

Package ‘sandwich’

November 18, 2009

Version 2.2-2

Date 2009-11-18

Title Robust Covariance Matrix Estimators

Author Thomas Lumley, Achim Zeileis

Maintainer Achim Zeileis <Achim.Zeileis@R-project.org>

Description Model-robust standard error estimators for cross-sectional, time series and longitudinal data.

LazyLoad yes

LazyData yes

Depends R (>= 2.0.0), stats, zoo

Suggests car, lmtest, strucchange, AER, survival, MASS

Imports stats

License GPL-2

Repository CRAN

Date/Publication 2009-11-18 03:01:05

R topics documented:

bread	2
estfun	3
Investment	4
isoacf	5
kweights	6
meat	7
NeweyWest	8
PublicSchools	10
sandwich	12

vcovHAC	13
vcovHC	15
vcovOPG	17
weightsAndrews	19
weightsLumley	21

Index	24
--------------	-----------

bread	<i>Bread for Sandwiches</i>
-------	-----------------------------

Description

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

Usage

```
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

Zeileis A (2006), Object-oriented Computation of Sandwich Estimators. *Journal of Statistical Software*, **16**(9), 1–16. URL <http://www.jstatsoft.org/v16/i09/>.

See Also

[lm, glm](#)

Examples

```
## 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
```

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

Zeileis A (2006), Object-oriented Computation of Sandwich Estimators. *Journal of Statistical Software*, **16**(9), 1–16. URL <http://www.jstatsoft.org/v16/i09/>.

See Also

[lm, glm](#)

Examples

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

## estimating function: (y - x'beta) * x
estfun(fm)
residuals(fm) * cbind(1, x)
```

Investment

*US Investment Data***Description**

US data for fitting an investment equation.

Usage

```
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

Greene W.H. (1993), *Econometric Analysis*, 2nd edition. Macmillan Publishing Company, New York.

Executive Office of the President (1984), *Economic Report of the President*. US Government Printing Office, Washington, DC.

Examples

```
## 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),
```

```

    type = "b", pch = 19, ylim = c(-35, 35), ylab = "Residuals")
sigma <- sqrt(sum(residuals(fm)^2)/fm$df.residual) ## maybe used df = 26 instead of 16 ??
abline(h = c(-2, 0, 2) * sigma, lty = 2)

if(require(lmtest)) {
## Newey-West covariances, Example 15.3
coefest(fm, vcov = NeweyWest(fm, lag = 4))
## Note, that the following is equivalent:
coefest(fm, vcov = kernHAC(fm, kernel = "Bartlett", bw = 5, prewhite = FALSE, adjust = FALSE))

## Durbin-Watson test, p. 424, Example 15.4
dwtest(fm)

## Breusch-Godfrey test, p. 427, Example 15.6
bgtest(fm, order = 4)
}

## visualize fitted series
plot(Investment[, "RealInv"], type = "b", pch = 19, ylab = "Real investment")
lines(ts(fitted(fm), start = 1964), col = 4)

## 3-d visualization of fitted model
if(require(scatterplot3d)) {
s3d <- scatterplot3d(Investment[,c(5,7,6)],
  type = "b", angle = 65, scale.y = 1, pch = 16)
s3d$plane3d(fm, lty.box = "solid", col = 4)
}

```

isoacf

Isotonic Autocorrelation Function

Description

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

Usage

```
isoacf(x, lagmax = NULL, weavel = FALSE)
```

Arguments

x	numeric vector.
lagmax	numeric. The maximal lag of the autocorrelations.
weavel	logical. If set to TRUE isoacf uses the <code>acf.R</code> and <code>pava.blocks</code> function from the original <code>weave</code> package, otherwise R's own <code>acf</code> and <code>isoreg</code> 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

Lumley A & Heagerty P (1999), Weighted Empirical Adaptive Variance Estimators for Correlated Data Regression. *Journal of the Royal Statistical Society B*, **61**, 459–477.

Robertson T, Wright FT, Dykstra RL (1988), *Order Restricted Statistical Inference*. New York. Wiley.

See Also

[weave](#), [weightsLumley](#)

Examples

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

kweights

Kernel Weights

Description

Kernel weights for kernel-based heteroskedasticity and autocorrelation consistent (HAC) covariance matrix estimators as introduced by Andrews (1991).

Usage

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

Arguments

<code>x</code>	numeric.
<code>kernel</code>	a character specifying the kernel used. All kernels used are described in Andrews (1991).
<code>normalize</code>	logical. If set to TRUE the kernels are normalized as described in Andrews (1991).

Value

Value of the kernel function at x .

References

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

See Also

[kernHAC](#), [weightsAndrews](#)

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)
```

 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

```
meat(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

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

Zeileis A (2006), Object-oriented Computation of Sandwich Estimators. *Journal of Statistical Software*, **16**(9), 1–16. URL <http://www.jstatsoft.org/v16/i09/>.

See Also

[sandwich](#), [bread](#), [estfun](#)

Examples

```
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

A set of functions implementing the Newey & West (1987, 1994) heteroskedasticity and autocorrelation consistent (HAC) covariance matrix estimators.

