```

---

`getsm` *General-to-Specific (GETS) Modelling of an AR-X model (the mean specification) with log-ARCH-X errors (the log-variance specification).*

---

## Description

The starting model, an object of the 'arx' class, is referred to as the General Unrestricted Model (GUM). The `getsm` function undertakes multi-path GETS modelling of the mean specification, whereas `getsv` does the same for the log-variance specification. The diagnostic tests are undertaken on the standardised residuals, and the `keep` option enables regressors to be excluded from possible removal.

## Usage

```
##GETS-modelling of mean specification:
getsm(object, t.pval=0.05, wald.pval=t.pval, vcov.type=NULL,
    do.pet=TRUE, ar.LjungB=list(lag=NULL, pval=0.025),
    arch.LjungB=list(lag=NULL, pval=0.025), normality.JarqueB=NULL,
    user.diagnostics=NULL, info.method=c("sc","aic","aicc", "hq"),
    gof.function=NULL, gof.method=NULL, keep=NULL, include.gum=FALSE,
    include.1cut=TRUE, include.empty=FALSE, max.paths=NULL, turbo=FALSE,
    print.searchinfo=TRUE, plot=NULL, alarm=FALSE)

##GETS modelling of log-variance specification:
getsv(object, t.pval=0.05, wald.pval=t.pval,
    do.pet=TRUE, ar.LjungB=list(lag=NULL, pval=0.025),
    arch.LjungB=list(lag=NULL, pval=0.025), normality.JarqueB=NULL,
    user.diagnostics=NULL, info.method=c("sc","aic","aicc","hq"),
    gof.function=NULL, gof.method=NULL, keep=c(1), include.gum=FALSE,
    include.1cut=TRUE, include.empty=FALSE, max.paths=NULL, turbo=FALSE,
    print.searchinfo=TRUE, plot=NULL, alarm=FALSE)
```

## Arguments

| | |
|---|---|
| `object` | an object of class 'arx' |
| `t.pval` | numeric value between 0 and 1. The significance level used for the two-sided regressor significance t-tests |
| `wald.pval` | numeric value between 0 and 1. The significance level used for the Parsimonious Encompassing Tests (PETs). By default, it is the same as `t.pval` |
| `vcov.type` | the type of variance-covariance matrix used. If `NULL` (default), then the type used in the estimation of the 'arx' object is used. This can be overridden by either "ordinary" (i.e. the ordinary variance-covariance matrix) or "white" (i.e. the White (1980) heteroscedasticity robust variance-covariance matrix) |

do.pet — logical. If TRUE (default), then a Parsimonious Encompassing Test (PET) against the GUM is undertaken at each regressor removal for the joint significance of all the deleted regressors along the current path. If FALSE, then a PET is not undertaken at each regressor removal

ar.LjungB — a two-item list with names lag and pval, or NULL. In the former case lag contains the order of the Ljung and Box (1979) test for serial correlation in the standardised residuals, and pval contains the significance level. If lag=NULL (default), then the order used is that of the estimated 'arx' object. If ar.Ljungb=NULL, then the standardised residuals are not checked for serial correlation

arch.LjungB — a two-item list with names lag and pval, or NULL. In the former case, lag contains the order of the Ljung and Box (1979) test for serial correlation in the squared standardised residuals, and pval contains the significance level. If lag=NULL (default), then the order used is that of the estimated 'arx' object. If arch.Ljungb=NULL, then the standardised residuals are not checked for ARCH

normality.JarqueB — a value between 0 and 1, or NULL. In the former case, the Jarque and Bera (1980) test for non-normality is conducted using a significance level equal to the numeric value. If NULL, then no test for non-normality is undertaken

user.diagnostics — NULL or a list with two entries, name and pval, see the user.fun argument in diagnostics

info.method — character string, "sc" (default), "aic" or "hq", which determines the information criterion to be used when selecting among terminal models. The abbreviations are short for the Schwarz or Bayesian information criterion (sc), the Akaike information criterion (aic) and the Hannan-Quinn (hq) information criterion

gof.function — NULL (default) or a list, see getsFun. If NULL, then infocrit is used

gof.method — NULL (default) or a character, see getsFun. If NULL and gof.function is also NULL, then the best goodness-of-fit is characterised by a minimum value

keep — the regressors to be excluded from removal in the specification search. Note that keep=c(1) is obligatory when using getsv. This excludes the log-variance intercept from removal. The regressor numbering is contained in the reg.no column of the GUM

include.gum — logical. If TRUE, then the GUM (i.e. the starting model) is included among the terminal models. If FALSE (default), then the GUM is not included

include.1cut — logical. If TRUE, then the 1-cut model is added to the list of terminal models. If FALSE (default), then the 1-cut is not added, unless it is a terminal model in one of the paths

include.empty — logical. If TRUE, then an empty model is included among the terminal models, if it passes the diagnostic tests, even if it is not equal to one of the terminals. If FALSE (default), then the empty model is not included (unless it is one of the terminals)

max.paths — NULL (default) or an integer greater than 0. If NULL, then there is no limit to the number of paths. If an integer (e.g. 1), then this integer constitutes the maximum number of paths searched (e.g. a single path)

`turbo` logical. If `TRUE`, then (parts of) paths are not searched twice (or more) unnecessarily, thus yielding a significant potential for speed-gain. However, the checking of whether the search has arrived at a point it has already been comes with a slight computational overhead. Accordingly, if `turbo=TRUE`, then the total search time might in fact be higher than if `turbo=FALSE`. This happens if estimation is very fast, say, less than quarter of a second. Hence the default is `FALSE`

`print.searchinfo` logical. If `TRUE` (default), then a print is returned whenever simiplification along a new path is started

`plot` `NULL` or logical. If `TRUE`, then the fitted values and the residuals of the final model are plotted after model selection. If `FALSE`, then they are not. If `NULL` (default), then the value set by `options` determines whether a plot is produced or not

`alarm` logical. If `TRUE`, then a sound or beep is emitted (in order to alert the user) when the model selection ends

## Details

For an overview, see Pretis, Reade and Sucarrat (2018): https://www.jstatsoft.org/article/view/v086i03.

The arguments `user.diagnostics` and `gof.function` enable the specification of user-defined diagnostics and a user-defined goodness-of-fit function. For the former, see the documentation of `diagnostics`. For the latter, the principles of the same arguments in `getsFun` are followed, see its documentation under "Details", and Sucarrat (2019): https://mpra.ub.uni-muenchen.de/96653/.

## Value

A list of class 'gets'

## Author(s)

Genaro Sucarrat, http://www.sucarrat.net/

## References

C. Jarque and A. Bera (1980): 'Efficient Tests for Normality, Homoscedasticity and Serial Independence'. Economics Letters 6, pp. 255-259

G. Ljung and G. Box (1979): 'On a Measure of Lack of Fit in Time Series Models'. Biometrika 66, pp. 265-270

Felix Pretis, James Reade and Genaro Sucarrat (2018): 'Automated General-to-Specific (GETS) Regression Modeling and Indicator Saturation for Outliers and Structural Breaks'. Journal of Statistical Software 86, Number 3, pp. 1-44. https://www.jstatsoft.org/article/view/v086i03

Genaro Sucarrat (2019): 'User-Specified General-to-Specific and Indicator Saturation Methods'. https://mpra.ub.uni-muenchen.de/96653/

## See Also

Extraction functions: coef.gets, fitted.gets, paths, plot.gets, print.gets, residuals.gets, summary.gets, terminals, vcov.gets

Related functions: arx, eqwma, leqwma, zoo, getsFun

## Examples

```
##Simulate from an AR(1):
set.seed(123)
y <- arima.sim(list(ar=0.4), 80)

##Simulate four independent Gaussian regressors:
xregs <- matrix(rnorm(2*80), 80, 2)

##estimate an AR(2) with intercept and four conditioning
##regressors in the mean, and a log-ARCH(3) with log(xregs^2) as
##regressors in the log-variance:
gum01 <- arx(y, mc=TRUE, ar=1:2, mxreg=xregs, arch=1:3,
  vxreg=log(xregs^2))

##GETS model selection of the mean:
meanmod01 <- getsm(gum01)

##GETS model selection of the log-variance:
varmod01 <- getsv(gum01)

##GETS model selection of the mean with the mean intercept
##excluded from removal:
meanmod02 <- getsm(gum01, keep=1)

##GETS model selection of the mean with non-default
#serial-correlation diagnostics settings:
meanmod03 <- getsm(gum01, ar.LjungB=list(pval=0.05))

##GETS model selection of the mean with very liberal
##(20 percent) significance levels:
meanmod04 <- getsm(gum01, t.pval=0.2)

##GETS model selection of log-variance with all the
##log-ARCH terms excluded from removal:
varmod03 <- getsv(gum01, keep=2:4)
```

---

hpdata *Hoover and Perez (1999) data*

---

## Description

Data used by Hoover and Perez (1999) in their evaluation of General-to-Specific (GETS) modelling. A detailed description of the data is found in their Table 1 (page 172). The data are quarterly, comprise 20 variables (the first variable is the quarterly index) and runs from 1959:1 to 1995:1. This corresponds to 145 observations. The original source of the data is Citibank.

## Usage

