Package 'gmm'

July 2, 2014

Version 1.5-0

Date 2013-04-02

Title Generalized Method of Moments and Generalized Empirical Likelihood

Author Pierre Chausse <pchausse@uwaterloo.ca>

Maintainer Pierre Chausse <pchausse@uwaterloo.ca>

Description It is a complete suite to estimate models based on moment conditions. It includes the two step Generalized method of moments (GMM) of Hansen(1982), the iterated GMM and continuous updated estimator (CUE) of Hansen-Eaton-Yaron(1996) and several methods that belong to the Generalized Empirical Likelihood (GEL) family of estimators, as presented by Smith(1997),Kitamura(1997), Newey-Smith(2004) and Anatolyev(2005).

Depends R (>= 2.10.0), sandwich

Suggests mvtnorm, car, stabledist, MASS, timeDate, timeSeries

Imports stats

License GPL (>= 2)

Repository CRAN

Repository/R-Forge/Project gmm

Repository/R-Forge/Revision 74

Repository/R-Forge/DateTimeStamp 2013-12-10 20:37:59

Date/Publication 2013-12-11 14:57:30

NeedsCompilation no

R topics documented:

`bread` *Bread for sandwiches*

Description

Computes the bread of the sandwich covariance matrix

Usage

```
## S3 method for class 'gmm'
bread(x, ...)
## S3 method for class 'gel'
bread(x, ...)
## S3 method for class 'tsls'
bread(x, ...)
```

Arguments

x	A fitted model of class gmm or gel.
...	Other arguments when bread is applied to another class object

Details

When the weighting matrix is not the optimal one, the covariance matrix of the estimated coefficients is: $(G'WG)^{-1}G'WVWG(G'WG)^{-1}$, where $G = d\bar{g}/d\theta$, W is the matrix of weights, and V is the covariance matrix of the moment function. Therefore, the bread is $(G'WG)^{-1}$, which is the second derivative of the objective function.

The method if not yet available for gel objects.

Value

A $k \times k$ matrix (see details).

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/.

Examples

```
# See \code{\link{gmm}} for more details on this example.
# With the identity matrix
# bread is the inverse of (G'G)

n <- 1000
x <- rnorm(n, mean = 4, sd = 2)
g <- function(tet, x)
        {
        m1 <- (tet[1] - x)
        m2 <- (tet[2]^2 - (x - tet[1])^2)
        m3 <- x^3 - tet[1]*(tet[1]^2 + 3*tet[2]^2)
        f <- cbind(m1, m2, m3)
        return(f)
        }
Dg <- function(tet, x)
        {
       jacobian <- matrix(c( 1, 2*(-tet[1]+mean(x)), -3*tet[1]^2-3*tet[2]^2,0, 2*tet[2],
-6*tet[1]*tet[2]), nrow=3,ncol=2)
        return(jacobian)
        }

res <- gmm(g, x, c(0, 0), grad = Dg,weightsMatrix=diag(3))
G <- Dg(res$coef, x)
bread(res)
solve(crossprod(G))
```

charStable *The characteristic function of a stable distribution*

Description

It computes the theoretical characteristic function of a stable distribution for two different parametrizations. It is used in the vignette to illustrate the estimation of the parameters using GMM.

Usage

```
charStable(theta, tau, pm = 0)
```

Arguments

theta	Vector of parameters of the stable distribution. See details.
tau	A vector of numbers at which the function is evaluated.
pm	The type of parametization. It takes the values 0 or 1.

Details

The function returns the vector $\Psi(\theta, \tau, pm)$ defined as $E(e^{ix\tau}$, where τ is a vector of real numbers, i is the imaginary number, x is a stable random variable with parameters $\theta = (\alpha, \beta, \gamma, \delta)$ and pm is the type of parametrization. The vector of parameters are the characteristic exponent, the skewness, the scale and the location parameters, respectively. The restrictions on the parameters are: $\alpha \in (0, 2]$, $\beta \in [-1, 1]$ and $\gamma > 0$. For mode details see Nolan(2009).

Value

It returns a vector of complex numbers with the dimension equals to `length(tau)`.

References

Nolan J. P. (2009), Stable Disttributions. *Math/Stat Department, American University*. URL http://academic2.american.edu/~jpnolan/stable/stable.html.

Examples

```
# GMM is like GLS for linear models without endogeneity problems

pm <- 0
theta <- c(1.5,.5,1,0)
tau <- seq(-3, 3, length.out = 20)
char_fct <- charStable(theta, tau, pm)
```

coef *Coefficients of GEL or GMM*

Description

It extracts the coefficients from gel or gmm objects.

Usage

```
## S3 method for class 'gmm'
coef(object, ...)
## S3 method for class 'gel'
coef(object, lambda = FALSE, ...)
```

Arguments

object	An object of class gel or gmm returned by the function gel or gmm
lambda	If set to TRUE, the lagrange multipliers are extracted instead of the vector of coefficients
...	Other arguments when coef is applied to an other class object

Value

Vector of coefficients

Examples

```
#################
n = 500
phi<-c(.2,.7)
thet <- 0
sd <- .2
x <- matrix(arima.sim(n=n,list(order=c(2,0,1),ar=phi,ma=thet,sd=sd)),ncol=1)
y <- x[7:n]
ym1 <- x[6:(n-1)]
ym2 <- x[5:(n-2)]

H <- cbind(x[4:(n-3)], x[3:(n-4)], x[2:(n-5)], x[1:(n-6)])
g <- y ~ ym1 + ym2
x <- H
t0 <- c(0,.5,.5)

res <- gel(g, x, t0)

coef(res)
coef(res, lambda = TRUE)
###################
```

```
res <- gmm(g, x)
coef(res)
```

---

confint *Confidence intervals for GMM or GEL*

Description

It produces confidence intervals for the coefficients from gel or gmm estimation.

Usage

```
## S3 method for class 'gel'
confint(object, parm, level = 0.95, lambda = FALSE, ...)
## S3 method for class 'gmm'
confint(object, parm, level = 0.95, ...)
```

Arguments

object	An object of class gel or gmm returned by the function gel or gmm
parm	A specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered.
level	The confidence level
lambda	If set to TRUE, the confidence intervals for the Lagrange multipliers are produced.
...	Other arguments when confint is applied to another classe object

Value

It returns a matrix with the first column being the lower bound and the second the upper bound.

References

Hansen, L.P. (1982), Large Sample Properties of Generalized Method of Moments Estimators. *Econometrica*, **50**, 1029-1054,

Hansen, L.P. and Heaton, J. and Yaron, A.(1996), Finit-Sample Properties of Some Alternative GMM Estimators. *Journal of Business and Economic Statistics*, **14** 262-280.

Examples

```
#################
n = 500
phi<-c(.2,.7)
thet <- 0
sd <- .2
x <- matrix(arima.sim(n = n, list(order = c(2,0,1), ar = phi, ma = thet, sd = sd)), ncol = 1)
y <- x[7:n]
ym1 <- x[6:(n-1)]
ym2 <- x[5:(n-2)]

H <- cbind(x[4:(n-3)], x[3:(n-4)], x[2:(n-5)], x[1:(n-6)])
g <- y ~ ym1 + ym2
x <- H
t0 <- c(0,.5,.5)

res <- gel(g, x, t0)

confint(res)
confint(res, level = 0.90)
confint(res, lambda = TRUE)

########################

res <- gmm(g, x)

confint(res)
confint(res, level = 0.90)
```

`estfun` *Extracts the empirical moment function*

Description

It extracts the matrix of empirical moments so that it can be used by the `kernHAC` function.

Usage

```
## S3 method for class 'gmmFct'
estfun(x, y = NULL, theta = NULL, ...)
## S3 method for class 'gmm'
estfun(x, ...)
## S3 method for class 'gel'
estfun(x, ...)
## S3 method for class 'tsls'
estfun(x, ...)
## S3 method for class 'tsls'
model.matrix(object, ...)
```

Arguments

x	A function of the form $g(\theta, y)$ or a $n \times q$ matrix with typical element $g_i(\theta, y_t)$ for $i = 1, ...q$ and $t = 1, ..., n$ or an object of class gmm. See gmm for more details. For tsls, it is an object of class tsls.
object	An object of class tsls.
y	The matrix or vector of data from which the function $g(\theta, y)$ is computed if g is a function.
theta	Vector of parameters if g is a function.
...	Other arguments when estfun is applied to another class object

Details

For estfun.gmmFct, it returns a $n \times q$ matrix with typical element $g_i(\theta, y_t)$ for $i = 1, ...q$ and $t = 1, ..., n$. It is only used by gmm to obtain the estimates.