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

<code>x</code>	a fitted model object of class "lm" or "glm".
<code>lag</code>	integer specifying the maximum lag with positive weight for the Newey-West estimator. If set to <code>NULL</code> <code>floor(bwNeweyWest(x, ...))</code> is used.
<code>order.by</code>	Either a vector <code>z</code> or a formula with a single explanatory variable like <code>~ z</code> . The observations in the model are ordered by the size of <code>z</code> . If set to <code>NULL</code> (the default) the observations are assumed to be ordered (e.g., a time series).
<code>prewhite</code>	logical or integer. Should the estimating functions be prewhitened? If <code>TRUE</code> or greater than 0 a VAR model of order <code>as.integer(prewhite)</code> is fitted via <code>ar</code> with method "ols" and <code>demean = FALSE</code> . The default is to use VAR(1) prewhitening.
<code>kernel</code>	a character specifying the kernel used. All kernels used are described in Andrews (1991). <code>bwNeweyWest</code> can only compute bandwidths for "Bartlett", "Parzen" and "Quadratic Spectral".
<code>adjust</code>	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.
<code>diagnostics</code>	logical. Should additional model diagnostics be returned? See <code>vcovHAC</code> for details.
<code>sandwich</code>	logical. Should the sandwich estimator be computed? If set to <code>FALSE</code> only the middle matrix is returned.
<code>ar.method</code>	character. The <code>method</code> argument passed to <code>ar</code> for prewhitening (only, not for bandwidth selection).
<code>data</code>	an optional data frame containing the variables in the <code>order.by</code> model. By default the variables are taken from the environment which the function is called from.
<code>verbose</code>	logical. Should the lag truncation parameter used be printed?
<code>weights</code>	numeric. A vector of weights used for weighting the estimated coefficients of the approximation model (as specified by <code>approx</code>). By default all weights are 1 except that for the intercept term (if there is more than one variable).
<code>...</code>	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 Autocorrelation Consistent Covariance Matrix. *Econometrica*, **55**, 703–708.

Newey WK & West KD (1994), Automatic Lag Selection in Covariance Matrix Estimation. *Review of Economic Studies*, **61**, 631–653.

Zeileis A (2004), Econometric Computing with HC and HAC Covariance Matrix Estimators. *Journal of Statistical Software*, **11**(10), 1–17. URL <http://www.jstatsoft.org/v11/i10/>.

See Also

`vcovHAC`, `weightsAndrews`, `kernHAC`

Examples

```
## 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)
```

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

Cribari-Neto F. (2004), Asymptotic Inference Under Heteroskedasticity of Unknown Form, *Computational Statistics & Data Analysis*, **45**, 215-233.

Greene W.H. (1993), *Econometric Analysis*, 2nd edition. Macmillan Publishing Company, New York.

US Department of Commerce (1979), *Statistical Abstract of the United States*. US Government Printing Office, Washington, DC.

Examples

```
## Willam 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)
ps$Income <- ps$Income * 0.0001

## fit quadratic regression, p. 385, Table 14.2
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
coefptest(fmq, df = Inf)[3,4]
coefptest(fmq, df = Inf, vcov = vcovHC(fmq, type = "HC0"))[3,4]
coefptest(fmq, df = Inf, vcov = vcovHC(fmq, type = "HC3"))[3,4]
coefptest(fmq, df = Inf, vcov = vcovHC(fmq, type = "HC4"))[3,4]
## without Alaska (observation 2)
fmq1 <- lm(Expenditure ~ Income + I(Income^2), data = ps[-2,])
coefptest(fmq1, df = Inf)[3,4]
coefptest(fmq1, df = Inf, vcov = vcovHC(fmq1, type = "HC0"))[3,4]
coefptest(fmq1, df = Inf, vcov = vcovHC(fmq1, type = "HC3"))[3,4]
coefptest(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)

```

sandwich

Making Sandwiches with Bread and Meat

Description

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

Usage

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

Arguments

<code>x</code>	a fitted model object.
<code>bread.</code>	either a bread matrix or a function for computing this via <code>bread.(x)</code> .
<code>meat.</code>	either a bread matrix or a function for computing this via <code>meat.(x, ...)</code> .
<code>...</code>	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

Zeileis A (2006), Object-oriented Computation of Sandwich Estimators. *Journal of Statistical Software*, **16**(9), 1–16. URL <http://www.jstatsoft.org/v16/i09/>.

