Package ‘sandwich’

March 26, 2015

Version 2.3-3
Date 2015-03-26
Title Robust Covariance Matrix Estimators
Description
Model-robust standard error estimators for cross-sectional, time series, and longitudinal data.

Depends R (>= 2.0.0)
Imports stats, zoo
Suggests car, lmtest, strucchange, AER, survival, MASS, scatterplot3d
License GPL-2 | GPL-3
Author Thomas Lumley [aut],
Achim Zeileis [aut, cre]
Maintainer Achim Zeileis <Achim.Zeileis@R-project.org>
NeedsCompilation no
Repository CRAN
Date/Publication 2015-03-26 19:56:56

R topics documented:
bread ......................................................... 2
estfun ..................................................... 3
Investment ............................................... 4
isoacf ...................................................... 5
kweights .................................................. 6
lrvar ....................................................... 7
meat ....................................................... 8
NeweyWest ............................................... 10
PublicSchools ........................................... 12
sandwich .................................................. 13
vcovHAC ................................................. 15
vcovHC ................................................... 17
vcovOPG .................................................. 19
weightsAndrews ....................................... 20
weightsLumley ......................................... 23
Description

Generic function for extracting an estimator for the bread of sandwiches.

Usage

```r
bread(x, ...)```

Arguments

- `x` a fitted model object.
- `...` arguments passed to methods.

Value

A matrix containing an estimator for the expectation of the negative derivative of the estimating functions, usually the Hessian. Typically, this should be an $k \times k$ matrix corresponding to $k$ parameters. The rows and columns should be named as in `coef` or `terms`, respectively.

References


See Also

- `lm`, `glm`

Examples

```r
## linear regression
x <- sin(1:10)
y <- rnorm(10)
fm <- lm(y ~ x)

## bread: n * (x'x)^{-1}
bread(fm)
solve(crossprod(cbind(1, x))) * 10
```
estfun

Extract Empirical Estimating Functions

Description

Generic function for extracting the empirical estimating functions of a fitted model.

Usage

estfun(x, ...)

Arguments

x a fitted model object.

... arguments passed to methods.

Value

A matrix containing the empirical estimating functions. Typically, this should be an \( n \times k \) matrix corresponding to \( n \) observations and \( k \) parameters. The columns should be named as in `coef` or `terms`, respectively.

The estimating function (or score function) for a model is the derivative of the objective function with respect to the parameter vector. The empirical estimating functions is the evaluation of the estimating function at the observed data (\( n \) observations) and the estimated parameters (of dimension \( k \)).

References


See Also

`lm`, `glm`

Examples

```r
## linear regression
x <- sin(1:10)
y <- rnorm(10)
fm <- lm(y ~ x)

## estimating function: \((y - x'\beta) \times x\)
estfun(fm)
residuals(fm) * cbind(1, x)
```
**US Investment Data**

**Description**

US data for fitting an investment equation.

**Usage**

```r
data(Investment)
```

**Format**

An annual time series from 1963 to 1982 with 7 variables.

- **GNP** nominal gross national product (in billion USD),
- **Investment** nominal gross private domestic investment (in billion USD),
- **Price** price index, implicit price deflator for GNP,
- **Interest** interest rate, average yearly discount rate charged by the New York Federal Reserve Bank,
- **RealGNP** real GNP (= GNP/Price),
- **RealInv** real investment (= Investment/Price),
- **RealInt** approximation to the real interest rate (= Interest - 100 * diff(Price)/Price).

**Source**

Table 15.1 in Greene (1993)

**References**


**Examples**

```r
## Willam H. Greene, Econometric Analysis, 2nd Ed.
## Chapter 15
## load data set, p. 411, Table 15.1
data(Investment)