For estfun.gmm, it returns the matrix of first order conditions of $\min_\theta \bar{g}' W \bar{g}/2$, which is a $n \times k$ matrix with the t^{th} row being $g(\theta, y_t)WG$, where G is $d\bar{g}/d\theta$. It allows to compute the sandwich covariance matrix using kernHAC or vcovHAC when W is not the optimal matrix.

The method if not yet available for gel objects.

For tsls, model.matrix and estfun are used by vcov() to compute different covariance matrices using the sandwich package. See vcov.tsls. model.matrix returns the fitted values frin the first stage regression and esfun the residuals.

Value

A $n \times q$ matrix (see details).

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/.

Examples

```
n = 500
phi<-c(.2,.7)
thet <- 0
sd <- .2
x <- matrix(arima.sim(n=n,list(order=c(2,0,1),ar=phi,ma=thet,sd=sd)),ncol=1)
y <- x[7:n]
ym1 <- x[6:(n-1)]
ym2 <- x[5:(n-2)]
H <- cbind(x[4:(n-3)], x[3:(n-4)], x[2:(n-5)], x[1:(n-6)])
g <- y ~ ym1 + ym2
x <- H
res <- gmm(g, x,weightsMatrix = diag(5))

gt <- res$gt
```

```
G <- res$G

foc <- gt
foc2 <- estfun(res)

foc[1:5,]
foc2[1:5,]
```

`Finance` *Returns on selected stocks*

Description

Daily returns on selected stocks, the Market portfolio and factors of Fama and French from 1993-01-05 to 2009-01-30 for CAPM and APT analysis

Usage

```
data(Finance)
```

Format

A data frame containing 24 time series. Dates are reported as rownames(). In the following description, company symboles are used.

WMK Returns of WEIS MARKETS INC

UIS Returns of UNISYS CP NEW

ORB Returns of ORBITAL SCIENCES CP

MAT Returns of Mattel, Inc.

ABAX Returns of ABAXIS, Inc.

T Returns of AT&T INC.

EMR Returns of EMERSON ELEC CO

JCS Returns of Communications Systems Inc.

VOXX Returns of Audiovox Corp.

ZOOM Returns of ZOOM Technologies Inc.

TDW Returns of TIDEWATER INC

ROG Returns of Rogers Corporation

GGG Returns of Graco Inc.

PC Returns of Panasonic Corporation

GCO Returns of Genesco Inc.

EBF Returns of ENNIS, INC

F Returns of FORD MOTOR CO

FNM Returns of FANNIE MAE

NHP Returns of NATIONWIDE HLTH PROP

AA Returns of ALCOA INC

rf Risk-free rate of Fama-French

rm Return of the market portfolio of Fama-French

hml Factor High-Minus-Low of Fama-French

smb Factor Small-Minus-Big of Fama-French

Source

http://ca.finance.yahoo.com/ and http://mba.tuck.dartmouth.edu/pages/faculty/ken.french/

`FinRes` *Method to finalize the result of the momentEstim method*

Description

It computes the final results that will be needed to create the object of class `gmm`.).

Usage

```
## S3 method for class 'baseGmm.res'
FinRes(z, object, ...)
```

Arguments

`z`	An object of class determined by the method `momentEstim`.
`object`	An object produced my `getModel`
`...`	Other argument to be passed to other `FinRes` methods.

Value

It returns an object of class `gmm`. See `gmm` for more details.

References

Hansen, L.P. (1982), Large Sample Properties of Generalized Method of Moments Estimators. *Econometrica*, **50**, 1029-1054,

Hansen, L.P. and Heaton, J. and Yaron, A.(1996), Finit-Sample Properties of Some Alternative GMM Estimators. *Journal of Business and Economic Statistics*, **14** 262-280.

fitted *Fitted values of GEL and GMM*

Description

Method to extract the fitted values of the model estimated by gel or gmm.

Usage

```
## S3 method for class 'gel'
fitted(object, ...)
## S3 method for class 'gmm'
fitted(object, ...)
```

Arguments

object	An object of class gel or gel returned by the function gel or gmm
...	Other arguments when fitted is applied to an other class object

Value

It returns a matrix of the estimated mean $\hat{y}$ in g=y~x as it is done by fitted.lm.

Examples

```
# GEL can deal with endogeneity problems

n = 200
phi<-c(.2,.7)
thet <- 0.2
sd <- .2
set.seed(123)
x <- matrix(arima.sim(n = n, list(order = c(2,0,1), ar = phi, ma = thet, sd = sd)), ncol = 1)

y <- x[7:n]
ym1 <- x[6:(n-1)]
ym2 <- x[5:(n-2)]
H <- cbind(x[4:(n-3)], x[3:(n-4)], x[2:(n-5)], x[1:(n-6)])
g <- y ~ ym1 + ym2
x <- H

res <- gel(g, x, c(0,.3,.6))
plot(y, main = "Fitted ARMA with GEL")
lines(fitted(res), col = 2)

# GMM is like GLS for linear models without endogeneity problems

set.seed(345)
```

```
n = 200
phi<-c(.2,.7)
thet <- 0
sd <- .2
x <- matrix(arima.sim(n = n, list(order = c(2,0,1), ar = phi, ma = thet, sd = sd)), ncol = 1)
y <- 10 + 5*rnorm(n) + x

res <- gmm(y ~ x, x)
plot(x, y, main = "Fitted model with GMM")
lines(x, fitted(res), col = 2)
legend("topright", c("Y","Yhat"), col = 1:2, lty = c(1,1))
```

`formula` *Formula method for gel and gmm objects*

Description

Method to extract the formula from `gel` or `gmm` objects.

Usage

```
## S3 method for class 'gel'
formula(x, ...)
## S3 method for class 'gmm'
formula(x, ...)
```

Arguments

`x`	An object of class `gel` or `gmm` returned by the function `gel` or `gmm`
`...`	Other arguments to pass to other methods

Examples

```
## GEL ##
n = 200
phi<-c(.2,.7)
thet <- 0.2
sd <- .2
set.seed(123)
x <- matrix(arima.sim(n = n, list(order = c(2,0,1), ar = phi, ma = thet, sd = sd)), ncol = 1)

y <- x[7:n]
ym1 <- x[6:(n-1)]
ym2 <- x[5:(n-2)]
H <- cbind(x[4:(n-3)], x[3:(n-4)], x[2:(n-5)], x[1:(n-6)])
g <- y ~ ym1 + ym2
x <- H
```

```
res <- gel(g, x, c(0,.3,.6))
formula(res)

# GMM is like GLS for linear models without endogeneity problems

set.seed(345)
n = 200
phi<-c(.2,.7)
thet <- 0
sd <- .2
x <- matrix(arima.sim(n = n, list(order = c(2,0,1), ar = phi, ma = thet, sd = sd)), ncol = 1)
y <- 10 + 5*rnorm(n) + x

res <- gmm(y ~ x, x)
formula(res)
```

---

gel *Generalized Empirical Likelihood estimation*

Description

Function to estimate a vector of parameters based on moment conditions using the GEL method as presented by Newey-Smith(2004) and Anatolyev(2005).