See Also

`bread`, `meat`, `meatHC`, `meatHAC`

Examples

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

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

vcovHAC

Heteroskedasticity and Autocorrelation Consistent (HAC) Covariance Matrix Estimation

Description

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

Usage

```
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

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

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.

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      multiplicative bias correction
df                   Approximate denominator degrees of freedom
```

References

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

Andrews DWK & Monahan JC (1992), An Improved Heteroskedasticity and Autocorrelation Consistent Covariance Matrix Estimator. *Econometrica*, **60**, 953–966.

Lumley A & Heagerty P (1999), Weighted Empirical Adaptive Variance Estimators for Correlated Data Regression. *Journal of the Royal Statistical Society B*, **61**, 459–477.

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

Zeileis A (2004), Econometric Computing with HC and HAC Covariance Matrix Estimators. *Journal of Statistical Software*, **11**(10), 1–17. URL <http://www.jstatsoft.org/v11/i10/>.

Zeileis A (2006), Object-oriented Computation of Sandwich Estimators. *Journal of Statistical Software*, **16**(9), 1–16. URL <http://www.jstatsoft.org/v16/i09/>.

See Also

[weightsLumley](#), [weightsAndrews](#), [weave](#), [kernHAC](#)

Examples

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

Description

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

Usage

```
vcovHC(x, ...)

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

meatHC(x, type = , omega = NULL)
```

Arguments

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

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).

When `type = "const"` constant variances are assumed 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 Ω . 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

```
omega <- function(residuals, diaghat, df) residuals^2
```

Instead of specifying of providing the diagonal `omega` or a function for estimating it, the `type` argument can be used to specify the HC0 to HC4 estimators. If `omega` is used, `type` is ignored.

Long & Ervin (2000) conduct a simulation study of HC estimators in the linear regression model, recommending to use HC3 which is thus the default in `vcovHC`. Cribari-Neto (2004) suggests the HC4 type estimator which is 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

- Cribari-Neto F. (2004), Asymptotic inference under heteroskedasticity of unknown form. *Computational Statistics & Data Analysis* **45**, 215–233.
- Long J. S., Ervin L. H. (2000), Using Heteroscedasticity Consistent Standard Errors in the Linear Regression Model. *The American Statistician*, **54**, 217–224.
- MacKinnon J. G., White H. (1985), Some heteroskedasticity-consistent covariance matrix estimators with improved finite sample properties. *Journal of Econometrics* **29**, 305–325.
- White H. (1980), A heteroskedasticity-consistent covariance matrix and a direct test for heteroskedasticity. *Econometrica* **48**, 817–838.
- Zeileis A (2004), Econometric Computing with HC and HAC Covariance Matrix Estimators. *Journal of Statistical Software*, **11**(10), 1–17. URL <http://www.jstatsoft.org/v11/i10/>.
- Zeileis A (2006), Object-oriented Computation of Sandwich Estimators. *Journal of Statistical Software*, **16**(9), 1–16. URL <http://www.jstatsoft.org/v16/i09/>.

See Also

[lm](#), [hccm](#), [bptest](#), [ncv.test](#)

Examples

```
## generate linear regression relationship
## with homoskedastic variances
x <- sin(1:100)
y <- 1 + x + 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)))
```

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 `meat`) and the inverse of the “bread” matrix (inverse of the derivative of the estimating functions, see `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 **sandwich** infrastructure, the OPG estimator could easily be computed via `solve(meat(obj))` (modulo scaling). To employ numerically more stable implementation of the inversion, this simple convenience function can be used: `vcovOPG(obj)`.

Note that this only works if the `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

Cameron AC and Trivedi PK (2005), *Microeconometrics: Methods and Applications*. Cambridge University Press, Cambridge.

Zeileis A (2006), Object-oriented Computation of Sandwich Estimators. *Journal of Statistical Software*, **16**(9), 1–16. URL <http://http://www.jstatsoft.org/v16/i09/>.

See Also

`meat`, `bread`, `sandwich`

Examples

```
## 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)
vcov(fm)
vcovOPG(fm)
```

weightsAndrews

Kernel-based HAC Covariance Matrix Estimation

Description

A set of functions implementing a class of kernel-based heteroskedasticity and autocorrelation consistent (HAC) covariance matrix estimators as introduced by Andrews (1991).