## fit linear model, p. 412, Table 15.2
fm <- lm(RealInv ~ RealGNP + RealInt, data = Investment)
summary(fm)

## visualize residuals, p. 412, Figure 15.1
plot(ts(residuals(fm), start = 1964),
```
isoacf

Isotonic Autocorrelation Function

Description

Autocorrelation function (forced to be decreasing by isotonic regression).

Usage

isoacf(x, lagmax = NULL, weave1 = FALSE)

Arguments

x numeric vector.

lagmax numeric. The maximal lag of the autocorrelations.

weave1 logical. If set to TRUE isoacf uses the acf.R and pava.blocks function from the original weave package, otherwise R’s own acf and isoreg functions are used.
Details

isoacf computes the autocorrelation function (ACF) of x enforcing the ACF to be decreasing by isotonic regression. See also Robertson et al. (1988).

Value

isoacf returns a numeric vector containing the ACF.

References


See Also

weave, weightsLumley

Examples

x <- filter(rnorm(100), 0.9, "recursive")
isoacf(x)
acf(x, plot = FALSE)$acf

kweights

Kernel Weights

Description


Usage

kweights(x, kernel = c("Truncated", "Bartlett", "Parzen", "Tukey-Hanning", "Quadratic Spectral"), normalize = FALSE)

Arguments

x numeric.

kernel a character specifying the kernel used. All kernels used are described in Andrews (1991).

normalize logical. If set to TRUE the kernels are normalized as described in Andrews (1991).
**lrvar**

**Long-Run Variance of the Mean**

Convenience function for computing the long-run variance (matrix) of a (possibly multivariate) series of observations.

**Usage**

```r
lrvar(x, type = c("Andrews", "Newey-West"), prewhite = TRUE, adjust = TRUE, ...)
```

**Arguments**

- `x` numeric vector, matrix, or time series.
- `type` character specifying the type of estimator, i.e., whether `kernHAC` for the Andrews quadratic spectral kernel HAC estimator is used or `NeweyWest` for the Newey-West Bartlett HAC estimator.
- `prewhite` logical or integer. Should the series be prewhitened? Passed to `kernHAC` or `NeweyWest`.
- `adjust` logical. Should a finite sample adjustment be made? Passed to `kernHAC` or `NeweyWest`.
- `...` further arguments passed on to `kernHAC` or `NeweyWest`.

**Value**

Value of the kernel function at \( x \).

**References**


**See Also**

`kernHAC`, `weightsAndrews`

**Examples**

```r
curve(kweights(x, kernel = "Quadratic", normalize = TRUE),
     from = 0, to = 3.2, xlab = "x", ylab = "k(x)")
curve(kweights(x, kernel = "Bartlett", normalize = TRUE),
     from = 0, to = 3.2, col = 2, add = TRUE)
curve(kweights(x, kernel = "Parzen", normalize = TRUE),
     from = 0, to = 3.2, col = 3, add = TRUE)
curve(kweights(x, kernel = "Tukey", normalize = TRUE),
     from = 0, to = 3.2, col = 4, add = TRUE)
curve(kweights(x, kernel = "Truncated", normalize = TRUE),
     from = 0, to = 3.2, col = 5, add = TRUE)
```
Details

lrvar is a simple wrapper function for computing the long-run variance (matrix) of a (possibly multivariate) series x. First, this simply fits a linear regression model x ~ 1 by lm. Second, the corresponding variance of the mean(s) is estimated either by kernHAC (Andrews quadratic spectral kernel HAC estimator) or by NeweyWest (Newey-West Bartlett HAC estimator).

Value

For a univariate series x a scalar variance is computed. For a multivariate series x the covariance matrix is computed.

See Also

kernHAC, NeweyWest, vcovHAC

Examples

```r
set.seed(1)
## iid series (with variance of mean 1/n)
## and Andrews kernel HAC (with prewhitening)
x <- rnorm(100)
lrvar(x)

## analogous multivariate case with Newey-West estimator (without prewhitening)
y <- matrix(rnorm(200), ncol = 2)
lrvar(y, type = "Newey-West", prewhite = FALSE)

## AR(1) series with autocorrelation 0.9
z <- filter(rnorm(100), 0.9, method = "recursive")
lrvar(z)
```

---

meat  
A Simple Meat Matrix Estimator

Description

Estimating the variance of the estimating functions of a regression model by cross products of the empirical estimating functions.

Usage

```r
meat(x, adjust = FALSE, ...)
```
Arguments

- **x**: a fitted model object.
- **adjust**: logical. Should a finite sample adjustment be made? This amounts to multiplication with \( \frac{n}{n - k} \)

where

\( n \) is the number of observations and

\( k \) the number of estimated parameters.

... arguments passed to the `estfun` function.

Details

For some theoretical background along with implementation details see Zeileis (2006).

Value

A

\[ k \times k \]

matrix corresponding containing the scaled cross products of the empirical estimating functions.

References


See Also

`sandwich`, `bread`, `estfun`

Examples