```
data(hpdata)
```

## Format

`Date` a factor that contains the (quarterly) dates of the observations

`DCOINC` index of four coincident indicators

`GD` GNP price deflator

`GGEQ` government purchases of goods and services

`GGFEQ` federal purchases of goods and services

`GGFR` federal government receipts

`GNPQ` GNP

`GYDQ` disposable personal income

`GPIQ` gross private domestic investment

`FMRRA` total member bank reserves

`FMBASE` monetary base (feredal reserve bank of St. Louis)

`FM1DQ` M1

`FM2DQ` M2

`FSDJ` Dow Jones stock price

`FYAAAC` Moody's AAA corporate bond yield

`LHC` labour force (16 years+, civilian)

`LHUR` unemployment rate

`MU` unfilled orders (manufacturing, all industries)

`MO` new orders (manufacturing, all industries)

`GCQ` personal consumption expenditure

## Details

The data have been used for comparison and illustration of GETS model selection in several studies of the GETS methodology, including Hendry and Krolzig (1999, 2005), Doornik (2009) and Sucarrat and Escribano (2012).

## Source

Retrieved 14 October 2014 from: http://www.csus.edu/indiv/p/perezs/Data/data.htm

## References

David F. Hendry and Hans-Martin Krolzig (1999): 'Improving on 'Data mining reconsidered' by K.D. Hoover and S.J Perez', Econometrics Journal, Vol. 2, pp. 202-219.

David F. Hendry and Hans-Martin Krolzig (2005): 'The properties of automatic Gets modelling', Economic Journal 115, C32-C61.

Jurgen Doornik (2009): 'Autometrics', in Jennifer L. Castle and Neil Shephard (eds), 'The Methodology and Practice of Econometrics: A Festschrift in Honour of David F. Hendry', Oxford University Press, Oxford, pp. 88-121.

Pretis, Felix, Reade, James and Sucarrat, Genaro (2018): 'Automated General-to-Specific (GETS) Regression Modeling and Indicator Saturation for Outliers and Structural Breaks'. Journal of Statistical Software 86, Number 3, pp. 1-44.

## Examples

```
##load Hoover and Perez (1999) data:
data(hpdata)

##make quarterly data-matrix of zoo type:
newhpdata <- zooreg(hpdata[,-1], start=c(1959,1), frequency=4)

##plot data:
plot(newhpdata)

##transform data to log-differences in percent:
dloghpdata <- diff(log(newhpdata))*100

##plot log-differenced data:
plot(dloghpdata)
```

---

# iim *Make Indicator Matrices (Impulses, Steps, Trends)*

---

## Description

Auxiliary functions to make, respectively, matrices of impulse indicators (`iim`), step indicators (`sim`) and trend indicators (`tim`)

## Usage

```
##make matrix of impulse indicators:
iim(x, which.ones = NULL)

##make matrix of step indicators:
sim(x, which.ones = NULL)

##make matrix of trend indicators:
tim(x, which.ones = NULL, log.trend = FALSE)
```

## Arguments

| | |
|---|---|
| x | either an integer (the length of the series in question) or a series (a vector or matrix) from which to use the time-series index to make indicators of |
| which.ones | the locations of the impulses. If NULL (the default), then all impulses are returned |
| log.trend | logical. If TRUE, then the natural log is applied on the trends |

## Details

If x is a series or vector of observations, then the index of x will be used for the labelling of the impulses, and in the returned zoo object.

Note: For sim and tim the first indicator is removed, since it is exactly colinear with the others.

## Value

A zoo matrix containing the impulses

## Author(s)

Genaro Sucarrat, http://www.sucarrat.net/

## See Also

isat, zoo

## Examples

```
##generate series:
y <- rnorm(40)

##make matrix of impulse indicators:
mIIM <- iim(40)

##make matrix of step-indicators, but only every third:
mSIM <- sim(y, which.ones=seq(1,40,3))

##give quarterly time-series attributes to y-series:
y <- zooreg(y, frequency=4, end=c(2015,4))

##make matrix of trend-indicators with quarterly labels:
mTIM <- tim(y)
```

`infldata` *Quarterly Norwegian year-on-year CPI inflation*

## Description

Quarterly Norwegian year-on-year CPI inflation from 1989(1) to 2015(4).

## Usage

```
data("infldata")
```

## Format

A data frame with 108 observations on the following 5 variables:

`date` a factor containing the dates

`infl` year-on-year inflation

`q2dum` a dummy variable equal to 1 in quarter 2 and 0 otherwise

`q3dum` a dummy variable equal to 1 in quarter 3 and 0 otherwise

`q4dum` a dummy variable equal to 1 in quarter 4 and 0 otherwise

## Source

Statistics Norway (SSB): http://www.ssb.no/. The raw data comprise monthly CPI data obtained via http://www.ssb.no/tabell/08183/.

## References

Pretis, Felix, Reade, James and Sucarrat, Genaro (2018): 'Automated General-to-Specific (GETS) Regression Modeling and Indicator Saturation for Outliers and Structural Breaks'. Journal of Statistical Software 86, Number 3, pp. 1-44

## Examples

```
data(infldata)
infldata <- zooreg(infldata[,-1], frequency=4, start=c(1989,1))
plot(infldata[,"infl"])
```

`infocrit` *Computes the Average Value of an Information Criterion*

## Description

Given a log-likelihood, the number of observations and the number of estimated parameters, the average value of a chosen information criterion is computed. This facilitates comparison of models that are estimated with a different number of observations, e.g. due to different lags.

## Usage

```
infocrit(x, method=c("sc","aic","aicc","hq"))

info.criterion(logl, n=NULL, k=NULL, method=c("sc","aic","aicc","hq"))
```

## Arguments

| | |
|---|---|
| `x` | a `list` that contains, at least, three items: `logl` (a numeric, the log-likelihood), `k` (a numeric, usually the number of estimated parameters) and `n` (a numeric, the number of observations) |
| `method` | character, either "sc" (default), "aic", "aicc" or "hq" |
| `logl` | numeric, the value of the log-likelihood |
| `n` | integer, number of observations |
| `k` | integer, number of parameters |

## Details

Contrary to `AIC` and `BIC`, `info.criterion` computes the average criterion value (i.e. division by the number of observations). This facilitates comparison of models that are estimated with a different number of observations, e.g. due to different lags.

## Value

`infocrit`: a numeric (i.e. the value of the chosen information criterion)

`info.criterion`: a list with elements

| | |
|---|---|
| `method` | type of information criterion |
| `n` | number of observations |
| `k` | number of parameters |
| `value` | the value on the information criterion |

## Author(s)

Genaro Sucarrat, http://www.sucarrat.net/

## References

H. Akaike (1974): 'A new look at the statistical model identification'. IEEE Transactions on Automatic Control 19, pp. 716-723

E. Hannan and B. Quinn (1979): 'The determination of the order of an autoregression'. Journal of the Royal Statistical Society B 41, pp. 190-195

C.M. Hurvich and C.-L. Tsai (1989): 'Regression and Time Series Model Selection in Small Samples'. Biometrika 76, pp. 297-307

Pretis, Felix, Reade, James and Sucarrat, Genaro (2018): 'Automated General-to-Specific (GETS) Regression Modeling and Indicator Saturation for Outliers and Structural Breaks'. Journal of Statistical Software 86, Number 3, pp. 1-44

G. Schwarz (1978): 'Estimating the dimension of a model'. The Annals of Statistics 6, pp. 461-464

---

`isat` *Indicator Saturation*

---

## Description

The `isat` function undertakes multi-path indicator saturation to detect outliers and mean-shifts using impulses (IIS), step-shifts (SIS), or trend-indicators (TIS). Indicators are partitioned into blocks and selected over at a chosen level of significance (`t.pval`) using the `getsm` function.

## Usage