Usage

```
gel(g, x, tet0, gradv = NULL, smooth = FALSE, type = c("EL","ET","CUE","ETEL"),
    kernel = c("Truncated", "Bartlett"), bw = bwAndrews,
    approx = c("AR(1)", "ARMA(1,1)"), prewhite = 1, ar.method = "ols",
    tol_weights = 1e-7, tol_lam = 1e-9, tol_obj = 1e-9, tol_mom = 1e-9,
    maxiterlam = 100, constraint = FALSE, optfct = c("optim", "optimize",
   "nlminb"), optlam = c("nlminb", "optim", "iter"), data, Lambdacontrol = list(),
     model = TRUE, X = FALSE, Y = FALSE, TypeGel = "baseGel", alpha = NULL, ...)
```

Arguments

g	A function of the form $g(\theta, x)$ and which returns a $n \times q$ matrix with typical element $g_i(\theta, x_t)$ for $i = 1, ...q$ and $t = 1, ..., n$. This matrix is then used to build the q sample moment conditions. It can also be a formula if the model is linear (see details below).
tet0	A $k \times 1$ vector of starting values. If the dimension of θ is one, see the argument "optfct".
x	The matrix or vector of data from which the function $g(\theta, x)$ is computed. If "g" is a formula, it is an $n \times Nh$ matrix of instruments (see details below).

gradv — A function of the form $G(\theta, x)$ which returns a $q \times k$ matrix of derivatives of $\bar{g}(\theta)$ with respect to θ. By default, the numerical algorithm numericDeriv is used. It is of course strongly suggested to provide this function when it is possible. This gradiant is used compute the asymptotic covariance matrix of $\hat{\theta}$. If "g" is a formula, the gradiant is not required (see the details below).

smooth — If set to TRUE, the moment function is smoothed as proposed by Kitamura(1997)

type — "EL" for empirical likelihood, "ET" for exponential tilting, "CUE" for continuous updated estimator and "ETEL" for exponentially tilted empirical likelihood of Schennach(2007).

kernel — type of kernel used to compute the covariance matrix of the vector of sample moment conditions (see kernHAC for more details) and to smooth the moment conditions if "smooth" is set to TRUE. Only two types of kernel are available. The truncated implies a Bartlett kernel for the HAC matrix and the Bartlett implies a Parzen kernel (see Smith 2004).

bw — The method to compute the bandwidth parameter. By default it is bwAndrews which is proposed by Andrews (1991). The alternative is bwNeweyWest of Newey-West(1994).

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.

ar.method — character. The method argument passed to ar for prewhitening.

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

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

tol_lam — Tolerance for λ between two iterations. The algorithm stops when $\|\lambda_i - \lambda_{i-1}\|$ reaches tol_lamb (see getLamb)

maxiterlam — The algorithm to compute λ stops if there is no convergence after "maxiterlam" iterations (see getLamb).

tol_obj — Tolerance for the gradiant of the objective function to compute λ (see getLamb).

optfct — Only when the dimension of θ is 1, you can choose between the algorithm optim or optimize. In that case, the former is unreliable. If optimize is chosen, "t0" must be 1×2 which represents the interval in which the algorithm seeks the solution.It is also possible to choose the nlminb algorithm. In that case, borns for the coefficients can be set by the options upper= and lower=.

constraint — If set to TRUE, the constraint optimization algorithm is used. See constrOptim to learn how it works. In particular, if you choose to use it, you need to provide "ui" and "ci" in order to impose the constraint $ui\theta - ci \geq 0$.

tol_mom — It is the tolerance for the moment condition $\sum_{t=1}^{n} p_t g(\theta(x_t) = 0$, where $p_t = \frac{1}{n} D\rho(< g_t, \lambda >)$ is the implied probability. It adds a penalty if the solution diverges from its goal.

optlam — The default is "iter" which solves for λ using the Newton iterative method getLamb. If set to "numeric", the algorithm optim is used to compute λ instead.

data — A data.frame or a matrix with column names (Optional).

Lambdacontrol	Controls for the optimization of the vector of Lagrange multipliers used by either optim, nlminb or constrOptim
model, X, Y	logicals. If TRUE the corresponding components of the fit (the model frame, the model matrix, the response) are returned if g is a formula.
TypeGel	The name of the class object created by the method getModel. It allows developers to extand the package and create other GEL methods.
alpha	Regularization coefficient for discrete CGEL estimation (experimental). By setting alpha to any value, the model is estimated by CGEL of type specified by the option type. See Chausse (2011)
...	More options to give to optim, optimize or constrOptim.

Details

If we want to estimate a model like $Y_t = \theta_1 + X_{2t}\theta_2 + ... + X_k\theta_k + \epsilon_t$ using the moment conditions $Cov(\epsilon_t H_t) = 0$, where H_t is a vector of Nh instruments, than we can define "g" like we do for lm. We would have `g = y~x2+x3+...+xk` and the argument "x" above would become the matrix H of instruments. As for lm, Y_t can be a $Ny \times 1$ vector which would imply that $k = Nh \times Ny$. The intercept is included by default so you do not have to add a column of ones to the matrix H. You do not need to provide the gradiant in that case since in that case it is embedded in gel. The intercept can be removed by adding -1 to the formula. In that case, the column of ones need to be added manually to H.

If "smooth" is set to TRUE, the sample moment conditions $\sum_{t=1}^n g(\theta, x_t)$ is replaced by: $\sum_{t=1}^n g^k(\theta, x_t)$, where $g^k(\theta, x_t) = \sum_{i=-r}^r k(i) g(\theta, x_{t+i})$, where r is a truncated parameter that depends on the bandwidth and $k(i)$ are normalized weights so that they sum to 1.

The method solves $\hat{\theta} = \arg\min \left[\arg\max_\lambda \frac{1}{n} \sum_{t=1}^n \rho(< g(\theta, x_t), \lambda >) - \rho(0)\right]$

Value

'gel' returns an object of 'class' '"gel"'

The functions 'summary' is used to obtain and print a summary of the results.

The object of class "gel" is a list containing at least the following:

coefficients	$k \times 1$ vector of parameters
residuals	the residuals, that is response minus fitted values if "g" is a formula.
fitted.values	the fitted mean values if "g" is a formula.
lambda	$q \times 1$ vector of Lagrange multipliers.
vcov_par	the covariance matrix of "coefficients"
vcov_lambda	the covariance matrix of "lambda"
pt	The implied probabilities
objective	the value of the objective function
conv_lambda	Convergence code for "lambda" (see getLamb)
conv_mes	Convergence message for "lambda" (see getLamb)
conv_par	Convergence code for "coefficients" (see optim, optimize or constrOptim)

terms	the terms object used when g is a formula.
call	the matched call.
y	if requested, the response used (if "g" is a formula).
x	if requested, the model matrix used if "g" is a formula or the data if "g" is a function.
model	if requested (the default), the model frame used if "g" is a formula.

References

Anatolyev, S. (2005), GMM, GEL, Serial Correlation, and Asymptotic Bias. *Econometrica*, **73**, 983-1002.

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

Kitamura, Yuichi (1997), Empirical Likelihood Methods With Weakly Dependent Processes. *The Annals of Statistics*, **25**, 2084-2102.

Newey, W.K. and Smith, R.J. (2004), Higher Order Properties of GMM and Generalized Empirical Likelihood Estimators. *Econometrica*, **72**, 219-255.

Smith, R.J. (2004), GEL Criteria for Moment Condition Models. *Working paper, CEMMAP*.

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.

Schennach, Susanne, M. (2007), Point Estimation with Exponentially Tilted Empirical Likelihood. *Econometrica*, **35**, 634-672.

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

Chausse (2010), Computing Generalized Method of Moments and Generalized Empirical Likelihood with R. *Journal of Statistical Software*, **34**(11), 1–35. URL http://www.jstatsoft.org/v34/i11/.

Chausse (2011), Generalized Empirical likelihood for a continumm of moment conditions. *Working Paper*, *Department of Economics*, *University of Waterloo*.

Examples

```
# First, an exemple with the fonction g()

g <- function(tet, x)
{
n <- nrow(x)
u <- (x[7:n] - tet[1] - tet[2]*x[6:(n-1)] - tet[3]*x[5:(n-2)])
f <- cbind(u, u*x[4:(n-3)], u*x[3:(n-4)], u*x[2:(n-5)], u*x[1:(n-6)])
return(f)
}

Dg <- function(tet,x)
```

```
{
n <- nrow(x)
xx <- cbind(rep(1, (n-6)), x[6:(n-1)], x[5:(n-2)])
        H  <- cbind(rep(1, (n-6)), x[4:(n-3)], x[3:(n-4)], x[2:(n-5)], x[1:(n-6)])
f <- -crossprod(H, xx)/(n-6)
return(f)
}
n = 200
phi<-c(.2, .7)
thet <- 0.2
sd <- .2
set.seed(123)
x <- matrix(arima.sim(n = n, list(order = c(2, 0, 1), ar = phi, ma = thet, sd = sd)), ncol = 1)

res <- gel(g, x, c(0, .3, .6), grad = Dg)
summary(res)

# The same model but with g as a formula....  much simpler in that case

y <- x[7:n]
ym1 <- x[6:(n-1)]
ym2 <- x[5:(n-2)]

H <- cbind(x[4:(n-3)], x[3:(n-4)], x[2:(n-5)], x[1:(n-6)])
g <- y ~ ym1 + ym2
x <- H

res <- gel(g, x, c(0, .3, .6))
summary(res)
```

`getDat` *Extracting data from a formula*

Description

It extract the data from a formula y~z with instrument h and put everything in a matrix. It helps redefine the function $g(\theta, x)$ that is required by `gmm` and `gel`.