```r
x <- sin(1:10)
y <- rnorm(10)
fm <- lm(y ~ x)

meat(fm)
meathc(fm, type = "HC")
meathAC(fm)
```
**NeweyWest**

*Newey-West HAC Covariance Matrix Estimation*

## Description


## Usage

NeweyWest(x, lag = NULL, order.by = NULL, prewhite = TRUE, adjust = FALSE, diagnostics = FALSE, sandwich = TRUE, ar.method = "ols", data = list(), verbose = FALSE)

bwNeweyWest(x, order.by = NULL, kernel = c("Bartlett", "Parzen", "Quadratic Spectral", "Truncated", "Tukey-Hanning"), weights = NULL, prewhite = 1, ar.method = "ols", data = list(), ...)

## Arguments

- **x**
  - a fitted model object.

- **lag**
  - integer specifying the maximum lag with positive weight for the Newey-West estimator. If set to NULL floor(bwNeweyWest(x, ...)) is used.

- **order.by**
  - Either a vector z or a formula with a single explanatory variable like ~ z. The observations in the model are ordered by the size of z. If set to NULL (the default) the observations are assumed to be ordered (e.g., a time series).

- **prewhite**
  - logical or integer. Should the estimating functions be prewhitened? If TRUE or greater than 0 a VAR model of order as.integer(prewhite) is fitted via ar with method "ols" and demean = FALSE. The default is to use VAR(1) prewhitening.

- **kernel**
  - a character specifying the kernel used. All kernels used are described in Andrews (1991). bwNeweyWest can only compute bandwidths for "Bartlett", "Parzen" and "Quadratic Spectral".

- **adjust**
  - logical. Should a finite sample adjustment be made? This amounts to multiplication with \( n/(n-k) \) where \( n \) is the number of observations and \( k \) the number of estimated parameters.

- **diagnostics**
  - logical. Should additional model diagnostics be returned? See vcovHAC for details.

- **sandwich**
  - logical. Should the sandwich estimator be computed? If set to FALSE only the middle matrix is returned.

- **ar.method**
  - character. The method argument passed to ar for prewhitening (only, not for bandwidth selection).

- **data**
  - an optional data frame containing the variables in the order.by model. By default the variables are taken from the environment which the function is called from.
verbose logical. Should the lag truncation parameter used be printed?
weights numeric. A vector of weights used for weighting the estimated coefficients of
the approximation model (as specified by approx). By default all weights are 1
except that for the intercept term (if there is more than one variable).
...
currently not used.

Details

`NeweyWest` is a convenience interface to `vcovHAC` using Bartlett kernel weights as described in
Newey & West (1987, 1994). The automatic bandwidth selection procedure described in Newey &
West (1994) is used as the default and can also be supplied to `kernHAC` for the Parzen and quadratic
spectral kernel. It is implemented in `bwNeweyWest` which does not truncate its results - if the results
for the Parzen and Bartlett kernels should be truncated, this has to be applied afterwards. For Bartlett
weights this is implemented in `NeweyWest`.

To obtain the estimator described in Newey & West (1987), prewhitening has to be suppressed.

Value

`NeweyWest` returns the same type of object as `vcovHAC` which is typically just the covariance matrix.
`bwNeweyWest` returns the selected bandwidth parameter.

References

Andrews DWK (1991), Heteroskedasticity and Autocorrelation Consistent Covariance Matrix Estimation.
Econometrica, 59, 817–858.

Newey WK & West KD (1987), A Simple, Positive Semi-Definite, Heteroskedasticity and Auto-
correlation Consistent Covariance Matrix. Econometrica, 55, 703–708.

of Economic Studies, 61, 631–653.


See Also

`vcovHAC`, `weightsAndrews`, `kernHAC`

Examples

```r
## fit investment equation
data(Investment)
fm <- lm(RealInv ~ RealGNP + RealInt, data = Investment)

## Newey & West (1994) compute this type of estimator
NeweyWest(fm)

## The Newey & West (1987) estimator requires specification
## of the lag and suppression of prewhitening
NeweyWest(fm, lag = 4, prewhite = FALSE)
```
# bwNeweyWest() can also be passed to kernHAC(), e.g.
# for the quadratic spectral kernel
kernHAC(fm, bw = bwNeweyWest)

<table>
<thead>
<tr>
<th>PublicSchools</th>
<th>US Expenditures for Public Schools</th>
</tr>
</thead>
</table>

## Description
Per capita expenditure on public schools and per capita income by state in 1979.

## Usage
data(PublicSchools)

## Format
A data frame containing 51 observations of 2 variables.

- **Expenditure**: per capita expenditure on public schools,
- **Income**: per capita income.

## Source
Table 14.1 in Greene (1993)

## References

## Examples
```r
## William H. Greene, Econometric Analysis, 2nd Ed.
## Chapter 14
## load data set, p. 385, Table 14.1
data(PublicSchools)