```
isat(y, mc=TRUE, ar=NULL, ewma=NULL, mxreg=NULL, iis=FALSE, sis=TRUE,
  tis=FALSE, uis=FALSE, blocks=NULL, ratio.threshold=0.8, max.block.size=30,
  t.pval=0.001, wald.pval=t.pval,
  vcov.type= c("ordinary","white","newey-west"), do.pet=FALSE, ar.LjungB=NULL,
  arch.LjungB=NULL, normality.JarqueB=NULL, info.method=c("sc","aic","hq"),
  user.diagnostics=NULL, user.estimator=NULL, gof.function=NULL,
  gof.method = c("min", "max"), include.gum=NULL, include.1cut=FALSE,
  include.empty=FALSE, max.paths=NULL, parallel.options=NULL, turbo=FALSE,
  tol=1e-07, LAPACK=FALSE, max.regs=NULL, print.searchinfo=TRUE, plot=NULL,
  alarm=FALSE)
```

## Arguments

| | |
|---|---|
| `y` | numeric vector, time-series or `zoo` object. Missing values in the beginning and at the end of the series is allowed, as they are removed with the `na.trim` command |
| `mc` | logical. `TRUE` (default) includes an intercept in the mean specification, whereas FALSE does not |
| `ar` | integer vector, say, c(2,4) or 1:4. The AR-lags to include in the mean specification |
| `ewma` | either NULL (default) or a list with arguments sent to the `eqwma` function. In the latter case a lagged moving average of y is included as a regressor |

mxreg — numeric vector or matrix, say, a zoo object, of conditioning variables. Note that missing values in the beginning or at the end of the series is allowed, as they are removed with the na.trim command. Note also that, if both y and mxreg are zoo objects, then their samples are chosen to match

iis — logical. If TRUE, impulse indicator saturation is performed.

sis — logical. If TRUE, step indicator saturation is performed.

tis — logical. If TRUE, trend indicator saturation is performed.

uis — a matrix of regressors, or a list of matrices.

blocks — NULL (default), an integer (the number of blocks) or a user-specified list that indicates how blocks should be put together. If NULL, then the number of blocks is determined automatically

ratio.threshold — Minimum ratio of variables in each block to total observations to determine the block size, default=0.8. Only relevant if blocks = NULL

max.block.size — Maximum size of block of variables to be selected over, default=30. Block size used is the maximum of given by either the ratio.threshold and max.block.size

t.pval — numeric value between 0 and 1. The significance level used for the two-sided regressor significance t-tests

wald.pval — numeric value between 0 and 1. The significance level used for the Parsimonious Encompassing Tests (PETs)

vcov.type — the type of variance-covariance matrix used. If NULL (default), then the type used is that of the 'arx' object. This can be overridden by either "ordinary" (i.e. the ordinary variance-covariance matrix) or "white" (i.e. the White (1980) heteroscedasticity robust variance-covariance matrix)

do.pet — logical. If TRUE, then a Parsimonious Encompassing Test (PET) against the GUM is undertaken at each regressor removal for the joint significance of all the deleted regressors along the current path. If FALSE (default), then a PET is not undertaken at each regressor removal. By default, the numeric value is the same as that of t.pval

ar.LjungB — a two-item list with names lag and pval, or NULL (default). In the former case lag contains the order of the Ljung and Box (1979) test for serial correlation in the standardised residuals, and pval contains the significance level. If lag=NULL (default), then the order used is that of the estimated 'arx' object. If ar.Ljungb=NULL, then the standardised residuals are not checked for serial correlation

arch.LjungB — a two-item list with names lag and pval, or NULL (default). In the former case, lag contains the order of the Ljung and Box (1979) test for serial correlation in the squared standardised residuals, and pval contains the significance level. If lag=NULL (default), then the order used is that of the estimated 'arx' object. If arch.Ljungb=NULL, then the standardised residuals are not checked for ARCH

normality.JarqueB — NULL (the default) or a value between 0 and 1. In the latter case, a test for non-normality is conducted using a significance level equal to normality.JarqueB. If NULL, then no test for non-normality is conducted

`info.method` character string, "sc" (default), "aic" or "hq", which determines the information criterion to be used when selecting among terminal models. The abbreviations are short for the Schwarz or Bayesian information criterion (sc), the Akaike information criterion (aic) and the Hannan-Quinn (hq) information criterion

`user.diagnostics` NULL or a list with two entries, name and pval, see the user.fun argument in diagnostics

`user.estimator` NULL or a list with at least one entry, name, see the user.estimator argument in getsFun

`gof.function` NULL or a list with at least one entry, name, see the user.estimator argument in getsFun

`gof.method` NULL or a character that determines whether the best Goodness-of-Fit is a minimum or maximum

`include.gum` ignored (temporarily deprecated)

`include.1cut` logical. If TRUE, then the 1-cut model is included among the terminal models, if it passes the diagnostic tests, even if it is not equal to one of the terminals. If FALSE (default), then the 1-cut model is not included (unless it is one of the terminals)

`include.empty` logical. If TRUE, then an empty model is included among the terminal models, if it passes the diagnostic tests, even if it is not equal to one of the terminals. If FALSE (default), then the empty model is not included (unless it is one of the terminals)

`max.paths` NULL (default) or an integer indicating the maximum number of paths to search

`parallel.options` NULL or an integer, i.e. the number of cores/threads to be used for parallel computing (implemented w/makeCluster and parLapply)

`turbo` logical. If TRUE, then (parts of) paths are not searched twice (or more) unnecessarily, thus yielding a significant potential for speed-gain. However, the checking of whether the search has arrived at a point it has already been comes with a slight computational overhead. Accordingly, if turbo=TRUE, then the total search time might in fact be higher than if turbo=FALSE. This happens if estimation is very fast, say, less than quarter of a second. Hence the default is FALSE

`tol` numeric value (default = 1e-07). The tolerance for detecting linear dependencies in the columns of the regressors (see qr function). Only used if LAPACK is FALSE (default)

`LAPACK` logical. If TRUE, then use LAPACK. If FALSE (default), then use LINPACK (see qr function)

`max.regs` integer. The maximum number of regressions along a deletion path. It is not recommended that this is altered

`print.searchinfo` logical. If TRUE (default), then a print is returned whenever simiplification along a new path is started, and whenever regressors are dropped due to exact multicolinearity

| | |
|---|---|
| plot | NULL or logical. If TRUE, then the fitted values and the residuals of the final model are plotted after model selection. If NULL (default), then the value set by options determines whether a plot is produced or not. |
| alarm | logical. If TRUE, then a sound is emitted (in order to alert the user) when the model selection ends |

## Details

Multi-path indicator saturation using impulses (IIS), step-shifts (SIS), or trend-indicators (TIS). Indicators are partitioned into sequential blocks (as of beta version 0.7) where the block intervals are defined by the ratio of variables to observations in each block and a specified maximum block size. Indicators are selected over using the getsm function. Retained indicators in each block are combined and re-selected over. Fixed covariates that are not selected over can be included in the regression model either in the mxreg matrix, or for auto-regressive terms through the ar specification. See Hendry, Johansen and Santos (2007) and Castle, Doornik, Hendry, and Pretis (2015)

## Value

A list of class 'gets'

## Author(s)

Felix Pretis, http://www.felixpretis.org/
James Reade, https://sites.google.com/site/jjamesreade/
Genaro Sucarrat, http://www.sucarrat.net/

## References

Castle, Jennifer, L., Doornik, Jurgen, A., Hendry, David F., and Pretis, Felix (2015): 'Detecting Location Shifts during Model Selection by Step-Indicator Saturation', Econometrics, vol 3:2, 240-264.

Hendry, David, F., Johansen, Soren, and Santos, Carlos (2007): 'Automatic selection of indicators in a fully saturated regression'. Computational Statistics, vol 23:1, pp.317-335.

Pretis, Felix, Reade, James and Sucarrat, Genaro (2018): 'Automated General-to-Specific (GETS) Regression Modeling and Indicator Saturation for Outliers and Structural Breaks'. Journal of Statistical Software 86, Number 3, pp. 1-44

## See Also

Extraction functions for 'gets' objects: coef.gets, fitted.gets, paths, plot.gets, print.gets, residuals.gets, summary.gets, terminals, vcov.gets

Related functions: arx, eqwma, leqwma, zoo, getsFun

## Examples

```
##SIS using the Nile data
data(Nile)
isat(Nile, sis=TRUE, iis=FALSE, plot=TRUE, t.pval=0.005)

##SIS using the Nile data in an autoregressive model
#isat(Nile, ar=1:2, sis=TRUE, iis=FALSE, plot=TRUE, t.pval=0.005)

##HP Data
##load Hoover and Perez (1999) data:
#data(hpdata)

##make quarterly data-matrix of zoo type
##(GCQ = personal consumption expenditure):
#y <- zooreg(hpdata$GCQ, 1959, frequency=4)

##transform data to log-differences:
#dlogy <- diff(log(y))

##run isat with step impulse saturation on four
##lags and a constant 1 percent significance level:
#isat(dlogy, ar=1:4, sis=TRUE, t.pval =0.01)

##Example with additional covariates entering through mxreg:

##(GYDQ = disposable personal income):
#x <- zooreg(hpdata$GYDQ, 1959, frequency=4)

##transform data to log-differences:
#dlogx <- diff(log(x))

##run isat with step impulse saturation on four
##lags and a constant 1 percent significance level:
#isat(dlogy, mxreg=dlogx, ar=1:4, sis=TRUE, t.pval =0.01)
```

---

# isatdates *Extracting Indicator Saturation Breakdates*

---

## Description

Takes an isat object and extracts the break dates together with their estimated coefficients.

## Usage

```
isatdates(x)
```

## Arguments

| | |
|---|---|
| x | an isat object |

### Details

The function extracts the breakdates determined by isat for iis, sis, and tis, together with their estimated coefficients and standard errors.

### Value

Returns a list of three elements (one for iis, sis, and tis). Each element lists the name of the break variable, the time index of the break (labelled 'date'), the index of the break date, the estimated coefficient, the standard error of the estimated coefficient, as well as the corresponding t-statistic and p-value.

### Author(s)

Felix Pretis, http://www.felixpretis.org/

### References

Pretis, F., Reade, J., & Sucarrat, G. (2018). Automated General-to-Specific (GETS) regression modeling and indicator saturation methods for the detection of outliers and structural breaks. Journal of Statistical Software, 86(3).

### See Also

isat

### Examples

```
###Break date extraction of the Nile data
nile <- as.zoo(Nile)
isat.nile <- isat(nile, sis=TRUE, iis=FALSE, plot=TRUE, t.pval=0.005)
isatdates(isat.nile)
```

---

## isatloop *Repeated Impulse Indicator Saturation*

---

### Description

Runs isat repeatedly at pre-specified significance levels to yield multiple iterations used in outlierscaletest.

### Usage

```
isatloop(num=c(seq(from=20, to=1, by=-1)), t.pval.spec = FALSE,
print=FALSE, y, ar=NULL, iis=TRUE, sis=FALSE, ...)
```

## Arguments

| | |
|---|---|
| num | numeric, target expected number of outliers under the null hypothesis, or target proportion of outliers if t.pval.spec==TRUE |
| t.pval.spec | logical, if TRUE, then num specifies proportion rather than number of targeted outliers |
| print | logical, if TRUE, then iterations are printed |
| y | numeric vector, time-series or zoo object. Missing values in the beginning and at the end of the series is allowed, as they are removed with the na.trim command |
| ar | integer vector, say, c(2,4) or 1:4. The AR-lags to include in the mean specification |
| iis | logical, whether to use iis |
| sis | logical, whether to use sis, default is FALSE |
| ... | any argument from isat can also be used in isatloop |

## Details

The function repeatedly runs isat detecting outliers in a model of y at different chosen target levels of significance speciefied in num. The output of this function is used as the input for the outlierscaletest function. All additional arguments from isat can be passed to isatloop.

## Value

Returns a list of two items. The first item is the number of observations. The second item is a dataframe containing the expected and observed proportion (and number of outliers) for each specified significance level of selection.

## Author(s)

Felix Pretis, http://www.felixpretis.org/

## References

Jiao, X. & Pretis, F. (2019). Testing the Presence of Outliers in Regression Models. Discussion Paper.

Pretis, F., Reade, J., & Sucarrat, G. (2018). Automated General-to-Specific (GETS) regression modeling and indicator saturation methods for the detection of outliers and structural breaks. Journal of Statistical Software, 86(3).

## See Also

isat, outlierscaletest

## Examples

```
###Repeated isat models using the Nile dataset
### where p-values are chosen such that the expected number of outliers under the null
### corresponds to 1, 2, 3, 4 and 5.
nile <- as.zoo(Nile)
isat.nile.loop <- isatloop(y=nile, iis=TRUE, num=c(1,2, 3, 4, 5))
```

---

# isattest *Indicator Saturation Test*

---

## Description

Takes an 'isat' object returned by the `isat` function as input and returns the results of a hypothesis test on the time-varying intercept or long-run equilibrium against a specified null-hypothesis for a chosen level of significance - see Pretis (2015).

## Usage