Usage

```
getDat(formula, h, data)
```

Arguments

`formula`	A formula that defines the linear model to be estimated (see details).
`h`	A $n \times nh$ matrix of intruments(see details).
`data`	A data.frame or a matrix with colnames (Optionnal).

Details

The model to be estimated is based on the moment conditions $< h, (y - z\theta) >= 0$. It adds a column of ones to z and h by default. They are removed if -1 is added to the formula

Value

x: A $n \times l$ matrix, where $l = ncol(y) + ncol(z) + ncol(h) + 2$ if "intercept" is TRUE and $ncol(y) + ncol(z) + xcol(h)$ if "intercept" is FALSE.

nh: dimension of h

k: dimension of z

ny: dimension of y

Examples

```
n = 500
phi<-c(.2, .7)
thet <- 0.2
sd <- .2
x <- matrix(arima.sim(n = n, list(order = c(2, 0, 1), ar = phi, ma = thet, sd = sd)), ncol = 1)
y <- x[7:n]
ym1 <- x[6:(n-1)]
ym2 <- x[5:(n-2)]
H <- cbind(x[4:(n-3)], x[3:(n-4)], x[2:(n-5)], x[1:(n-6)])

x <- getDat(y ~ ym1 + ym2, H)
```

---

getLamb *Solving for the Lagrange multipliers of Generalized Empirical Likelihood (GEL)*

Description

It computes the vector of Lagrange multipliers, which maximizes the GEL objective function, using an iterative Newton method.

Usage

```
getLamb(gt, l0, type = c("EL","ET","CUE", "ETEL"), tol_lam = 1e-7, maxiterlam = 100,
tol_obj = 1e-7, k = 1, method = c("nlminb", "optim", "iter"), control = list())
```

Arguments

gt	A $n \times q$ matrix with typical element $g_i(\theta, x_t)$
l0	Vector of starting values for lambda
type	"EL" for empirical likelihood, "ET" for exponential tilting and "CUE" for continuous updated estimator. See details for "ETEL".

tol_lam	Tolerance for λ between two iterations. The algorithm stops when $\|\lambda_i - \lambda_{i-1}\|$ reaches tol_lam
maxiterlam	The algorithm stops if there is no convergence after "maxiterlam" iterations.
tol_obj	Tolerance for the gradiant of the objective function. The algorithm returns a non-convergence message if $\max(\|gradiant\|)$ does not reach tol_obj. It helps the gel algorithm to select the right space to look for θ
k	It represents the ratio k1/k2, where $k1 = \int_{-\infty}^{\infty} k(s)ds$ and $k2 = \int_{-\infty}^{\infty} k(s)^2 ds$. See Smith(2004).
method	The iterative procedure uses a Newton method for solving the FOC. It i however recommended to use optim or nlminb. If type is set to "EL" and method to "optim", constrOptim is called to prevent $log(1 - gt'\lambda)$ from producing NA. The gradient and hessian is providewd to nlminb which speed up the convergence. The latter is therefore the default value.
control	Controls to send to optim, nlminb or constrOptim

Details

It solves the problem $\max_\lambda \frac{1}{n}\sum_{t=1}^{n} \rho(gt'\lambda)$. For the type "ETEL", it is only used by gel. In that case λ is obtained by maximizing $\frac{1}{n}\sum_{t=1}^{n} \rho(gt'\lambda)$, using $\rho(v) = -\exp v$ (so ET) and θ by minimizing the same equation but with $\rho(v) - \log(1 - v)$. To avoid NA's, constrOptim is used with the restriction $\lambda' g_t < 1$.

Value

lambda: A $q \times 1$ vector of Lagrange multipliers which solve the system of equations given above.
conv: Details on the type of convergence.

References

Newey, W.K. and Smith, R.J. (2004), Higher Order Properties of GMM and Generalized Empirical Likelihood Estimators. *Econometrica*, **72**, 219-255.

Smith, R.J. (2004), GEL Criteria for Moment Condition Models. *Working paper, CEMMAP*.

Examples

```
g <- function(tet,x)
{
n <- nrow(x)
u <- (x[7:n] - tet[1] - tet[2]*x[6:(n-1)] - tet[3]*x[5:(n-2)])
f <- cbind(u, u*x[4:(n-3)], u*x[3:(n-4)], u*x[2:(n-5)], u*x[1:(n-6)])
return(f)
}
n = 500
phi<-c(.2, .7)
thet <- 0.2
sd <- .2
x <- matrix(arima.sim(n = n, list(order = c(2, 0, 1), ar = phi, ma = thet, sd = sd)), ncol = 1)
gt <- g(c(0,phi),x)
getLamb(gt, type = "EL",method="optim")
```

getModel *Method for setting the properties of a model*

Description

It collects what is needed by the method momentEstim (see details).

Usage

```
## S3 method for class 'baseGmm'
getModel(object, ...)
## S3 method for class 'baseGel'
getModel(object, ...)
```

Arguments

object	An object of class baseGmm
...	Other arguments when getModel is applied to another class object

Value

It returns an object of the right class which determines how the method momentEstim will treat it. For example, if g is a formula and type is set to "cue", it creates an object of class baseGmm.cue.formula. It this case, momentEstim, applied to this object, computes the continuously updated GMM of a linear model. It allows more flexibility this way. For example, it could be easy to add a GMM method which is robust in presence of weak identification simply by creating a new class of model and the associated momentEstime method.

gmm *Generalized method of moment estimation*

Description

Function to estimate a vector of parameters based on moment conditions using the GMM method of Hansen(82).

Usage

```
gmm(g,x,t0=NULL,gradv=NULL, type=c("twoStep","cue","iterative"),
    wmatrix = c("optimal","ident"), vcov=c("HAC","iid","TrueFixed"),
   kernel=c("Quadratic Spectral","Truncated", "Bartlett", "Parzen", "Tukey-Hanning"),
   crit=10e-7,bw = bwAndrews, prewhite = FALSE, ar.method = "ols", approx="AR(1)",
   tol = 1e-7, itermax=100,optfct=c("optim","optimize","nlminb", "constrOptim"),
    model=TRUE, X=FALSE, Y=FALSE, TypeGmm = "baseGmm", centeredVcov = TRUE,
    weightsMatrix = NULL, traceIter = FALSE, data, eqConst = NULL,
    eqConstFullVcov = FALSE, ...)
gmmWithConst(obj, which, value)
```