## omit NA in Wisconsin and scale income
ps <- na.omit(PublicSchools)
p$Income <- ps$Income * 0.0001

## fit quadratic regression, p. 385, Table 14.2
```
```r
fmq <- lm(Expenditure ~ Income + I(Income^2), data = ps)
summary(fmq)

## compare standard and HC0 standard errors
## p. 391, Table 14.3
library(sandwich)
coef(fmq)
sqrt(diag(vcovHC(fmq, type = "const")))
sqrt(diag(vcovHC(fmq, type = "HC0")))

if(require(lmtest)) {
## compare t ratio
coeftest(fmq, vcov = vcovHC(fmq, type = "HC0"))

## White test, p. 393, Example 14.5
wt <- lm(residuals(fmq)^2 ~ poly(Income, 4), data = ps)
wt.stat <- summary(wt)$r.squared * nrow(ps)
c(wt.stat, pchisq(wt.stat, df = 3, lower = FALSE))

## Bresch-Pagan test, p. 395, Example 14.7
bptest(fmq, studentize = FALSE)
bptest(fmq)

## Francisco Cribari-Neto, Asymptotic Inference, CSDA 45
## quasi z-tests, p. 229, Table 8
## with Alaska
coeftest(fmq, df = Inf)[3,4]
coeftest(fmq, df = Inf, vcov = vcovHC(fmq, type = "HC0"))[3,4]
coeftest(fmq, df = Inf, vcov = vcovHC(fmq, type = "HC3"))[3,4]
coeftest(fmq, df = Inf, vcov = vcovHC(fmq, type = "HC4"))[3,4]
## without Alaska (observation 2)
fmq1 <- lm(Expenditure ~ Income + I(Income^2), data = ps[-2,1])
coeftest(fmq1, df = Inf)[3,4]
coeftest(fmq1, df = Inf, vcov = vcovHC(fmq1, type = "HC0"))[3,4]
coeftest(fmq1, df = Inf, vcov = vcovHC(fmq1, type = "HC3"))[3,4]
coeftest(fmq1, df = Inf, vcov = vcovHC(fmq1, type = "HC4"))[3,4]
}

## visualization, p. 230, Figure 1
plot(Expenditure ~ Income, data = ps,
xlab = "per capita income",
     ylab = "per capita spending on public schools")
inc <- seq(0.5, 1.2, by = 0.001)
lines(inc, predict(fmq, data.frame(Income = inc)), col = 4)
fml <- lm(Expenditure ~ Income, data = ps)
abline(fml)
text(ps[2,2], ps[2,1], rownames(ps)[2], pos = 2)
```

Description

Constructing sandwich covariance matrix estimators by multiplying bread and meat matrices.

Usage

```r
sandwich(x, bread. = bread, meat. = meat, ...)
```

Arguments

- **x**: a fitted model object.
- **bread.**: either a bread matrix or a function for computing this via `bread.(x)`.
- **meat.**: either a bread matrix or a function for computing this via `meat.(x, ...)`.
- **...**: arguments passed to the `meat` function.

Details

`sandwich` is a simple convenience function that takes a bread matrix (i.e., estimator of the expectation of the negative derivative of the estimating functions) and a meat matrix (i.e., estimator of the variance of the estimating functions) and multiplies them to a sandwich with meat between two slices of bread. By default `bread` and `meat` are called.

Some theoretical background along with implementation details is given in Zeileis (2006).

Value

A matrix containing the sandwich covariance matrix estimate. Typically, this should be an $k \times k$ matrix corresponding to $k$ parameters.

References


See Also

`bread`, `meat`, `meathc`, `meathac`

Examples

```r
x <- sin(1:10)
y <- rnorm(10)
fm <- lm(y ~ x)

sandwich(fm)
vcovHC(fm, type = "HC")
```
Description

Heteroskedasticity and autocorrelation consistent (HAC) estimation of the covariance matrix of the coefficient estimates in a (generalized) linear regression model.

Usage

```r
vcovHAC(x, ...) # Default S3 method:
vcovHAC(x, order.by = NULL, prewhite = FALSE, weights = weightsAndrews,
   adjust = TRUE, diagnostics = FALSE, sandwich = TRUE, ar.method = "ols",
   data = list(), ...)