```
isattest(x, hnull=0, lr=FALSE, ci.pval=0.99, plot=NULL, plot.turn=FALSE,
  conscorr=FALSE, effcorr=FALSE, mcor = 1, biascorr=FALSE, mxfull = NULL,
  mxbreak=NULL)
```

## Arguments

| | |
|---|---|
| `x` | a 'gets' object obtained with the `isat` function |
| `hnull` | numeric. the null-hypothesis value to be tested against. |
| `lr` | logical. If TRUE and 'x' contains autoregressive elements, then `isattest` tests on the long-run equilibrium path. See Pretis (2015). |
| `ci.pval` | numeric between 0 and 1. Default is 0.99, the level of significance for the confidence interval of the test against 'hnull'. |
| `plot` | logical. If TRUE, then a plot showing the coefficient path and bias relative to 'hnull' is shown. |
| `plot.turn` | logical. If TRUE, then the plot output adds the time of the breaks to the plot showing the bias relative to 'hnull'. |
| `biascorr` | logical. If TRUE, then the coefficient path is bias-corrected using `biascorr`. This is only valid for the non-dynamic test without additional covariates. |
| `conscorr` | logical. If TRUE then the Johansen and Nielsen (2016) impulse-indicator consistency correction is applied to estimated residual variance. |
| `effcorr` | logical. If TRUE then the Johansen and Nielsen (2016) m-step efficiency correction is applied to estimated standard errors of 'fixed' regressors. |
| `mcor` | integer. The m-step efficiency correction factor, where m=mcor. |
| `mxfull` | string. The name of the full-sample variable when constructing the coefficient path of user-specified break variables. |
| `mxbreak` | string. The name of the break variables used to construct the coefficient path of user-specified break variables. |

## Details

The function tests the coefficient path (or long-run equilibrium path) against a specified null hypothesis at a chosen level of significance. If conducted on an `isat` model of a forecast error or relative forecast differential, then this corresponds to the test of time-varying predictive accuracy of Pretis (2015). The resulting output plot shows the coefficient path in the top panel (where 'hnull' is plotted as dotted lines), with the bias (significant difference relative to 'hnull') in the lower panel. If `mxfull` and `mxbreak` are specified, then the function tests on the coefficient path of the user-specified variable, where `mxfull` denotes the ful-sample variable name, to which the `mxbreak` variables are added. To correct for the under-estimation of the residual variance, the argument `conscorr` implements the Johansen and Nielsen (2016) consistency correction, and `effcorr` adds the efficiency correction for standard errors on fixed regressors which are not selected over.

## Value

A Tx4 matrix (with T = number of observations) where the first two columns denote the confidence interval of the coefficient path (or the long-run equilibrium path if 'lr=TRUE'). The third and fourth column denote the bias of the coefficient path relative to the chosen null-hypothesis, where 'bias.high' denotes the bias when the series tested is above the hypothesized value, and 'bias.low' denotes the bias when the series tested is significantly below the hypothesized value.

## Author(s)

Felix Pretis, http://www.felixpretis.org/

## References

Johansen, S., & Nielsen, B. (2016): 'Asymptotic theory of outlier detection algorithms for linear time series regression models.' Scandinavian Journal of Statistics, 43(2), 321-348.

Pretis, F. (2015): 'Testing for time-varying predictive accuracy using bias-corrected indicator saturation'. Oxford Department of Economics Discussion Paper.

Hendry, David, F., Johansen, Soren, and Santos, Carlos (2007): 'Automatic selection of indicators in a fully saturated regression'. Computational Statistics, vol 23:1, pp.317-335.

## See Also

`isat`, `coef.gets`, `plot.gets`, `biascorr`, `isatvar`

## Examples

```
##Using artificial data:
#set.seed(123)
#d <- matrix(0,100,1)
#d[35:55] <- 1
#e <- rnorm(100, 0, 1)
#y <- d*2  +e
#plot(y, type="l")

##Static Test against hnull=0 using bias-correction:
```

```
#ys <- isat(y, sis=TRUE, iis=FALSE, tis=FALSE, t.pval=0.01)
#isattest(ys, hnull=0, lr=FALSE, ci.pval = 0.99, plot.turn = FALSE, biascorr=TRUE)

##Dynamic Test of the long-run equilibrium against hnull=2 with breakpoints
##labelled in the plot:

#ys <- isat(y, sis=TRUE, iis=FALSE, tis=FALSE, t.pval=0.01, ar=1:2)
#isattest(ys, hnull=2, lr=TRUE, ci.pval = 0.99, plot.turn = TRUE, biascorr=FALSE)
```

---

`isatvar` *Variance of the coefficient path*

---

## Description

Takes an 'isat' object returned by the `isat` function as input and returns the coefficient path of the constant (and long-run equilibrium if 'lr' is specified) together with its approximate variance and standard errors. If `mxfull` and `mxbreak` are specified, then the function returns the coefficient path of the user-specified variable.

## Usage

```
isatvar(x, lr=FALSE, conscorr=FALSE, effcorr=FALSE, mcor = 1,
  mxfull = NULL, mxbreak=NULL)
```

## Arguments

| | |
|---|---|
| `x` | a 'gets' object obtained with the `isat` function |
| `lr` | logical. If TRUE and 'x' contains autoregressive elements, then `isatvar` also returns the long-run equilibrium coefficient path with its variance and standard deviation. See Pretis (2015). |
| `conscorr` | logical. If TRUE then the Johansen and Nielsen (2016) impulse-indicator consistency correction is applied to estimated residual variance. |
| `effcorr` | logical. If TRUE then the Johansen and Nielsen (2016) m-step efficiency correction is applied to estimated standard errors of 'fixed' regressors. |
| `mcor` | integer. The m-step efficiency correction factor, where m=mcor. |
| `mxfull` | string. The name of the full-sample variable when constructing the coefficient path of user-specified break variables. |
| `mxbreak` | string. The name of the break variables used to construct the coefficient path of user-specified break variables. |

## Details

The function computes the approximate variance and standard errors of the intercept term with structural breaks determined by isat. This permits hypothesis testing and plotting of approximate confidence intervals for the intercept in the presence of structural breaks. For dynamic autoregressive models in isat the lr argument returns the time-varying long-run equilibrium together with its approximate variance and standard errors. If mxfull and mxbreak are specified, then the function returns the coefficient path of the user-specified variable, where mxfull denotes the ful-sample variable name, to which the mxbreak variables are added. To correct for the under-estimation of the residual variance, the argument conscorr implements the Johansen and Nielsen (2016) consistency correction, and effcorr adds the efficiency correction for standard errors on fixed regressors which are not selected over.

## Value

If lr=FALSE: A Tx4 matrix (with T = number of observations) where the first column denotes the coefficient path relative to the full sample coefficient, the second column the coefficient path of the intercept, the third the approximate variance of the coefficient path, and the fourth column the approximate standard errors of the coefficient path. If lr=TRUE: A Tx7 matrix where the first four columns are identical to the lr=FALSE case, and the additional columns denote the long-run equilibrium coefficient path, together with the approximate variance and standard errors of the long-run equilibrium coefficient path.

## Author(s)

Felix Pretis, http://www.felixpretis.org/
James Reade, https://sites.google.com/site/jjamesreade/

## References

Pretis, F. (2015): 'Testing for time-varying predictive accuracy using bias-corrected indicator saturation'. Oxford Department of Economics ???orking Paper.

Johansen, S., & Nielsen, B. (2016): 'Asymptotic theory of outlier detection algorithms for linear time series regression models.' Scandinavian Journal of Statistics, 43(2), 321-348.

Pretis, Felix, Reade, James and Sucarrat, Genaro (2018): 'Automated General-to-Specific (GETS) Regression Modeling and Indicator Saturation for Outliers and Structural Breaks'. Journal of Statistical Software 86, Number 3, pp. 1-44

## See Also

isat, coef.gets, plot.gets, biascorr, isattest

## Examples

```
##Variance in presence of a break
#nile <- as.zoo(Nile)
#isat.nile <- isat(nile, sis=TRUE, iis=FALSE, plot=FALSE, t.pval=0.005)
#var <- isatvar(isat.nile)

#plot(nile)
```

```
#lines(isat.nile$mean.fit, col="red")
#lines(isat.nile$mean.fit + 2*var$const.se, col="blue", lty=3)
#lines(isat.nile$mean.fit - 2*var$const.se, col="blue", lty=3)

##Variance when there is no break
#set.seed(1)
#x <- as.zoo(rnorm(100, 0, 1))
#isat.x <- isat(x, sis=TRUE, iis=FALSE, plot=TRUE, t.pval=0.005)
#var.x <- isatvar(isat.x)

#plot(x)
#lines(isat.x$mean.fit, col="red")
#lines(isat.x$mean.fit + 2*var.x[,2], col="blue", lty=3)
#lines(isat.x$mean.fit - 2*var.x[,2], col="blue", lty=3)

##Variance of the long-run equilibrium coefficient path

#nile <- as.zoo(Nile)
#isat.nile <- isat(nile, sis=TRUE, iis=FALSE, plot=TRUE, t.pval=0.005, ar=1:2)
#var <- isatvar(isat.nile, lr=TRUE)
```

---

`isatvarcorrect` *Consistency and Efficiency Correction for Impulse Indicator Saturation*

---

## Description

Takes an `isat` object and corrects the estimates of the error variance and the estimated standard errors of 'forced' regressors.

## Usage