Arguments

g	A function of the form $g(\theta, x)$ and which returns a $n \times q$ matrix with typical element $g_i(\theta, x_t)$ for $i = 1, ...q$ and $t = 1, ..., n$. This matrix is then used to build the q sample moment conditions. It can also be a formula if the model is linear (see details below).
x	The matrix or vector of data from which the function $g(\theta, x)$ is computed. If "g" is a formula, it is an $n \times Nh$ matrix of instruments or a formula (see details below).
t0	A $k \times 1$ vector of starting values. It is required only when "g" is a function because only then a numerical algorithm is used to minimize the objective function. If the dimension of θ is one, see the argument "optfct".
gradv	A function of the form $G(\theta, x)$ which returns a $q \times k$ matrix of derivatives of $\bar{g}(\theta)$ with respect to θ. By default, the numerical algorithm `numericDeriv` is used. It is of course strongly suggested to provide this function when it is possible. This gradient is used to compute the asymptotic covariance matrix of $\hat{\theta}$ and to obtain the analytical gradient of the objective function if the method is set to "CG" or "BFGS" in `optim` and if "type" is not set to "cue". If "g" is a formula, the gradiant is not required (see the details below).
type	The GMM method: "twostep" is the two step GMM proposed by Hansen(1982) and the "cue" and "iterative" are respectively the continuous updated and the iterative GMM proposed by Hansen, Eaton et Yaron (1996)
wmatrix	Which weighting matrix should be used in the objective function. By default, it is the inverse of the covariance matrix of $g(\theta, x)$. The other choice is the identity matrix which is usually used to obtain a first step estimate of θ
vcov	Assumption on the properties of the random vector x. By default, x is a weakly dependant process. The "iid" option will avoid using the HAC matrix which will accelerate the estimation if one is ready to make that assumption. The option "TrueFixed" is used only when the matrix of weights is provided and it is the optimal one.
kernel	type of kernel used to compute the covariance matrix of the vector of sample moment conditions (see `kernHAC` for more details)
crit	The stopping rule for the iterative GMM. It can be reduce to increase the precision.
bw	The method to compute the bandwidth parameter. By default it is `bwAndrews` which is proposed by Andrews (1991). The alternative is `bwNeweyWest` of Newey-West(1994).
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`.
ar.method	character. The `method` argument passed to `ar` for prewhitening.
approx	A character specifying the approximation method if the bandwidth has to be chosen by `bwAndrews`.
tol	Weights that exceed `tol` are used for computing the covariance matrix, all other weights are treated as 0.

itermax	The maximum number of iterations for the iterative GMM. It is unlikely that the algorithm does not converge but we keep it as a safety.
optfct	Only when the dimension of θ is 1, you can choose between the algorithm optim or optimize. In that case, the former is unreliable. If optimize is chosen, "t0" must be 1×2 which represents the interval in which the algorithm seeks the solution. It is also possible to choose the nlminb algorithm. In that case, boundaries for the coefficients can be set by the options upper= and lower=. The constrOptim is only available for nonlinear models for now. The standard errors may have to be corrected if the estimtes reach the boundary set by ui and ci.
model, X, Y	logical. If TRUE the corresponding components of the fit (the model frame, the model matrix, the response) are returned if g is a formula.
TypeGmm	The name of the class object created by the method getModel. It allows developers to extend the package and create other GMM methods.
centeredVcov	Should the moment function be centered when computing its covariance matrix. Doing so may improve inference.
weightsMatrix	It allows users to provide gmm with a fixed weighting matrix. This matrix must be $q \times q$, symmetric and strictly positive definite. When provided, the type option becomes irrelevant.
traceIter	Tracing information for GMM of type "iter"
data	A data.frame or a matrix with column names (Optional).
eqConst	Either a named vector (if "g" is a function), a simple vector for the nonlinear case indicating which of the θ_0 is restricted, or a qx2 vector defining equality constraints of the form $\theta_i = c_i$. See below for an example.
which, value	The equality constraint is of the form which=value. "which" can be a vector of type characters with the names of the coefficients being constrained, or a vector of type numeric with the position of the coefficient in the whole vector.
obj	Object of class "gmm"
eqConstFullVcov	If FALSE, the constrained coefficients are assumed to be fixed and only the covariance of the unconstrained coefficients is computed. If TRUE, the covariance matrix of the full set of coefficients is computed.
...	More options to give to optim.

Details

If we want to estimate a model like $Y_t = \theta_1 + X_{2t}\theta_2 + \cdots + X_k\theta_k + \epsilon_t$ using the moment conditions $Cov(\epsilon_t H_t) = 0$, where H_t is a vector of Nh instruments, than we can define "g" like we do for lm. We would have $g = y \tilde{} x2 + x3 + \cdots + xk$ and the argument "x" above would become the matrix H of instruments. As for lm, Y_t can be a $Ny \times 1$ vector which would imply that $k = Nh \times Ny$. The intercept is included by default so you do not have to add a column of ones to the matrix H. You do not need to provide the gradiant in that case since in that case it is embedded in gmm. The intercept can be removed by adding -1 to the formula. In that case, the column of ones need to be added manually to H. It is also possible to express "x" as a formula. For example, if the instruments are $\{1, z_1, z_2, z_3\}$, we can set "x" to $\tilde{} z1 + z2 + z3$. By default, a column of ones is added. To remove it, set "x" to $\tilde{} z1 + z2 + z3 - 1$.

The following explains the last example bellow. Thanks to Dieter Rozenich, a student from the Vienna University of Economics and Business Administration. He suggested that it would help to understand the implementation of the Jacobian.

For the two parameters of a normal distribution (μ, σ) we have the following three moment conditions:

$$m_1 = \mu - x_i$$
$$m_2 = \sigma^2 - (x_i - \mu)^2$$
$$m_3 = x_i^3 - \mu(\mu^2 + 3\sigma^2)$$

m_1, m_2 can be directly obtained by the definition of (μ, σ). The third moment condition comes from the third derivative of the moment generating function (MGF)

$$M_X(t) = exp\Big(\mu t + \frac{\sigma^2 t^2}{2}\Big)$$

evaluated at $(t = 0)$.

Note that we have more equations (3) than unknown parameters (2).

The Jacobian of these two conditions is (it should be an array but I can't make it work):

$$1\tilde{}\tilde{}\tilde{}\tilde{}\tilde{}\tilde{}\tilde{}\tilde{}\tilde{}\tilde{}0$$
$$-2\mu + 2x\tilde{}\tilde{}\tilde{}\tilde{}\tilde{}2\sigma$$
$$-3\mu^2 - 3\sigma^{2\tilde{}\tilde{}\tilde{}\tilde{}} - 6\mu\sigma$$

`gmmWithConst()` re-estimates an unrestricted model by adding an equality constraint.

Value

'gmm' returns an object of 'class' '"gmm"'

The functions 'summary' is used to obtain and print a summary of the results. It also compute the J-test of overidentying restriction

The object of class "gmm" is a list containing at least:

`coefficients`	$k \times 1$ vector of coefficients
`residuals`	the residuals, that is response minus fitted values if "g" is a formula.
`fitted.values`	the fitted mean values if "g" is a formula.
`vcov`	the covariance matrix of the coefficients
`objective`	the value of the objective function $\|var(\bar{g})^{-1/2}\bar{g}\|^2$
`terms`	the `terms` object used when g is a formula.
`call`	the matched call.
`y`	if requested, the response used (if "g" is a formula).
`x`	if requested, the model matrix used if "g" is a formula or the data if "g" is a function.
`model`	if requested (the default), the model frame used if "g" is a formula.
`algoInfo`	Information produced by either `optim` or `nlminb` related to the convergence if "g" is a function. It is printed by the `summary.gmm` method.

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/.

Pierre Chausse (2010), Computing Generalized Method of Moments and Generalized Empirical Likelihood with R. *Journal of Statistical Software*, **34**(11), 1–35. URL http://www.jstatsoft.org/v34/i11/.

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.

Hansen, L.P. (1982), Large Sample Properties of Generalized Method of Moments Estimators. *Econometrica*, **50**, 1029-1054,

Hansen, L.P. and Heaton, J. and Yaron, A.(1996), Finit-Sample Properties of Some Alternative GMM Estimators. *Journal of Business and Economic Statistics*, **14** 262-280.