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"),
               prewhite = 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 of class "lm" or "glm".
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.
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 <code>bw</code> has to be chosen by <code>bwAndrews</code> .
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 <code>vcovHAC</code> for details.
sandwich	logical. Should the sandwich estimator be computed? If set to <code>FALSE</code> only the middle matrix is returned.
ar.method	character. The <code>method</code> argument passed to <code>ar</code> for prewhitening (only, not for bandwidth selection).
tol	numeric. Weights that exceed <code>tol</code> are used for computing the covariance matrix, all other weights are treated as 0.
data	an optional data frame containing the variables in the <code>order.by</code> 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 <code>bwAndrews</code> .
weights	numeric. A vector of weights used for weighting the estimated coefficients of the approximation model (as specified by <code>approx</code>). 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

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 Autocorrelation Consistent Covariance Matrix. *Econometrica*, **55**, 703–708.

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)
```

Description

A set of functions implementing a class of kernel-based heteroskedasticity and autocorrelation consistent (HAC) covariance matrix estimators as introduced by Andrews (1991).

Usage

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

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

Arguments

<code>x</code>	a fitted model object of class "lm" or "glm".
<code>order.by</code>	Either a vector <code>z</code> or a formula with a single explanatory variable like <code>~ z</code> . The observations in the model are ordered by the size of <code>z</code> . If set to <code>NULL</code> (the default) the observations are assumed to be ordered (e.g., a time series).
<code>prewhite</code>	logical or integer. Should the estimating functions be prewhitened? If <code>TRUE</code> or greater than 0 a VAR model of order <code>as.integer(prewhite)</code> is fitted via <code>ar</code> with method <code>"ols"</code> and <code>demean = FALSE</code> .
<code>C</code>	numeric. The cutoff constant <code>C</code> is by default 4 for method <code>"truncate"</code> and 1 for method <code>"smooth"</code> .
<code>method</code>	a character specifying the method used, see details.
<code>acf</code>	a function that computes the autocorrelation function of a vector, by default <code>isoacf</code> is used.
<code>adjust</code>	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.
<code>diagnostics</code>	logical. Should additional model diagnostics be returned? See <code>vcovHAC</code> for details.
<code>sandwich</code>	logical. Should the sandwich estimator be computed? If set to <code>FALSE</code> only the middle matrix is returned.
<code>tol</code>	numeric. Weights that exceed <code>tol</code> are used for computing the covariance matrix, all other weights are treated as 0.
<code>data</code>	an optional data frame containing the variables in the <code>order.by</code> model. By default the variables are taken from the environment which the function is called from.
<code>...</code>	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 ρ (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

$$\min\{1, Cn\rho^2\}$$

where n is the number of observations in the model and C is the truncation constant.

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

Lumley A & Heagerty P (1999), Weighted Empirical Adaptive Variance Estimators for Correlated Data Regression. *Journal of the Royal Statistical Society B*, **61**, 459–477.

See Also

[vcovHAC](#), [weightsAndrews](#), [kernHAC](#)

Examples

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

Index

*Topic **datasets**

Investment, 4
PublicSchools, 10

*Topic **regression**

bread, 2
estfun, 3
isoacf, 5
kweights, 6
meat, 7
NeweyWest, 8
sandwich, 12
vcovHAC, 13
vcovHC, 15
vcovOPG, 17
weightsAndrews, 19
weightsLumley, 21

*Topic **ts**

isoacf, 5
kweights, 6
NeweyWest, 8
vcovHAC, 13
vcovHC, 15
vcovOPG, 17
weightsAndrews, 19
weightsLumley, 21

ar, 9, 14, 20

bptest, 17

bread, 2, 8, 13, 14, 16, 18

bwAndrews (*weightsAndrews*), 19

bwNeweyWest (*NeweyWest*), 8

coef, 2, 3

estfun, 3, 8, 14, 18

glm, 2, 3

hccm, 17

Investment, 4

isoacf, 5, 22

kernHAC, 7, 10, 14, 15, 23

kernHAC (*weightsAndrews*), 19

kweights, 6, 20

lm, 2, 3, 17

meat, 7, 13, 18

meatHAC, 13

meatHAC (*vcovHAC*), 13

meatHC, 13

meatHC (*vcovHC*), 15

ncv.test, 17

NeweyWest, 8, 20, 21

pava.blocks (*isoacf*), 5

PublicSchools, 10

sandwich, 8, 12, 14, 16, 18

terms, 2, 3

vcovHAC, 9, 10, 13, 20–23

vcovHC, 15

vcovOPG, 17

weave, 6, 14, 15, 21

weave (*weightsLumley*), 21

weightsAndrews, 7, 10, 14, 15, 19, 23

weightsLumley, 6, 14, 15, 21, 21