```
isatvarcorrect(x, mcor=1)
```

## Arguments

| | |
|---|---|
| `x` | an `isat` object |
| `mcor` | integer, number of iterations in the correction. Default = 1. |

## Details

Impulse indicator saturation results in an under-estimation of the error variance as well as the variance of regressors not selected over. The magnitude of the inconsistency increases with the p-value of selection (`t.pval`). The function takes an `isat` object and applies the impulse indicator consistency (`isvarcor`) and efficiency correction (`isvareffcor`) of the estimated error variance and the estimated variance of regressors not selected over. See Johansen and Nielsen (2016a) and (2016b).

## Value

Returns an isat object in which the estimated standard errors, t-statistics, p-values, standard error of the regression, and log-likelihood are consistency and efficiency corrected when using impulse indicator saturation (iis=TRUE).

## Author(s)

Felix Pretis, http://www.felixpretis.org/

## References

Johansen, S., & Nielsen, B. (2016a). Asymptotic theory of outlier detection algorithms for linear time series regression models. Scandinavian Journal of Statistics, 43(2), 321-348.

Johansen, S., & Nielsen, B. (2016b). Rejoinder: Asymptotic Theory of Outlier Detection Algorithms for Linear. Scandinavian Journal of Statistics, 43(2), 374-381.

Pretis, F., Reade, J., & Sucarrat, G. (2018). Automated General-to-Specific (GETS) regression modeling and indicator saturation methods for the detection of outliers and structural breaks. Journal of Statistical Software, 86(3).

## See Also

isat, isvarcor, isvareffcor

## Examples

```
###Consistency and Efficiency Correction of Impulse Indicator Estimates
nile <- as.zoo(Nile)
isat.nile <- isat(nile, sis=FALSE, iis=TRUE, plot=TRUE, t.pval=0.1)
isat.nile.corrected <- isatvarcorrect(isat.nile)

isat.nile$sigma2
isat.nile.corrected$sigma2
```

---

# isvarcor *IIS Consistency Correction*

---

## Description

Consistency correction for estimate of residual variance when using impulse indicator saturation.

## Usage

```
isvarcor(t.pval, sigma)
```

### Arguments

| | |
|---|---|
| t.pval | numeric value. the p-value of selection in the impulse indicator saturation model. |
| sigma | numeric value. The estimated standard deviation of the residuals from the impulse indicator saturation model. |

### Details

The Johansen and Nielsen (2016) impulse-indicator consistency correction for the estimated residual standard deviation.

### Value

a data frame containing the corrected standard deviation `$sigma.cor` and the correction factor used `$corxi`

### Author(s)

Felix Pretis, http://www.felixpretis.org/

### References

Johansen, S., & Nielsen, B. (2016): 'Asymptotic theory of outlier detection algorithms for linear time series regression models.' Scandinavian Journal of Statistics, 43(2), 321-348.

Pretis, Felix, Reade, James and Sucarrat, Genaro (2018): 'Automated General-to-Specific (GETS) Regression Modeling and Indicator Saturation for Outliers and Structural Breaks'. Journal of Statistical Software 86, Number 3, pp. 1-44

### See Also

isatvar

### Examples

```
isvarcor(t.pval=0.05, sigma=2)
```

---

## isvareffcor *IIS Efficiency Correction*

---

### Description

Efficiency correction for the estimates of coefficient standard errors on fixed regressors.

### Usage

```
isvareffcor(t.pval, se, m=1)
```

### Arguments

| | |
|---|---|
| t.pval | numeric value. the p-value of selection in the impulse indicator saturation model. |
| se | numeric value or vector. The estimated standard errors of the coefficients on fixed regressors in impulse indicator saturation model. |
| m | integer. The m-step correction factor. |

### Details

The Johansen and Nielsen (2016) impulse-indicator efficiency correction for the estimated standard errors on fixed regressors in impulse indicator models.

### Value

a data frame containing the corrected standard deviation `$se.cor` and the correction factor used `$eta.m`

### Author(s)

Felix Pretis, http://www.felixpretis.org/

### References

Johansen, S., & Nielsen, B. (2016): 'Asymptotic theory of outlier detection algorithms for linear time series regression models.' Scandinavian Journal of Statistics, 43(2), 321-348.

Pretis, Felix, Reade, James and Sucarrat, Genaro (2018): 'Automated General-to-Specific (GETS) Regression Modeling and Indicator Saturation for Outliers and Structural Breaks'. Journal of Statistical Software 86, Number 3, pp. 1-44

### See Also

isatvar

### Examples

```
isvareffcor(t.pval=0.05, se=2, m=1)
```

---

## mvrnormsim *Simulate from a Multivariate Normal Distribution*

---

### Description

Produces one or more samples from the specified multivariate normal distribution. Used in outlierscaletest.

## Usage

```
mvrnormsim(n = 1, mu, Sigma, tol = 1e-6, empirical = FALSE)
```

## Arguments

| | |
|---|---|
| n | the number of samples required. |
| mu | a vector giving the means of the variables. |
| Sigma | a positive-definite symmetric matrix specifying the covariance matrix of the variables. |
| tol | tolerance (relative to largest variance) for numerical lack of positive-definiteness in Sigma. |
| empirical | logical. If true, mu and Sigma specify the empirical not population mean and covariance matrix. |

## Details

Original function mvrnorm developed by Venables, W. N. & Ripley. in package MASS, https://CRAN.R-project.org/package=MASS.

## Value

If n = 1 a vector of the same length as mu, otherwise an n by length(mu) matrix with one sample in each row.

## Author(s)

Venables, W. N. & Ripley, with modifications by Felix Pretis, http://www.felixpretis.org/

## References

Venables, W. N. & Ripley, B. D. (2019): 'MASS: Support Functions and Datasets for Venables and Ripley's MASS'. https://CRAN.R-project.org/package=MASS

Venables, W. N. & Ripley, B. D. (2002) Modern Applied Statistics with S. Fourth Edition. Springer, New York. ISBN 0-387-95457-0

## See Also

outlierscaletest

## Examples

```
Sigma <- matrix(c(3,2,1,7),2,2)
mvrnormsim(n=2, mu=c(1,2), Sigma)
```

`ols` *OLS estimation*

## Description

OLS estimation with the QR decomposition and, for some options, computation of variance- covariance matrices

## Usage

```
ols(y, x, untransformed.residuals=NULL, tol=1e-07, LAPACK=FALSE, method=3,
  variance.spec=NULL, ...)
```

## Arguments

| | |
|---|---|
| `y` | numeric vector, the regressand |
| `x` | numeric matrix, the regressors |
| `untransformed.residuals` | `NULL` (default) or, when `ols` is used with `method=6`, a numeric vector containing the untransformed residuals |
| `tol` | numeric value. The tolerance for detecting linear dependencies in the columns of the regressors, see the `qr` function. Only used if `LAPACK` is `FALSE` |
| `LAPACK` | logical, `TRUE` or `FALSE` (default). If true use LAPACK otherwise use LINPACK, see the `qr` function |
| `method` | an integer, 1 to 6, that determines the estimation method |
| `variance.spec` | `NULL` or a `list` with items that specifies the log-variance model to be estimated, see `arx` |
| `...` | further arguments (currently ignored) |

## Details

`method = 1` or `method = 2` only returns the OLS coefficient estimates together with the QR- information, the former being slightly faster. `method=3` returns, in addition, the ordinary variance-covariance matrix of the OLS estimator. `method=4` returns the White (1980) heteroscedasticity robust variance-covariance matrix in addition to the information returned by `method=3`, whereas `method=5` does the same except that the variance-covariance matrix now is that of Newey and West (1987). `method=6` undertakes OLS estimation of a log-variance model, see Pretis, Reade and Sucarrat (2018, Section 4). Alternatively, for `method` 1 to 5, a log-variance model is also estimated if `variance.spec` is not `NULL`.

## Value

A list with items depending on `method`

### Author(s)

Genaro Sucarrat, http://www.sucarrat.net/

### References

W. Newey and K. West (1987): 'A Simple Positive Semi-Definite, Heteroskedasticity and Autocorrelation Consistent Covariance Matrix', Econometrica 55, pp. 703-708.

F. Pretis, J. Reade and G. Sucarrat (2018): 'Automated General-to-Specific (GETS) Regression Modeling and Indicator Saturation for Outliers and Structural Breaks', Journal of Statistical Software 86, Issue 3, pp. 1-44, DOI: https://doi.org/10.18637/jss.v086.i03

H. White (1980): 'A Heteroskedasticity-Consistent Covariance Matrix and a Direct Test for Heteroskedasticity', Econometrica 48, pp. 817-838.

### See Also

qr, solve.qr, arx

---

## outlierscaletest *Sum and Sup Scaling Outlier Tests*

---

### Description

Computes the Sum and Supremum Scaling Tests for the overall presence of outliers based on Jiao and Pretis (2019).

### Usage

```
outlierscaletest(x, nsim = 10000)
```

### Arguments

| | |
|---|---|
| x | list, output of the isatloop function |
| nsim | integer, number of replications to simulate critical values for the Sup test |

### Details

The function takes the output of the isatloop function and computes the Scaling Sum and Supremum Tests for the presence of outliers from Jiao and Pretis (2019). The test compares the expected and observed proportion of outliers over the range of different significance levels of selection specified in isatloop. The Sum test compares the sum of deviations against the standard normal distribution, the Sup test compares the supremum of deviations against critical values simulated with nsim replications. The null hypothesis is that the observed proportion of outliers scales with the proportion of outliers under the null of no outliers.

## Value

Returns a list of two `htest` objects. The first providing the results of the Sum test on the sum of the deviation of outliers against a standard normal distribution. The second providing the results on the supremum of the deviation of outliers against simulated critical values.

## Author(s)

Xiyu Jiao, & Felix Pretis, http://www.felixpretis.org/

## References

Jiao, X. & Pretis, F. (2019). Testing the Presence of Outliers in Regression Models. Discussion Paper.

Pretis, F., Reade, J., & Sucarrat, G. (2018). Automated General-to-Specific (GETS) regression modeling and indicator saturation methods for the detection of outliers and structural breaks. Journal of Statistical Software, 86(3).

## See Also

isat, isatloop

## Examples

```
###Repeated isat models using the Nile dataset
### where p-values are chosen such that the expected number of outliers under the null
### corresponds to 1, 2, ..., 20. Then computing the Outlier Scaling Tests:

#nile <- as.zoo(Nile)
#isat.nile.loop <- isatloop(y=nile)
#outlierscaletest(isat.nile.loop)
```

---

# outliertest *Jiao and Pretis Outlier Proportion and Count Tests*

---

## Description

Tests whether the proportion (or number) of outliers detected using impulse indicator saturation is different from the proportion (or number) of outliers expected under the null hypothesis of no outliers using the Jiao and Pretis (2019) proportion and count outlier tests.

## Usage