meatHAC(x, order.by = NULL, prewhite = FALSE, weights = weightsAndrews,
   adjust = TRUE, diagnostics = FALSE, ar.method = "ols", data = list())
```

Arguments

- **x**: a fitted model object.
- **order.by**: Either a vector \( z \) or a formula with a single explanatory variable like \( \sim z \). The observations in the model are ordered by the size of \( z \). If set to NULL (the default) the observations are assumed to be ordered (e.g., a time series).
- **prewhite**: logical or integer. Should the estimating functions be prewhitened? If TRUE or greater than 0 a VAR model of order as.integer(prewhite) is fitted via `ar` with method "ols" and demean = FALSE.
- **weights**: Either a vector of weights for the autocovariances or a function to compute these weights based on \( x \), order.by, prewhite, ar.method and data. If weights is a function it has to take these arguments. See also details.
- **adjust**: logical. Should a finite sample adjustment be made? This amounts to multiplication with \( n/(n-k) \) where \( n \) is the number of observations and \( k \) the number of estimated parameters.
- **diagnostics**: logical. Should additional model diagnostics be returned? See below for details.
- **sandwich**: logical. Should the sandwich estimator be computed? If set to FALSE only the meat matrix is returned.
- **ar.method**: character. The method argument passed to `ar` for prewhitening.
- **data**: an optional data frame containing the variables in the order.by model. By default the variables are taken from the environment which `vcovHAC` is called from.
- **...**: arguments passed to `sandwich`. 
Details

The function `meatHAC` is the real work horse for estimating the meat of HAC sandwich estimators – the default `vcovHAC` method is a wrapper calling `sandwich` and `bread`. See Zeileis (2006) for more implementation details. The theoretical background, exemplified for the linear regression model, is described in Zeileis (2004).

Both functions construct weighted information sandwich variance estimators for parametric models fitted to time series data. These are basically constructed from weighted sums of autocovariances of the estimating functions (as extracted by `estfun`). The crucial step is the specification of weights: the user can either supply `vcovHAC` with some vector of weights or with a function that computes these weights adaptively (based on the arguments `x`, `order.by`, `prewhite` and `data`). Two functions for adaptively choosing weights are implemented in `weightsAndrews` implementing the results of Andrews (1991) and in `weightsLumley` implementing the results of Lumley (1999). The functions `kernHAC` and `weave` respectively are to more convenient interfaces for `vcovHAC` with these functions.

Prewhitening based on VAR approximations is described as suggested in Andrews & Monahan (1992).

The covariance matrix estimators have been improved by the addition of a bias correction and an approximate denominator degrees of freedom for test and confidence interval construction. See Lumley & Heagerty (1999) for details.

Value

A matrix containing the covariance matrix estimate. If diagnostics was set to `TRUE` this has an attribute "diagnostics" which is a list with

- `bias.correction` multiplicitive bias correction
- `df` Approximate denominator degrees of freedom

References


See Also

`weightsLumley`, `weightsAndrews`, `weave`, `kernHAC`
vcovHC

Examples

```r
x <- sin(1:100)
y <- 1 + x + rnorm(100)
fm <- lm(y ~ x)
vcovHAC(fm)
vcov(fm)
```

vcovHC  

**Heteroskedasticity-Consistent Covariance Matrix Estimation**

Description

Heteroskedasticity-consistent estimation of the covariance matrix of the coefficient estimates in regression models.

Usage