Examples

```
## CAPM test with GMM
data(Finance)
r <- Finance[1:300, 1:10]
rm <- Finance[1:300, "rm"]
rf <- Finance[1:300, "rf"]

z <- as.matrix(r-rf)
t <- nrow(z)
zm <- rm-rf
h <- matrix(zm, t, 1)
res <- gmm(z ~ zm, x = h)
summary(res)

## linear tests can be performed using linearHypothesis from the car package
## The CAPM can be tested as follows:

library(car)
linearHypothesis(res,cbind(diag(10),matrix(0,10,10)),rep(0,10))

# The CAPM of Black
g <- function(theta, x) {
e <- x[,2:11] - theta[1] - (x[,1] - theta[1]) %*% matrix(theta[2:11], 1, 10)
gmat <- cbind(e, e*c(x[,1]))
return(gmat) }

x <- as.matrix(cbind(rm, r))
res_black <- gmm(g, x = x, t0 = rep(0, 11))

summary(res_black)$coefficients
```

```
## APT test with Fama-French factors and GMM

f1 <- zm
f2 <- Finance[1:300, "hml"] - rf
f3 <- Finance[1:300, "smb"] - rf
h <- cbind(f1, f2, f3)
res2 <- gmm(z ~ f1 + f2 + f3, x = h)
coef(res2)
summary(res2)$coefficients

## Same result with x defined as a formula:

res2 <- gmm(z ~ f1 + f2 + f3, ~ f1 + f2 + f3)
coef(res2)

## The following example has been provided by Dieter Rozenich (see details).
# It generates normal random numbers and uses the GMM to estimate
# mean and sd.
#-------------------------------------------------------------------------------
# Random numbers of a normal distribution
# First we generate normally distributed random numbers and compute the two parameters:
n <- 1000
x <- rnorm(n, mean = 4, sd = 2)
# Implementing the 3 moment conditions
g <- function(tet, x)
        {
        m1 <- (tet[1] - x)
        m2 <- (tet[2]^2 - (x - tet[1])^2)
        m3 <- x^3 - tet[1]*(tet[1]^2 + 3*tet[2]^2)
        f <- cbind(m1, m2, m3)
        return(f)
        }
# Implementing the jacobian
Dg <- function(tet, x)
        {
       jacobian <- matrix(c( 1, 2*(-tet[1]+mean(x)), -3*tet[1]^2-3*tet[2]^2,0, 2*tet[2],
   -6*tet[1]*tet[2]), nrow=3,ncol=2)
        return(jacobian)
        }
# Now we want to estimate the two parameters using the GMM.
gmm(g, x, c(0, 0), grad = Dg)

# Two-stage-least-squares (2SLS), or IV with iid errors.
# The model is:
# Y(t) = b[0] + b[1]C(t) + b[2]Y(t-1) + e(t)
# e(t) is an MA(1)
# The instruments are Z(t)={1 C(t) y(t-2) y(t-3) y(t-4)}

getdat <- function(n) {
e <- arima.sim(n,model=list(ma=.9))
C <- runif(n,0,5)
```

```
Y <- rep(0,n)
Y[1] = 1 + 2*C[1] + e[1]
for (i in 2:n){
Y[i] = 1 + 2*C[i] + 0.9*Y[i-1] + e[i]
}
Yt <- Y[5:n]
X <- cbind(1,C[5:n],Y[4:(n-1)])
Z <- cbind(1,C[5:n],Y[3:(n-2)],Y[2:(n-3)],Y[1:(n-4)])
return(list(Y=Yt,X=X,Z=Z))
}

d <- getdat(5000)
res4 <- gmm(d$Y~d$X-1,~d$Z-1,vcov="iid")
res4

### Examples with equality constraint
########################################

# Random numbers of a normal distribution

## Not run:
# The following works but produces warning message because the dimension of coef is 1
# Brent should be used

# without named vector
gmm(g, x, c(4, 0), grad = Dg, eqConst=1)
# with named vector
gmm(g, x, c(mu=4, sig=2), grad = Dg, eqConst="sig")

## End(Not run)

gmm(g, x, c(4, 0), grad = Dg, eqConst=1,method="Brent",lower=0,upper=6)
gmm(g, x, c(mu=4, sig=2), grad = Dg, eqConst="sig",method="Brent",lower=0,upper=6)

# Example with formula
# first coef = 0 and second coef = 1
# Only available for one dimensional yt

z <- z[,1]
res2 <- gmm(z ~ f1 + f2 + f3, ~ f1 + f2 + f3, eqConst = matrix(c(1,2,0,1),2,2))
res2

# CUE with starting t0 requires eqConst to be a vector

res3 <- gmm(z ~ f1 + f2 + f3, ~ f1 + f2 + f3, t0=c(0,1,.5,.5), type="cue", eqConst = c(1,2))
res3

### Examples with equality constraints, where the constrained coefficients is used to compute
### the covariance matrix.
### Useful when some coefficients have been estimated before, they are just identified in GMM
### and don't need to be re-estimated.
### To use with caution because the covariance won't be valid if the coefficients do not solve
### the GMM FOC.
```

```
#####################################

res4 <- gmm(z ~ f1 + f2 + f3, ~ f1 + f2 + f3, t0=c(0,1,.5,.5), eqConst = c(1,2),
       eqConstFullVcov=TRUE)
summary(res4)


### Examples with equality constraint using gmmWithConst
############################################################

res2 <- gmm(z ~ f1 + f2 + f3, ~ f1 + f2 + f3)
gmmWithConst(res2,c("f2","f3"),c(.5,.5))
gmmWithConst(res2,c(2,3),c(.5,.5))
```

KTest *Compute the K statistics of Kleibergen*

Description

The test is proposed by Kleibergen (2005). It is robust to weak identification.

Usage

```
KTest(obj, theta0 = NULL, alphaK = 0.04, alphaJ = 0.01)
## S3 method for class 'gmmTests'
print(x, digits = 5, ...)
```

Arguments

obj	Object of class "gmm" returned by gmm
theta0	The null hypothesis being tested. See details.
alphaK, alphaJ	The size of the J and K tests when combining the two. The overall size is alphaK+alphaJ.
x	An object of class gmmTests returned by KTest
digits	The number of digits to be printed
...	Other arguments when print is applied to another class object

Details

The function produces the J-test and K-statistics which are robust to weak identification. The test is either $H0 : \theta = theta_0$, in which case theta0 must be provided, or $\beta = \beta_0$, where $\theta = (\alpha', \beta')'$, and α is assumed to be identified. In the latter case, theta0 is NULL and obj is a restricted estimation in which β is fixed to β_0. See gmm and the option "eqConst" for more details.

Value

Tests and p-values

References

Keibergen, F. (2005), Testing Parameters in GMM without assuming that they are identified. *Econometrica*, **73**, 1103-1123,

Examples

```
library(mvtnorm)
sig <- matrix(c(1,.5,.5,1),2,2)
n <- 400
e <- rmvnorm(n,sigma=sig)
x4 <- rnorm(n)
w <- exp(-x4^2) + e[,1]
y <- 0.1*w + e[,2]
h <- cbind(x4, x4^2, x4^3, x4^6)
g3 <- y~w
res <- gmm(g3,h)

# Testing the whole vector:

KTest(res,theta0=c(0,.1))

# Testing a subset of the vector (See \code{\link{gmm}})

res2 <- gmm(g3, h, eqConst=matrix(c(2,.1),1,2))
res2
KTest(res2)
```

---

momentEstim *Method for estimating models based on moment conditions*

Description

It estimates a model which is caracterized by the method `getModel` (see details).

Usage

```
## S3 method for class 'baseGmm.twoStep'
momentEstim(object, ...)
## S3 method for class 'baseGmm.twoStep.formula'
momentEstim(object, ...)
## S3 method for class 'baseGmm.iterative.formula'
momentEstim(object, ...)
## S3 method for class 'baseGmm.iterative'
```

```
momentEstim(object, ...)
## S3 method for class 'baseGmm.cue.formula'
momentEstim(object, ...)
## S3 method for class 'baseGmm.cue'
momentEstim(object, ...)
```

Arguments

object	An object created by the method getModel
...	Other arguments when momentEstim is applied to an other class object

Value

It returns an object of class determined by the argument "TypeGMM" of gmm. By default, it is of class baseGmm.res. It estimates the model and organize the results that will be finalized by the method FinRes. More methods can be created in order to use other GMM methods not yet included in the package.

References

Hansen, L.P. (1982), Large Sample Properties of Generalized Method of Moments Estimators. *Econometrica*, **50**, 1029-1054,

Hansen, L.P. and Heaton, J. and Yaron, A.(1996), Finit-Sample Properties of Some Alternative GMM Estimators. *Journal of Business and Economic Statistics*, **14** 262-280.

---

plot *Plot Diagnostics for gel and gmm objects*

Description

It is a plot method for gel or gmm objects.