```
outliertest(x, noutl=NULL, t.pval=NULL, T=NULL,
m=1, infty=FALSE, alternative="two.sided")
```

## Arguments

| | |
|---|---|
| x | an isat object |
| noutl | integer, number of detected outliers if no isat object is provided i.e. x=NULL |
| t.pval | numeric, between 0 and 1. Selection p-value used in indicator saturation if no isat object is provided i.e. x=NULL |
| T | integer, sample sized used in indicator saturation if no isat object is provided i.e. x=NULL |
| m | integer, number of iterations in variance computation, default=1 |
| infty | logical, argument used for variance computation |
| alternative | "two-sided", "less", "greater", alternative hypothesis of outlier test. |

## Details

The function computes the estimated proportion of outliers (gauge) based on impulse indicator saturation and constructs the proportion and count outlier test statistics from Jiao and Pretis (2019). The null hypothesis is that the proportion (or count) of outliers is not different than the proportion (or count) of outliers detected under the null hypothesis of no outliers. The first test compares the estimated proportion of outliers scaled by its estimated variance against a standard normal distribution. The second test compares the number of outliers against a Poisson distribution. If an isat object is provided in x, then the function automatically extracts the detected impulses and computes the estimated outlier proportion. If no isat object is provided and x=NULL, then the tests can be conducted manually by providing the number of detected outliers (noutl), the sample size (T), and the chosen level of signficance used to detect outliers (t.pval).

## Value

Returns a list of two htest objects. The first providing the results of the test on the proportion of outliers against a standard normal distribution. The second providing the results on the number of outliers against the Poisson distribution.

## Author(s)

Xiyu Jiao, & Felix Pretis, http://www.felixpretis.org/

## References

Jiao, X. & Pretis, F. (2019). Testing the Presence of Outliers in Regression Models. Discussion Paper.

Pretis, F., Reade, J., & Sucarrat, G. (2018). Automated General-to-Specific (GETS) regression modeling and indicator saturation methods for the detection of outliers and structural breaks. Journal of Statistical Software, 86(3).

## See Also

isat

## Examples

```
###Testing the Presence of Outliers in the Nile Data
nile <- as.zoo(Nile)
isat.nile <- isat(nile, sis=FALSE, iis=TRUE, plot=TRUE, t.pval=0.1)
outliertest(isat.nile)

###Testing the number of outliers when the sample is T=200,
### with 7 detected outliers at t.pval=0.05 if no isat object is provided:
outliertest(x=NULL, noutl=7, t.pval=0.05, T=200)
```

---

# paths *Extraction functions for 'arx', 'gets' and 'isat' objects*

---

## Description

Extraction functions for objects of class 'arx', 'gets' and 'isat'

## Usage

```
paths(object, ...)
terminals(object, ...)
rsquared(object, adjusted=FALSE, ...)
```

## Arguments

| | |
|---|---|
| object | an object of class 'arx', 'gets' or 'isat' |
| adjusted | logical. If TRUE the adjusted R-squared is returned |
| ... | additional arguments |

## Details

paths and terminals can only be applied on objects of class 'gets' and 'isat'

## Value

| | |
|---|---|
| paths: | a list with the paths searched (each number refers to a regressor in the GUM) |
| terminals: | a list with the terminal models (each number refers to a regressor in the GUM) |
| rsquared: | a numeric, the R-squared of the regression, or adjusted R-squared if adjusted is set to TRUE |

## Author(s)

Genaro Sucarrat, http://www.sucarrat.net/

## See Also

getsm, getsm, getsv, isat

## Examples

```
##Simulate from an AR(1):
set.seed(123)
y <- arima.sim(list(ar=0.4), 50)

##Simulate four independent Gaussian regressors:
xregs <- matrix(rnorm(4*50), 50, 4)

##estimate an AR(2) with intercept and four conditioning
##regressors in the mean:
mymod <- arx(y, mc=TRUE, ar=1:2, mxreg=xregs)
rsquared(mymod)
rsquared(mymod, adjusted=TRUE)

##General-to-Specific (GETS) modelling of the mean:
meanmod <- getsm(mymod)
rsquared(meanmod)
rsquared(meanmod, adjusted=TRUE)

##extract the paths searched:
paths(meanmod)

##extract the terminal models:
terminals(meanmod)
```

---

`periodicdummies` *Make matrix of periodicity (e.g. seasonal) dummies*

---

## Description

Auxiliary function that generates periodicity dummies (e.g. seasonal dummies) for regular time series. The function is similar to, but more general than, the `seasonaldummy` function in the forecast package.

## Usage

```
periodicdummies(x, values=1)
```

## Arguments

| | |
|---|---|
| `x` | a regular time series (vector or matrix) |
| `values` | numeric of length 1 (default) or numeric vector of length equal to frequency(x) |

## Value

A matrix of class zoo with the periodicity dummies

### Author(s)

Genaro Sucarrat, http://www.sucarrat.net/

### See Also

is.regular, zooreg, zoo, ts

### Examples

```
##quarterly dummies:
x <- zooreg(rnorm(30), start=2000, frequency=4)
periodicdummies(x)

##monthly dummies:
y <- zooreg(rnorm(30), start=c(2000,1), frequency=12)
periodicdummies(y)
```

---

## predict.arx *Forecasting with 'arx' models*

---

### Description

Generate out-of-sample forecasts up to n steps ahead for objects of class arx. Optionally, quantiles of the forecasts are also returned if any of the arguments ci.levels or probs are specified. The forecasts, confidence intervals and quantiles are obtained via simulation. By default, 5000 simulations is used, but this can be changed via the n.sim argument. Also by default, the simulations uses a classical bootstrap to sample from the standardised residuals. To use an alternative set of standardised innovations, for example the standard normal, use the innov argument. If plot=TRUE, then a plot of the forecasts is created.

### Usage

```
  ## S3 method for class 'arx'
predict(object, spec=NULL, n.ahead=12, newmxreg=NULL,
    newvxreg=NULL, newindex=NULL, n.sim=5000, innov=NULL, probs=NULL,
    ci.levels=NULL, quantile.type=7, return=TRUE, verbose=FALSE,
    plot=NULL, plot.options=list(), ...)
```

### Arguments

| | |
|---|---|
| object | an object of class 'arx' |
| spec | NULL (default), "mean", "variance" or "both". If NULL, then it is automatically determined whether information pertaining to the mean or variance specification should be returned |
| n.ahead | integer that determines how many steps ahead predictions should be generated (the default is 12) |

| | |
|---|---|
| newmxreg | a matrix of n.ahead rows and NCOL(mxreg) columns with the out-of-sample values of the mxreg regressors |
| newvxreg | a matrix of n.ahead rows and NCOL(vxreg) columns with the out-of-sample values of the vxreg regressors |
| newindex | NULL (default) or the date-index for the zoo object returned by predict.arx. If NULL, then the function uses the in-sample index to generate the out-of-sample index |
| n.sim | integer, the number of replications used for the generation of the forecasts |
| innov | NULL (default) or a vector of length n.ahead * n.sim containing the standardised errors (that is, zero mean and unit variance) used for the forecast simulations. If NULL, then a classica bootstrap procedure is used to draw from the standardised in-sample residuals |
| probs | NULL (default) or a vector with the quantile-levels (values strictly between 0 and 1) of the forecast distribution. If NULL, then no quantiles are returned unless ci.levels is non-NULL |
| ci.levels | NULL (default) or a vector with the confidence levels (expressed as values strictly between 0 and 1) of the forecast distribution. The upper and lower values of the confidence interval(s) are returned as quantiles |
| quantile.type | an integer between 1 and 9 that selects which algorithm to be used in computing the quantiles, see the argument type in quantile |
| return | logical. If TRUE (default), then the out-of-sample predictions are returned. The value FALSE, which does not return the predictions, may be of interest if only a prediction plot is of interest |
| verbose | logical with default FALSE. If TRUE, then additional information (typically the quantiles and/or the simulated series) used in the generation of forecasts is returned. If FALSE, then only the forecasts are returned |
| plot | NULL (default) or logical. If NULL, then the value set by options$plot (see options) determines whether a plot is produced or not. If TRUE, then the out-of-sample forecasts are plotted. |
| plot.options | a list of options related to the plotting of forecasts, see 'Details' |
| ... | additional arguments |

## Details

The plot.options argument is a list that, optionally, can contain any of the following arguments:

- keep: integer greater than zero (the default is 12) that controls the number of in-sample actual values to plot
- line.at.origin: logical. If TRUE, then a vertical line is drawn at the forecast origin, that is, at the last in-sample observation
- start.at.origin: logical. If TRUE, then the drawing of the forecast line starts at the actual value of the forecast origin

- dot.at.origin: logical. If TRUE, then a dot is drawn at the forecast origin
- hlines: numeric vector that indicates where to draw grey horisontal grid lines
- col: numeric vector of length two that controls the colour of the plotted lines. The first value controls the colour of the forecasts and the fitted values, whereas the second controls the colour of the actual values
- lty: numeric vector of length two that controls the line type. The first value controls the line type of the forecast, whereas the second controls the line type of the actual and fitted values
- lwd: an integer that controls the width of the plotted lines (the default is 1)
- ylim: numeric vector of length two that contains the limits of the y-axis of the prediction plot
- ylab: a character that controls the text on the y-axis
- main: a character that controls the text in the overall title
- legend.text: a character vector of length two that controls how the forecast and actual lines should be named or referred to in the legend of the plot
- fitted: If TRUE, then the fitted values as well as actual values are plotted in-sample
- newmactual: numeric vector or NULL (default). Enables the plotting of actual values out-of-sample in the mean in addition to the forecasts
- newvactual: numeric vector or NULL (default). Enables the plotting of squared residuals ('actual values') out-of-sample in addition to the forecasts
- shades.of.grey: numeric vector of length length(ci.levels) that contains the shades of grey associated with the confidence intervals in the prediction plot. The shades can range from 100 (white) to 0 (black)

## Value

a vector of class zoo containing the out-of-sample forecasts, or a matrix of class zoo containing the out-of-sample forecasts together with prediction-quantiles, or NULL if return=FALSE

## Author(s)

Felix Pretis, http://www.felixpretis.org/
James Reade, https://sites.google.com/site/jjamesreade/
Genaro Sucarrat, http://www.sucarrat.net/

## See Also

arx

## Examples