```r
vcovHC(x, ...)
```

## Default S3 method:
```r
vcovHC(x,
  type = c("HC3", "const", "HC", "HC0", "HC1", "HC2", "HC4", "HC4m", "HC5"),
  omega = NULL, sandwich = TRUE, ...)
```

meatHC(x, type = , omega = NULL)

Arguments

- `x` a fitted model object.
- `type` a character string specifying the estimation type. For details see below.
- `omega` a vector or a function depending on the arguments residuals (the working residuals of the model), diaghat (the diagonal of the corresponding hat matrix) and df (the residual degrees of freedom). For details see below.
- `sandwich` logical. Should the sandwich estimator be computed? If set to FALSE only the meat matrix is returned.
- `...` arguments passed to `sandwich`.

Details

The function `meatHC` is the real work horse for estimating the meat of HC sandwich estimators – the default `vcovHC` method is a wrapper calling `sandwich` and `bread`. See Zeileis (2006) for more implementation details. The theoretical background, exemplified for the linear regression model, is described below and in Zeileis (2004). Analogous formulas are employed for other types of models.

When `type = "const"` constant variances are assumed and and `vcovHC` gives the usual estimate of the covariance matrix of the coefficient estimates:
\[ \hat{\sigma}^2(X^\top X)^{-1} \]

All other methods do not assume constant variances and are suitable in case of heteroskedasticity. "HC" (or equivalently "HC0") gives White’s estimator, the other estimators are refinements of this. They are all of form

\[ (X^\top X)^{-1}X^\top \Omega X(X^\top X)^{-1} \]

and differ in the choice of Omega. This is in all cases a diagonal matrix whose elements can be either supplied as a vector omega or as a function omega of the residuals, the diagonal elements of the hat matrix and the residual degrees of freedom. For White’s estimator

\[
\text{omega} \leftarrow \text{function(residuals, diaghat, df) residuals}^2
\]

Instead of specifying the diagonal omega or a function for estimating it, the type argument can be used to specify the HC0 to HC5 estimators. If omega is used, type is ignored.

Long & Ervin (2000) conduct a simulation study of HC estimators (HC0 to HC3) in the linear regression model, recommending to use HC3 which is thus the default in vcovHC. Cribari-Neto (2004), Cribari-Neto, Souza, & Vasconcellos (2007), and Cribari-Neto & Da Silva (2011), respectively, suggest the HC4, HC5, and modified HC4m type estimators. All of them are tailored to take into account the effect of leverage points in the design matrix. For more details see the references.

Value

A matrix containing the covariance matrix estimate.

References


See Also

lm, hccm, bptest, ncv.test

Examples

## generate linear regression relationship
## with homoskedastic variances
x <- sin(1:100)
y <- rnorm(100)
## model fit and HC3 covariance
fm <- lm(y ~ x)
vcovHC(fm)
## usual covariance matrix
vcovHC(fm, type = "const")
vcov(fm)

sigma2 <- sum(residuals(lm(y ~ x))^2)/98
sigma2 * solve(crossprod(cbind(1, x)))

vcovOPG

Outer Product of Gradients Covariance Matrix Estimation

Description

Outer product of gradients estimation for the covariance matrix of the coefficient estimates in regression models.

Usage

vcovOPG(x, adjust = FALSE, ...)

Arguments

x a fitted model object.
adjust logical. Should a finite sample adjustment be made? This amounts to multiplication with

\[
n/(n - k)
\]

where

\[n\]

is the number of observations and

\[k\]

the number of estimated parameters.

... arguments passed to the estfun function.
Details

In correctly specified models, the “meat” matrix (cross product of estimating functions, see \texttt{meat}) and the inverse of the “bread” matrix (inverse of the derivative of the estimating functions, see \texttt{bread}) are equal and correspond to the Fisher information matrix. Typically, an empirical version of the bread is used for estimation of the information but alternatively it is also possible to use the meat. This method is also known as the outer product of gradients (OPG) estimator (Cameron & Trivedi 2005).

Using the \texttt{sandwich} infrastructure, the OPG estimator could easily be computed via \texttt{solve(meat(obj))} (modulo scaling). To employ numerically more stable implementation of the inversion, this simple convenience function can be used: \texttt{vcovOPG(obj)}.

Note that this only works if the \texttt{estfun()} method computes the maximum likelihood scores (and not a scaled version such as least squares scores for "lm" objects).

Value

A matrix containing the covariance matrix estimate.

References


See Also

\texttt{meat, bread, sandwich}

Examples