Usage

```
## S3 method for class 'gel'
plot(x, which = c(1L:4),
    main = list("Residuals vs Fitted values", "Normal Q-Q",
    "Response variable and fitted values","Implied probabilities"),
    panel = if(add.smooth) panel.smooth else points,
    ask = prod(par("mfcol")) < length(which) && dev.interactive(), ...,
    add.smooth = getOption("add.smooth"))

## S3 method for class 'gmm'
plot(x, which = c(1L:3),
    main = list("Residuals vs Fitted values", "Normal Q-Q",
    "Response variable and fitted values"),
    panel = if(add.smooth) panel.smooth else points,
```

```
ask = prod(par("mfcol")) < length(which) && dev.interactive(), ...,
add.smooth = getOption("add.smooth"))
```

Arguments

x	gel or gmm object, typically result of gel or gmm.
which	if a subset of the plots is required, specify a subset of the numbers 1:4 for gel or 1:3 for gmm.
main	Vector of titles for each plot.
panel	panel function. The useful alternative to points, panel.smooth can be chosen by add.smooth = TRUE.
ask	logical; if TRUE, the user is *ask*ed before each plot, see par(ask=.).
...	other parameters to be passed through to plotting functions.
add.smooth	logical indicating if a smoother should be added to most plots; see also panel above.

Details

It is a beta version of a plot method for gel objects. It is a modified version of plot.lm. For now, it is available only for linear models expressed as a formula. Any suggestions are welcome regarding plots or options to include. The first two plots are the same as the ones provided by plot.lm, the third is the dependant variable y with its mean $\hat{y}$ (the fitted values) and the last plots the implied probabilities with the empirical density $1/T$.

Examples

```
# GEL #
n = 500
phi<-c(.2,.7)
thet <- 0
sd <- .2
x <- matrix(arima.sim(n = n,list(order = c(2,0,1), ar = phi, ma = thet, sd = sd)), ncol = 1)
y <- x[7:n]
ym1 <- x[6:(n-1)]
ym2 <- x[5:(n-2)]

H <- cbind(x[4:(n-3)], x[3:(n-4)], x[2:(n-5)], x[1:(n-6)])
g <- y ~ ym1 + ym2
x <- H
t0 <- c(0,.5,.5)

res <- gel(g, x, t0)

plot(res, which = 3)
plot(res, which = 4)

# GMM #
```

```
res <- gmm(g, x)
plot(res, which = 3)
```

---

print *Printing a gmm or gel object*

Description

It is a printing method for gmm or gel objects.

Usage

```
## S3 method for class 'gmm'
print(x, digits = 5, ...)
## S3 method for class 'gel'
print(x, digits = 5, ...)
```

Arguments

x	An object of class gmm or gel returned by the function gmm or gel
digits	The number of digits to be printed
...	Other arguments when print is applied to an other class object

Value

It prints some results from the estimation like the coefficients and the value of the objective function.

Examples

```
# GMM #

n = 500
phi<-c(.2,.7)
thet <- 0
sd <- .2
x <- matrix(arima.sim(n = n, list(order = c(2,0,1), ar = phi, ma = thet, sd = sd)), ncol = 1)
y <- x[7:n]
ym1 <- x[6:(n-1)]
ym2 <- x[5:(n-2)]

H <- cbind(x[4:(n-3)], x[3:(n-4)], x[2:(n-5)], x[1:(n-6)])
g <- y ~ ym1 + ym2
x <- H

res <- gmm(g, x)
print(res)
```

```
# GEL #

t0 <- c(0,.5,.5)
res <- gel(g,x,t0)
print(res)
```

`residuals` *Residuals of GEL or GMM*

Description

Method to extract the residuals of the model estimated by `gmm` or `gel`.

Usage

```
## S3 method for class 'gel'
residuals(object, ...)
## S3 method for class 'gmm'
residuals(object, ...)
```

Arguments

`object`	An object of class `gmm` or `gel` returned by the function `gmm` or `gel`
`...`	Other arguments when `residuals` is applied to an other classe object

Value

It returns the matrix of residuals $(y - \hat{y})$ in `g=y~x` as it is done by `residuals.lm`.

Examples

```
# GEL can deal with endogeneity problems

n = 200
phi<-c(.2,.7)
thet <- 0.2
sd <- .2
set.seed(123)
x <- matrix(arima.sim(n = n, list(order = c(2,0,1), ar = phi, ma = thet, sd = sd)), ncol = 1)

y <- x[7:n]
ym1 <- x[6:(n-1)]
ym2 <- x[5:(n-2)]
H <- cbind(x[4:(n-3)], x[3:(n-4)], x[2:(n-5)], x[1:(n-6)])
g <- y ~ ym1 + ym2
x <- H

res <- gel(g, x, c(0,.3,.6))
```

```
e <- residuals(res)
plot(e, type = 'l', main = "Residuals from an ARMA fit using GEL")

# GMM is like GLS for linear models without endogeneity problems

set.seed(345)
n = 200
phi<-c(.2,.7)
thet <- 0
sd <- .2
x <- matrix(arima.sim(n = n, list(order = c(2,0,1), ar = phi, ma = thet, sd = sd)), ncol = 1)
y <- 10 + 5*rnorm(n) + x

res <- gmm(y ~ x, x)
plot(x, residuals(res), main = "Residuals of an estimated model with GMM")
```

---

smoothG *Kernel smoothing of a matrix of time series*

Description

It applies the required kernel smoothing to the moment function in order for the GEL estimator to be valid. It is used by the gel function.

Usage

```
smoothG(x, bw = bwAndrews, prewhite = 1, ar.method = "ols", weights = weightsAndrews,
kernel = c("Bartlett", "Parzen", "Truncated", "Tukey-Hanning"),
approx = c("AR(1)", "ARMA(1,1)"), tol = 1e-7)
```

Arguments

x	a $n \times q$ matrix of time series, where n is the sample size.
bw	The method to compute the bandwidth parameter. By default, it uses the bandwidth proposed by Andrews(1991). As an alternative, we can choose bw=bwNeweyWest (without "") which is proposed by Newey-West(1996).
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.
ar.method	character. The method argument passed to ar for prewhitening.
weights	The smoothing weights can be computed by weightsAndrews of it can be provided manually. If provided, it has to be a $r \times 1$vector (see details).
approx	a character specifying the approximation method if the bandwidth has to be chosen by bwAndrews.
tol	numeric. Weights that exceed tol are used for computing the covariance matrix, all other weights are treated as 0.
kernel	The choice of kernel

Details

The sample moment conditions $\sum_{t=1}^{n} g(\theta, x_t)$ is replaced by: $\sum_{t=1}^{n} g^k(\theta, x_t)$, where $g^k(\theta, x_t) = \sum_{i=-r}^{r} k(i) g(\theta, x_{t+i})$, where r is a truncated parameter that depends on the bandwidth and $k(i)$ are normalized weights so that they sum to 1.

If the vector of weights is provided, it gives only one side weights. For exemple, if you provide the vector (1,.5,.25), $k(i)$ will become $(.25, .5, 1, .5, .25)/(.25 + .5 + 1 + .5 + .25) = (.1, .2, .4, .2, .1)$

Value

smoothx: A $q \times q$ matrix containing an estimator of the asymptotic variance of $\sqrt{n}\bar{x}$, where $\bar{x}$ is $q \times 1$vector with typical element $\bar{x}_i = \frac{1}{n} \sum_{j=1}^{n} x_{ji}$. This function is called by `gel` but can also be used by itself.

`kern_weights`: Vector of weights used for the smoothing.

References

Anatolyev, S. (2005), GMM, GEL, Serial Correlation, and Asymptotic Bias. *Econometrica*, **73**, 983-1002.

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

Kitamura, Yuichi (1997), Empirical Likelihood Methods With Weakly Dependent Processes. *The Annals of Statistics*, **25**, 2084-2102.

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

Examples

```
g <- function(tet, x)
{
n <- nrow(x)
u <- (x[7:n] - tet[1] - tet[2]*x[6:(n-1)] - tet[3]*x[5:(n-2)])
f <- cbind(u, u*x[4:(n-3)], u*x[3:(n-4)], u*x[2:(n-5)], u*x[1:(n-6)])
return(f)
}
n = 500
phi<-c(.2, .7)
thet <- 0.2
sd <- .2
x <- matrix(arima.sim(n = n, list(order = c(2, 0, 1), ar = phi, ma = thet, sd = sd)), ncol = 1)
gt <- g(c(0, phi), x)
sgt <- smoothG(gt)$smoothx
plot(gt[,1])
lines(sgt[,1])
```

specTest *Compute tests of specification*

Description

Generic function for testing the specification of estimated models. It computes the J-test from gmm objects and J-test, LR-test and LM-test from gel objects.