```
##simulate from an AR(1):
set.seed(123)
y <- arima.sim(list(ar=0.4), 40)

##estimate AR(2) model with intercept:
mymod <- arx(y, mc=TRUE, ar=c(1,2))

##generate out-of-sample forecasts:
predict(mymod)

##same, but plot the predictions in addition:
#predict(mymod, plot=TRUE)

##same, but return also the quantiles of the confidence intervals:
#predict(mymod, ci.levels=c(0.50,0.90), plot=TRUE)

##same, but with non-default levels on the confidence intervals:
#predict(mymod, ci.levels=c(0.20,0.80, 0.99), plot=TRUE)

##same, but with more confidence intervals:
#predict(mymod, ci.levels=seq(0.20, 0.95, by=0.05), plot=TRUE)

##same, but with less rugged ci's (achieved by increasing n.sim):
##predict(mymod, ci.levels=seq(0.20, 0.95, by=0.05), n.sim=50000, plot=TRUE)

##same, but using standard normals (instead of bootstrap) in the simulations:
#n.sim <- 2000
#n.ahead <- 12 #the default on n.ahead
#predict(mymod, ci.levels=seq(0.20, 0.95, by=0.05), n.sim=n.sim,
#  innov=rnorm(n.ahead*n.sim), plot=TRUE)

##make x-regressors:
x <- matrix(rnorm(40*3), 40, 3)

##estimate AR(1) model with intercept and covariates:
mymod <- arx(y, mc=TRUE, ar=1, mxreg=x)

##predict up to 5 steps ahead, setting x's to 0 out-of-sample:
predict(mymod, n.ahead=5, newmxreg=matrix(0,5,NCOL(x)))

##same, but do also plot:
#predict(mymod, n.ahead=5, newmxreg=matrix(0,5,NCOL(x)),
#  plot=TRUE)

##estimate an AR(2) model w/intercept and a log-ARCH(1) specification
##on the variance:
#mymodel <- arx(y, mc=TRUE, ar=1:2, arch=1)
```

```
##generate forecasts of the conditional variances:
#predict(mymodel, spec="variance")

##same, but do also plot:
#predict(mymodel, spec="variance", plot=TRUE)

##illustrations of the usage of plot.options:
#mymodel <- arx(y, mc=TRUE, ar=1)
#predict(mymodel, plot=TRUE, plot.options=list(keep=1))
#predict(mymodel, plot=TRUE, plot.options=list(line.at.origin=TRUE))
#predict(mymodel, plot=TRUE, plot.options=list(start.at.origin=FALSE))
#predict(mymodel, plot=TRUE,
#  plot.options=list(start.at.origin=FALSE, fitted=TRUE))
#predict(mymodel, plot=TRUE, plot.options=list(dot.at.origin=FALSE))
#predict(mymodel, plot=TRUE, plot.options=list(hlines=c(-2,-1,0,1,2)))
#predict(mymodel, plot=TRUE, plot.options=list(col=c("darkred","green")))
#predict(mymodel, plot=TRUE, plot.options=list(lty=c(3,2)))
#predict(mymodel, plot=TRUE, plot.options=list(lwd=3))
#predict(mymodel, plot=TRUE, plot.options=list(ylim=c(-8,8)))
#predict(mymodel, plot=TRUE, plot.options=list(ylab="User-specified y-axis"))
#predict(mymodel, plot=TRUE,
#  plot.options=list(main="User-specified overall title"))
#predict(mymodel, plot=TRUE,
#  plot.options=list(legend.text=c("User-specified 1","User-specified 2")))
#predict(mymodel, plot=TRUE, plot.options=list(fitted=TRUE))
#predict(mymodel, plot=TRUE, plot.options=list(newmactual=rep(0,6)))
#predict(mymodel, plot=TRUE, plot.options=list(shades.of.grey=c(95,50)))
#predict(mymodel, plot=TRUE, plot.options=list(shades.of.grey=c(50,95))) #invert shades
```

---

`printtex` *Generate LaTeX code of an estimation result*

---

## Description

Convenience functions that generates LaTeX-code of an estimation result in equation-form. `printtex` can, in principle, be applied to any object for which `coef`, `vcov` and `logLik` methods exist. Note: The generated LaTeX-code contains an `eqnarray` environment, which requires that the `amsmath` package is loaded in the preamble of the LaTeX document.

## Usage

```
  printtex(x, fitted.name=NULL, xreg.names=NULL, digits=4,
      intercept=TRUE, gof=TRUE, diagnostics=TRUE, nonumber=FALSE,
      nobs="T")
  ## S3 method for class 'arx'
toLatex(object, ...)
  ## S3 method for class 'gets'
toLatex(object, ...)
```

## Arguments

| | |
|---|---|
| x | an estimation result, e.g. arx, gets or isat object |
| object | an estimation result of class arx or gets |
| fitted.name | NULL or a user-specified name of left-hand side variable |
| xreg.names | NULL or a user-specified character vector with the names of regressors |
| digits | integer, the number of digits to be printed |
| intercept | logical or numeric. The argument determines whether one of the regressors is an intercept or not, or its location. If TRUE, then the intercept is assumed to be located at coef(x)[1], and hence the regressor-name of location 1 is excluded from the print. If FALSE, then it is assumed that there is no intercept among the regressors. If numeric, then it is assumed that the regressors contain an intercept at the location equal to the numeric value |
| gof | logical, whether to include goodness-of-fit in the print |
| diagnostics | logical, whether to include diagnostics in the print |
| nonumber | logical, whether to remove or not (default) the equation-numbering |
| nobs | character, the notation to use to denote the number of observations |
| ... | arguments passed on to printtex |

## Details

Currently, toLatex.arx and toLatex.gets are simply wrappers to printtex

## Value

LaTeX code of an estimation result

## Author(s)

Genaro Sucarrat, http://www.sucarrat.net/

## See Also

arx, getsm, getsv, isat

## Examples

```
##simulate random variates, estimate model:
y <- rnorm(30)
mX <- matrix(rnorm(30*2), 30, 2)
mymod <- arx(y, mc=TRUE, ar=1:3, mxreg=mX)

##print latex code of estimation result:
printtex(mymod)

##add intercept, at the end, to regressor matrix:
mX <- cbind(mX,1)
colnames(mX) <- c("xreg1", "xreg2", "intercept")
```

```
mymod <- arx(y, mxreg=mX)

##set intercept location to 3:
printtex(mymod, intercept=3)
```

---

`recursive` *Recursive estimation*

---

## Description

Recursive estimation of coefficients and standard errors

## Usage

```
recursive(object, spec="mean", std.errors=TRUE, from=40, tol=1e-07,
  LAPACK=FALSE, plot=TRUE, return=TRUE)
```

## Arguments

| | |
|---|---|
| `object` | an `arx`, `gets` or `isat` object |
| `spec` | 'mean' or 'variance'. If 'mean' (default), the the recursive estimates of the mean-equation are estimated |
| `std.errors` | logical. If TRUE (default), then the coefficient standard errors are also computed |
| `from` | integer. The starting point of the recursion |
| `tol` | numeric. The tolerance for linear dependency among regressors |
| `LAPACK` | logical, TRUE or FALSE (default). If true use LAPACK otherwise use LINPACK, see `qr` function |
| `plot` | NULL or logical. If TRUE, then the recursive coefficient estimates are plotted. If NULL (default), then the value set by `options` determines whether a plot is produced or not. |
| `return` | logical. If TRUE (default), then the recursive estimates are returned in a list |

## Value

If `return=TRUE`, then a `list` is returned with the following components:

| | |
|---|---|
| `estimates` | a `zoo` matrix with the recursive estimates |
| `standard.errors` | a `zoo` matrix with the standard errors |

## Author(s)

Genaro Sucarrat, http://www.sucarrat.net/

## See Also

ols, qr, solve.qr

## Examples

```
##generate random variates, estimate model:
y <- rnorm(100)
mX <- matrix(rnorm(4*100), 100, 4)
mymodel <- arx(y, mc=TRUE, mxreg=mX)

##compute recursive estimates and plot them:
recursive(mymodel)
```

---

regressorsMean *Create the regressors of the mean equation*

---

## Description

The function generates the regressors of the mean equation in an arx model. The returned value is a matrix with the regressors and, by default, the regressand in column one. By default, observations (rows) with missing values are removed in the beginning and the end with na.trim, and the returned matrix is a zoo object.

## Usage

```
regressorsMean(y, mc = FALSE, ar = NULL, ewma = NULL, mxreg = NULL,
  return.regressand = TRUE, return.as.zoo = TRUE, na.trim = TRUE,
  na.omit=FALSE)
```

## Arguments

| | |
|---|---|
| y | numeric vector, time-series or zoo object. |
| mc | logical. TRUE includes an intercept, whereas FALSE (default) does not. |
| ar | either NULL (default) or an integer vector, say, c(2,4) or 1:4 with the AR-lags to include in the mean specification. If NULL, then no lags are included. |
| ewma | either NULL (default) or a list with arguments sent to the eqwma function. In the latter case a lagged moving average of y is included as a regressor. |
| mxreg | either NULL (default), numeric vector or matrix, say, a zoo object, or data.frame containing conditioning variables (covariates). Note that, if both y and mxreg are zoo objects, then their samples are matched. |
| return.regressand | logical. TRUE, the default, includes the regressand as column one in the returned matrix. |
| return.as.zoo | TRUE, the default, returns the matrix as a zoo object. |
| na.trim | TRUE, the default, removes observations with NA-values in the beginning and the end with na.trim. |
| na.omit | TRUE, the non-default, removes observations with NA-values, not necessarily in the beginning or in the end, with na.omit. |

## Value

A matrix, by default of class zoo, with the regressand as column one (the default).

## Author(s)

Genaro Sucarrat, http://www.sucarrat.net/

## References

Pretis, Felix, Reade, James and Sucarrat, Genaro (2018): 'Automated General-to-Specific (GETS) Regression Modeling and Indicator Saturation for Outliers and Structural Breaks'. Journal of Statistical Software 86, Number 3, pp. 1-44. DOI: https://www.jstatsoft.org/article/view/v086i03

## See Also

arx, isat, regressorsVariance, zoo, eqwma, na.trim and na.trim.

## Examples