```r
## generate poisson regression relationship
x <- sin(1:100)
y <- rpois(100, exp(1 + x))
## compute usual covariance matrix of coefficient estimates
fm <- glm(y ~ x, family = poisson)
vco(fm)
vcovOPG(fm)
```

weightsAndrews

\textit{Kernel-based HAC Covariance Matrix Estimation}

Description

weightsAndrews

Usage

kernhac(x, order.by = NULL, prewhite = 1, bw = bwAndrews,
kernel = c("Quadratic Spectral", "Truncated", "Bartlett", "Parzen", "Tukey-Hanning"),
approx = c("AR(1)", "ARMA(1,1)"), adjust = TRUE, diagnostics = FALSE,
sandwich = TRUE, ar.method = "ols", tol = 1e-7, data = list(), verbose = FALSE, ...)

weightsAndrews(x, order.by = NULL, bw = bwAndrews,
kernel = c("Quadratic Spectral", "Truncated", "Bartlett", "Parzen", "Tukey-Hanning"),
prewhte = 1, ar.method = "ols", tol = 1e-7, data = list(), verbose = FALSE, ...)

bwAndrews(x, order.by = NULL, kernel = c("Quadratic Spectral", "Truncated",
"Bartlett", "Parzen", "Tukey-Hanning"), approx = c("AR(1)", "ARMA(1,1)"),
weights = NULL, prewhite = 1, ar.method = "ols", data = list(), ...)

Arguments

x a fitted model object.

order.by Either a vector z or a formula with a single explanatory variable like ~ z. The observations in the model are ordered by the size of z. If set to NULL (the default) the observations are assumed to be ordered (e.g., a time series).

prewhite logical or integer. Should the estimating functions be prewhitened? If TRUE or greater than 0 a VAR model of order as.integer(prewhte) is fitted via ar with method "ols" and demean = FALSE. The default is to use VAR(1) prewhitening.

bw numeric or a function. The bandwidth of the kernel (corresponds to the truncation lag). If set to to a function (the default is bwAndrews) it is adaptively chosen.

kernel a character specifying the kernel used. All kernels used are described in Andrews (1991).

approx a character specifying the approximation method if the bandwidth bw has to be chosen by bwAndrews.

adjust logical. Should a finite sample adjustment be made? This amounts to multiplication with \( n/(n - k) \) where \( n \) is the number of observations and \( k \) the number of estimated parameters.

diagnostics logical. Should additional model diagnostics be returned? See vcovHAC for details.

sandwich logical. Should the sandwich estimator be computed? If set to FALSE only the middle matrix is returned.

ar.method character. The method argument passed to ar for prewhitening (only, not for bandwidth selection).

tol numeric. Weights that exceed tol are used for computing the covariance matrix, all other weights are treated as 0.

data an optional data frame containing the variables in the order.by model. By default the variables are taken from the environment which the function is called from.
verbose logical. Should the bandwidth parameter used be printed?

... further arguments passed to bwAndrews.

weights numeric. A vector of weights used for weighting the estimated coefficients of the approximation model (as specified by approx). By default all weights are 1 except that for the intercept term (if there is more than one variable).

Details

kernHAC is a convenience interface to vcovHAC using weightsAndrews: first a weights function is defined and then vcovHAC is called.

The kernel weights underlying weightsAndrews are directly accessible via the function kweights and require the specification of the bandwidth parameter bw. If this is not specified it can be chosen adaptively by the function bwAndrews (except for the "Truncated" kernel). The automatic bandwidth selection is based on an approximation of the estimating functions by either AR(1) or ARMA(1,1) processes. To aggregate the estimated parameters from these approximations a weighted sum is used. The weights in this aggregation are by default all equal to 1 except that corresponding to the intercept term which is set to 0 (unless there is no other variable in the model) making the covariance matrix scale invariant.

Further details can be found in Andrews (1991).

The estimator of Newey & West (1987) is a special case of the class of estimators introduced by Andrews (1991). It can be obtained using the "Bartlett" kernel and setting bw to lag + 1. A convenience interface is provided in NeweyWest.

Value

kernHAC returns the same type of object as vcovHAC which is typically just the covariance matrix.

weightsAndrews returns a vector of weights.

bwAndrews returns the selected bandwidth parameter.

References


See Also

vcovHAC, NeweyWest, weightsLumley, weave

Examples

curve(kweights(x, kernel = "Quadratic", normalize = TRUE),
     from = 0, to = 3.2, xlab = "x", ylab = "k(x)"
) curve(kweights(x, kernel = "Bartlett", normalize = TRUE),
     from = 0, to = 3.2, col = 2, add = TRUE)
curve(kweights(x, kernel = "Parzen", normalize = TRUE),
from = 0, to = 3.2, col = 3, add = TRUE)
curve(kweights(x, kernel = "Tukey", normalize = TRUE),
from = 0, to = 3.2, col = 4, add = TRUE)
curve(kweights(x, kernel = "Truncated", normalize = TRUE),
from = 0, to = 3.2, col = 5, add = TRUE)

## fit investment equation
data(Investment)
fm <- lm(RealInv ~ RealGnp + RealInt, data = Investment)

## compute quadratic spectral kernel HAC estimator
kernHAC(fm)
kernHAC(fm, verbose = TRUE)

## use Parzen kernel instead, VAR(2) prewhitening, no finite sample
## adjustment and Newey & West (1994) bandwidth selection
kernHAC(fm, kernel = "Parzen", prewhite = 2, adjust = FALSE,
bw = bwNeweyWest, verbose = TRUE)

## compare with estimate under assumption of spheric errors
vcov(fm)

weightsLumley

Weighted Empirical Adaptive Variance Estimation

Description

Usage
weave(x, order.by = NULL, prewhite = FALSE, C = NULL,
method = c("trunc", "smooth"), acf = isoacf, adjust = FALSE,
diagnostics = FALSE, sandwich = TRUE, tol = 1e-7, data = list(), ...)

weightsLumley(x, order.by = NULL, C = NULL,
method = c("trunc", "smooth"), acf = isoacf, tol = 1e-7, data = list(), ...)

Arguments

- **x**: a fitted model object.
- **order.by**: Either a vector `z` or a formula with a single explanatory variable like `~ z`. The observations in the model are ordered by the size of `z`. If set to `NULL` (the default) the observations are assumed to be ordered (e.g., a time series).
- **prewhite**: logical or integer. Should the estimating functions be prewhitened? If TRUE or greater than 0 a VAR model of order as.integer(prewhite) is fitted via ar with method "ols" and demean = FALSE.
weightsLumley

C numeric. The cutoff constant C is by default 4 for method "truncate" and 1 for method "smooth".

method a character specifying the method used, see details.

acf a function that computes the autocorrelation function of a vector, by default isoacf is used.

adjust logical. Should a finite sample adjustment be made? This amounts to multiplication with \( \frac{n}{n - k} \) where \( n \) is the number of observations and \( k \) the number of estimated parameters.

diagnostics logical. Should additional model diagnostics be returned? See vcovHAC for details.

sandwich logical. Should the sandwich estimator be computed? If set to FALSE only the middle matrix is returned.

tol numeric. Weights that exceed tol are used for computing the covariance matrix, all other weights are treated as 0.

data an optional data frame containing the variables in the order by model. By default the variables are taken from the environment which the function is called from.

... currently not used.

Details

weave is a convenience interface to vcovHAC using weightsLumley: first a weights function is defined and then vcovHAC is called.

Both weighting methods are based on some estimate of the autocorrelation function \( \rho \) (as computed by acf) of the residuals of the model \( x \). The weights for the "truncate" method are

\[
I \{ n \rho^2 > C \}
\]

and the weights for the "smooth" method are

\[
\min\{1, C n \rho^2 \}
\]

where \( n \) is the number of observations in the model an C is the truncation constant C.

Further details can be found in Lumley & Heagerty (1999).

Value

weave returns the same type of object as vcovHAC which is typically just the covariance matrix.

weightsLumley returns a vector of weights.

References

weightsLumley

See Also

vcovHAC, weightsAndrews, kernHAC

Examples