Usage

```
## S3 method for class 'gmm'
specTest(x, ...)
## S3 method for class 'gel'
specTest(x, ...)
## S3 method for class 'specTest'
print(x, digits = 5, ...)
specTest(x, ...)
```

Arguments

x	A fitted model object.
digits	The number of digits to be printed.
...	Arguments passed to methods.

Value

Tests and p-values

References

Hansen, L.P. (1982), Large Sample Properties of Generalized Method of Moments Estimators. *Econometrica*, **50**, 1029-1054,

Smith, R. J. (2004), GEL Criteria for Moment Condition Models. *CeMMAP working papers, Institute for Fiscal Studies*

Examples

```
#################
n = 500
phi<-c(.2,.7)
thet <- 0
sd <- .2
x <- matrix(arima.sim(n=n,list(order=c(2,0,1),ar=phi,ma=thet,sd=sd)),ncol=1)
y <- x[7:n]
ym1 <- x[6:(n-1)]
ym2 <- x[5:(n-2)]
```

```
H <- cbind(x[4:(n-3)], x[3:(n-4)], x[2:(n-5)], x[1:(n-6)])
g <- y ~ ym1 + ym2
x <- H
t0 <- c(0,.5,.5)

res <- gel(g, x, t0)
specTest(res)

###################
res <- gmm(g, x)
specTest(res)
```

---

summary *Method for object of class gmm or gel*

Description

It presents the results from the gmm or gel estimation in the same fashion as summary does for the lm class objects for example. It also compute the tests for overidentifying restrictions.

Usage

```
## S3 method for class 'gmm'
summary(object, ...)
## S3 method for class 'gel'
summary(object, ...)
## S3 method for class 'tsls'
summary(object, vcov = NULL, ...)
## S3 method for class 'summary.gmm'
print(x, digits = 5, ...)
## S3 method for class 'summary.gel'
print(x, digits = 5, ...)
## S3 method for class 'summary.tsls'
print(x, digits = 5, ...)
```

Arguments

object	An object of class gmm or gel returned by the function gmm or gel
x	An object of class summary.gmm or summary.gel returned by the function summary.gmm summary.gel
digits	The number of digits to be printed
vcov	An alternative covariance matrix computed with vcov.tsls
...	Other arguments when summary is applied to another class object

Value

It returns a list with the parameter estimates and their standard deviations, t-stat and p-values. It also returns the J-test and p-value for the null hypothesis that $E(g(\theta, X) = 0$

References

Hansen, L.P. (1982), Large Sample Properties of Generalized Method of Moments Estimators. *Econometrica*, **50**, 1029-1054,

Hansen, L.P. and Heaton, J. and Yaron, A.(1996), Finit-Sample Properties of Some Alternative GMM Estimators. *Journal of Business and Economic Statistics*, **14** 262-280.

Anatolyev, S. (2005), GMM, GEL, Serial Correlation, and Asymptotic Bias. *Econometrica*, **73**, 983-1002.

Kitamura, Yuichi (1997), Empirical Likelihood Methods With Weakly Dependent Processes. *The Annals of Statistics*, **25**, 2084-2102.

Newey, W.K. and Smith, R.J. (2004), Higher Order Properties of GMM and Generalized Empirical Likelihood Estimators. *Econometrica*, **72**, 219-255.

Examples

```
# GMM #
set.seed(444)
n = 500
phi<-c(.2,.7)
thet <- 0
sd <- .2
x <- matrix(arima.sim(n = n, list(order = c(2,0,1), ar = phi, ma = thet, sd = sd)), ncol = 1)
y <- x[7:n]
ym1 <- x[6:(n-1)]
ym2 <- x[5:(n-2)]
ym3 <- x[4:(n-3)]
ym4 <- x[3:(n-4)]
ym5 <- x[2:(n-5)]
ym6 <- x[1:(n-6)]

g <- y ~ ym1 + ym2
x <- ~ym3+ym4+ym5+ym6

res <- gmm(g, x)

summary(res)

# GEL #

t0 <- res$coef
res <- gel(g, x, t0)
summary(res)


# tsls #
```

```
res <- tsls(y ~ ym1 + ym2,~ym3+ym4+ym5+ym6)
summary(res)
```

`tsls` *Two stage least squares estimation*

Description

Function to estimate a linear model by the two stage least squares method.

Usage

```
tsls(g,x,data)
```

Arguments

`g`	A formula describing the linear regression model (see details below).
`x`	The matrix of instruments (see details below).
`data`	A data.frame or a matrix with column names (Optionnal).

Details

The function just calls `gmm` with the option vcov="iid". It just simplifies the the implementation of 2SLS. The users don't have to worry about all the options offered in `gmm`. The model is

$$Y_i = X_i\beta + u_i$$

In the first step, `lm` is used to regress X_i on the set of instruments Z_i. The second step also uses `lm` to regress Y_i on the fitted values of the first step.

Value

'tsls' returns an object of 'class' '"tsls"' which inherits from class '"gmm"'.

The functions 'summary' is used to obtain and print a summary of the results. It also compute the J-test of overidentying restriction

The object of class "gmm" is a list containing at least:

`coefficients`	$k \times 1$ vector of coefficients
`residuals`	the residuals, that is response minus fitted values if "g" is a formula.
`fitted.values`	the fitted mean values if "g" is a formula.
`vcov`	the covariance matrix of the coefficients
`objective`	the value of the objective function $\|var(\bar{g})^{-1/2}\bar{g}\|^2$
`terms`	the `terms` object used when g is a formula.

call	the matched call.
y	if requested, the response used (if "g" is a formula).
x	if requested, the model matrix used if "g" is a formula or the data if "g" is a function.
model	if requested (the default), the model frame used if "g" is a formula.
algoInfo	Information produced by either optim or nlminb related to the convergence if "g" is a function. It is printed by the summary.gmm method.

References

Hansen, L.P. (1982), Large Sample Properties of Generalized Method of Moments Estimators. *Econometrica*, **50**, 1029-1054,

Examples

```
n <- 1000
e <- arima.sim(n,model=list(ma=.9))
C <- runif(n,0,5)
Y <- rep(0,n)
Y[1] = 1 + 2*C[1] + e[1]
for (i in 2:n){
Y[i] = 1 + 2*C[i] + 0.9*Y[i-1] + e[i]
}
Yt <- Y[5:n]
X <- cbind(C[5:n],Y[4:(n-1)])
Z <- cbind(C[5:n],Y[3:(n-2)],Y[2:(n-3)],Y[1:(n-4)])

res <- tsls(Yt~X,~Z)
res
```

---

vcov *Variance-covariance matrix of GMM or GEL*

Description

It extracts the matrix of variances and covariances from gmm or gel objects.

Usage

```
## S3 method for class 'gmm'
vcov(object, ...)
## S3 method for class 'gel'
vcov(object, lambda = FALSE, ...)
## S3 method for class 'tsls'
vcov(object, type=c("Classical","HC0","HC1","HAC"), hacProp = list(), ...)
```

Arguments

object	An object of class gmm or gmm returned by the function gmm or gel
lambda	If set to TRUE, the covariance matrix of the Lagrange multipliers is produced.
type	Type of covariance matrix for the meat
hacProp	A list of arguments to pass to kernHAC
...	Other arguments when vcov is applied to another class object

Details

For tsls(), if vcov is set to a different value thand "Classical", a sandwich covariance matrix is computed.

Value

A matrix of variances and covariances

Examples

```
# GMM #
n = 500
phi<-c(.2,.7)
thet <- 0
sd <- .2
x <- matrix(arima.sim(n = n,list(order = c(2,0,1), ar = phi, ma = thet, sd = sd)), ncol = 1)
y <- x[7:n]
ym1 <- x[6:(n-1)]
ym2 <- x[5:(n-2)]

H <- cbind(x[4:(n-3)], x[3:(n-4)], x[2:(n-5)], x[1:(n-6)])
g <- y ~ ym1 + ym2
x <- H

res <- gmm(g, x)
vcov(res)

## GEL ##

t0 <- c(0,.5,.5)
res <- gel(g, x, t0)
vcov(res)
vcov(res, lambda = TRUE)
```

Index