```
##generate some data:
y <- rnorm(10) #regressand
x <- matrix(rnorm(10*5), 10, 5) #regressors

##create regressors (examples):
regressorsMean(y, mxreg=x)
regressorsMean(y, mxreg=x, return.regressand=FALSE)
regressorsMean(y, mc=TRUE, ar=1:3, mxreg=x)
regressorsMean(log(y^2), mc=TRUE, ar=c(2,4))

##let y and x be time-series:
y <- ts(y, frequency=4, end=c(2018,4))
x <- ts(x, frequency=4, end=c(2018,4))
regressorsMean(y, mxreg=x)
regressorsMean(y, mc=TRUE, ar=1:3, mxreg=x)
regressorsMean(log(y^2), mc=TRUE, ar=c(2,4))

##missing values (NA):
y[1] <- NA
x[10,3] <- NA
regressorsMean(y, mxreg=x)
regressorsMean(y, mxreg=x, na.trim=FALSE)
```

regressorsVariance *Create the regressors of the variance equation*

## Description

The function generates the regressors of the log-variance equation in an arx model. The returned value is a matrix with the regressors and, by default, the regressand in column one. By default, observations (rows) with missing values are removed in the beginning and the end with na.trim, and the returned matrix is a zoo object.

## Usage

```
regressorsVariance(e, vc = TRUE, arch = NULL, asym = NULL,
  log.ewma = NULL, vxreg = NULL, zero.adj = 0.1, vc.adj = TRUE,
  return.regressand = TRUE, return.as.zoo = TRUE, na.trim = TRUE,
  na.omit = FALSE)
```

## Arguments

| | |
|---|---|
| e | numeric vector, time-series or zoo object. |
| vc | logical. TRUE includes an intercept in the log-variance specification, whereas FALSE (default) does not. If the log-variance specification contains any other item but the log-variance intercept, then vc is set to TRUE |
| arch | either NULL (default) or an integer vector, say, c(1,3) or 2:5. The log-ARCH lags to include in the log-variance specification |
| asym | either NULL (default) or an integer vector, say, c(1) or 1:3. The asymmetry (i.e. 'leverage') terms to include in the log-variance specification |
| log.ewma | either NULL (default) or a vector of the lengths of the volatility proxies, see leqwma |
| vxreg | either NULL (default) or a numeric vector or matrix, say, a zoo object, of conditioning variables. If both y and mxreg are zoo objects, then their samples are chosen to match |
| zero.adj | numeric value between 0 and 1. The quantile adjustment for zero values. The default 0.1 means the zero residuals are replaced by the 10 percent quantile of the absolute residuals before taking the logarithm |
| vc.adj | logical. If TRUE (default), then the log-variance intercept is adjusted by the estimate of E[ln(z^2)], where z is the standardised error. This adjustment is needed for the conditional scale to be equal to the conditional standard deviation. If FALSE, then the log-variance intercept is not adjusted |
| return.regressand | logical. TRUE, the default, includes the regressand as column one in the returned matrix. |
| return.as.zoo | TRUE, the default, returns the matrix as a zoo object. |

| | |
|---|---|
| na.trim | TRUE, the default, removes observations with NA-values in the beginning and the end with na.trim. |
| na.omit | TRUE, the non-default, removes observations with NA-values, not necessarily in the beginning or in the end, with na.omit. |

## Value

A matrix, by default of class zoo, with the regressand as column one (the default).

## Author(s)

Genaro Sucarrat, http://www.sucarrat.net/

## References

Pretis, Felix, Reade, James and Sucarrat, Genaro (2018): 'Automated General-to-Specific (GETS) Regression Modeling and Indicator Saturation for Outliers and Structural Breaks'. Journal of Statistical Software 86, Number 3, pp. 1-44. DOI: https://www.jstatsoft.org/article/view/v086i03

Sucarrat, Genaro and Escribano, Alvaro (2012): 'Automated Financial Model Selection: General-to-Specific Modelling of the Mean and Volatility Specifications', Oxford Bulletin of Economics and Statistics 74, Issue 5 (October), pp. 716-735

## See Also

regressorsMean, arx, zoo, leqwma, na.trim and na.omit.

## Examples

```
##generate some data:
eps <- rnorm(10) #error term
x <- matrix(rnorm(10*5), 10, 5) #regressors

##create regressors (examples):
regressorsVariance(eps, vxreg=x)
regressorsVariance(eps, vxreg=x, return.regressand=FALSE)
regressorsVariance(eps, arch=1:3, vxreg=x)
regressorsVariance(eps, arch=1:2, asym=1, vxreg=x)
regressorsVariance(eps, arch=1:2, asym=1, log.ewma=5)

##let eps and x be time-series:
eps <- ts(eps, frequency=4, end=c(2018,4))
x <- ts(x, frequency=4, end=c(2018,4))
regressorsVariance(eps, vxreg=x)
regressorsVariance(eps, arch=1:3, vxreg=x)
regressorsVariance(eps, arch=1:2, asym=1, vxreg=x)
regressorsVariance(eps, arch=1:2, asym=1, log.ewma=5)
```

so2data *UK SO2 Data*

## Description

UK Annual Total Anthropogenic Sulphur Dioxide (SO2) Emissions 1946-2005.

## Usage

```
data("so2data")
```

## Format

A data frame with 60 observations on the following 4 variables.

`year` Year of observation

`uk_tot_so2` UK annual total anthropogenic SO2 emissions in gigagrams

`Luk_tot_so2` Log of UK annual total anthropogenic SO2 emissions

`DLuk_tot_so2` First difference of Log UK annual total anthropogenic SO2 emissions

## Details

Data reports the total estimated anthropogenic SO2 emissions aggregated over coal, petroleum, biomass combustion, smelting, fuel processing, and other processes.

## Source

Smith, SJ, J van Aardenne, Z Klimont, RJ Andres, A Volke, and S Delgado Arias. (2011). Anthropogenic Sulfur Dioxide Emissions, 1850-2005: National and Regional Data Set by Source Category, Version 2.86. Data distributed by the NASA Socioeconomic Data and Applications Center (SEDAC), CIESIN, Columbia University, Palisades, New York. Available at

http://sedac.ciesin.columbia.edu/data/set/haso2-anthro-sulfur-dioxide-emissions-1850-2005-v2-86

## References

Pretis, Felix, Reade, James and Sucarrat, Genaro (2018): 'Automated General-to-Specific (GETS) Regression Modeling and Indicator Saturation for Outliers and Structural Breaks'. Journal of Statistical Software 86, Number 3, pp. 1-44

Smith, SJ, J van Aardenne, Z Klimont, RJ Andres, A Volke, and S Delgado Arias. (2011). Anthropogenic Sulfur Dioxide Emissions: 1850-2005, Atmospheric Chemistry and Physics, 11:1101-1116.

## Examples

```
data(so2data)

##create annual zoo object:
newso2data<- zooreg(so2data[,-1], start=1946, frequency=1)

##plot UK annual total anthropogenic SO2 emissions:
plot(newso2data$uk_tot_so2)
```

---

`sp500data` *Daily Standard and Poor's 500 index data*

---

## Description

Daily Standard and Poor's 500 (SP500) index data from 3 January 1950 to 8 March 2016.

## Usage

```
data("sp500data")
```

## Format

A data frame with 16652 observations on the following 7 variables:

`Date` the dates

`Open` the opening values of the index

`High` the daily maximum value of the index

`Low` the daily minimum value of the index

`Close` the closing values of the index

`Volume` the traded volume

`Adj.Close` the adjusted closing values of the index

## Source

Yahoo Finance, retrieved 9 March 2016

## References

Pretis, Felix, Reade, James and Sucarrat, Genaro (2018): 'Automated General-to-Specific (GETS) Regression Modeling and Indicator Saturation for Outliers and Structural Breaks'. Journal of Statistical Software 86, Number 3, pp. 1-44

## Examples

```
data(sp500data)
sp500data <- zoo(sp500data[, -1], order.by = as.Date(sp500data[, "Date"]))
plot(window(sp500data, start = as.Date("2000-01-03")))
```

vargaugeiis *Variance of the Impulse Indicator Saturation Gauge*

## Description

Computes the variance of the gauge (false-positive rate of outliers under the null of no outliers) in impulse indicator saturation based on Jiao and Pretis (2019).

## Usage

```
vargaugeiis(t.pval, T, infty=FALSE, m=1)
```

## Arguments

| | |
|---|---|
| t.pval | numeric, between 0 and 1. Selection p-value used in indicator saturation. |
| T | integer, sample sized used in indicator saturation. |
| m | integer, number of iterations in variance computation, default=1 |
| infty | logical, argument used for variance computation |

## Details

The function computes the variance of the Gauge (false-positive rate of outliers in impulse indicator saturation) for a given level of significance of selection (t.pval) and sample size (T) based on Jiao and Pretis (2019). This is an auxilliary function used within the outliertest function.

## Value

Returns a dataframe of the variance and standard deviation of the gauge, as well the asymptotic variance and standard deviation.

## Author(s)

Felix Pretis, http://www.felixpretis.org/

## References

Jiao, X. & Pretis, F. (2019). Testing the Presence of Outliers in Regression Models. Discussion Paper.

Pretis, F., Reade, J., & Sucarrat, G. (2018). Automated General-to-Specific (GETS) regression modeling and indicator saturation methods for the detection of outliers and structural breaks. Journal of Statistical Software, 86(3).

## See Also

isat, outliertest

## Examples

```
###Computing the variance of the gauge under the null for a sample of T=200 observations:
vargaugeiis(t.pval=0.05, T=200, infty=FALSE, m=1)
```

# Index