```r
x <- sin(1:100)
y <- 1 + x + rnorm(100)
fm <- lm(y ~ x)
weave(fm)
vcov(fm)
```
Index

* Topic datasets
  Investment, 4
  PublicSchools, 12

* Topic regression
  bread, 2
  estfun, 3
  isoacf, 5
  kernHAC, 7, 8, 11, 16, 25
  kernHAC (weightsAndrews), 20
  kweights, 6, 22
  lm, 2, 3, 8, 19
  lrvar, 7
  meat, 8
  NeweyWest, 10
  sandwich, 13
  vcovHAC, 15
  vcovHC, 17
  vcovOPG, 19
  weightsAndrews, 20
  weightsLumley, 23
  ncv.test, 19
  NeweyWest, 7, 8, 10, 22
  pavablocks (isoacf), 5
  PublicSchools, 12
  sandwich, 9, 13, 15–17, 20
  terms, 2, 3
  vcovHAC, 8, 10, 11, 15, 21, 22, 24, 25
  vcovHC, 17
  vcovOPG, 19
  weave, 6, 16, 22
  weave (weightsLumley), 23
  weightsAndrews, 7, 11, 16, 20, 25
  weightsLumley, 6, 16, 22, 23

ar, 10, 15, 21
bptest, 19
bread, 2, 9, 14, 16, 17, 20
bwAndrews (weightsAndrews), 20
bwNeweyWest (NeweyWest), 10

ccoef, 2, 3
cestfun, 3, 9, 16, 19
glm, 2, 3

hccm, 19
Investment, 4
isoacf, 5, 24
kernHAC, 7, 8, 11, 16, 25
kernHAC (weightsAndrews), 20
kweights, 6, 22
lm, 2, 3, 8, 19
lrvar, 7
meat, 8, 14, 20
meathAC, 14
meathAC (vcovHAC), 15
meathC, 14
meathC (vcovHC), 17
ncv.test, 19
NeweyWest, 7, 8, 10, 22
pava.blocks (isoacf), 5
PublicSchools, 12
sandwich, 9, 13, 15–17, 20
terms, 2, 3
vcovHAC, 8, 10, 11, 15, 21, 22, 24, 25
vcovHC, 17
vcovOPG, 19
weave, 6, 16, 22
weave (weightsLumley), 23
weightsAndrews, 7, 11, 16, 20, 25
weightsLumley, 6, 16, 22, 23