Package ‘pco’

October 14, 2022

Type Package

Version 1.0.1

Date 2015-07-26

Title Panel Cointegration Tests

Author Georgi Marinov

Maintainer Georgi Marinov <georgi.marinov@ue-varna.bg>

Depends stats

Description Computation of the Pedroni (1999) panel cointegration test statistics. Reported are the empirical and the standardized values.

License GPL-2

NeedsCompilation no

Repository CRAN

Date/Publication 2015-07-29 00:06:51

R topics documented:

`pco-package` *Panel Cointegration Tests*

Description

Computation of the Pedroni (1999) panel cointegration test statistics. Reported are the empirical values and the standardized values (as suggested in Pedroni, 1999).

Details

Package:	pco
Type:	Package
Version:	1.0.1
Date:	2015-07-26
License:	GPL-2

Computation of the Pedroni (1999) panel cointegration test statistics. Reported are the empirical and the standardized values (as suggested in Pedroni, 1999).

The package includes two functions: 'pedroni99' is for the bivariate case (one Y, one X, no NA values), data must be in matrices (easier for use); 'pedroni99m' is for the multivariate case (one Y, multiple X, no NA values), data must be in an array of all variables.

Author(s)

Georgi Marinov Maintainer: Georgi Marinov <georgi.marinov@ue-varna.bg>

References

Newey, Whitney K.; West, Kenneth D. (1994). "Automatic lag selection in covariance matrix estimation". Review of Economic Studies 61 (4): 631-654.

Pedroni, Peter, 1999. "Critical Values for Cointegration Tests in Heterogeneous Panels with Multiple Regressors," Oxford Bulletin of Economics and Statistics, Department of Economics, University of Oxford, vol. 61(0), pages 653-70, Special I.

See Also

pedroni99

pedroni99m

Examples

```
data(gdi)
data(gds)
pedroni99(gdi, gds)

xx<-array(cumsum(rnorm(10000)),dim=c(100,20,5))
pedroni99m(xx)
```

---

gdi *Investment in advanced countries.*

Description

Gross domestic investment as a share of GDP (in percent) for 25 high income OECD countries, 1973-2013. Retrieved from the World Development Indicators Worldbank Database.

Usage

```
data(gdi)
```

Format

The format is: num [1:41, 1:25] 29.1 28.6 25.9 26.3 25.9 ... - attr(*, "dimnames")=List of 2 ..$: chr [1:41] "X1973..YR1973." "X1974..YR1974." "X1975..YR1975." "X1976..YR1976." ... ..$: chr [1:25] "AUS" "AUT" "BEL" "CAN" ...

Details

Gross fixed capital formation (percent of GDP), "Gross fixed capital formation (NE.GDI.FTOT.ZS, formerly gross domestic fixed investment).", "World Bank national accounts data, and OECD National Accounts data files."

Source

http://databank.worldbank.org/data/reports.aspx?source=world-development-indicators

References

The World Bank: Dataset name: Gross fixed capital formation (percent of GDP).

Examples

```
data(gdi)
plot(gdi)
## maybe str(gdi) ; plot(gdi) ...
```

---

gds *Savings in advanced countries.*

Description

Gross domestic savings as a share of GDP (in percent) for 25 high income OECD countries, 1973-2013. Retrieved from the World Development Indicators Worldbank Database.

Usage

```
data(gds)
```

Format

The format is: num [1:41, 1:25] 31.5 30.5 26.4 26.6 26.4 ... - attr(*, "dimnames")=List of 2 ..$: chr [1:41] "X1973..YR1973." "X1974..YR1974." "X1975..YR1975." "X1976..YR1976." ... ..$: chr [1:25] "AUS" "AUT" "BEL" "CAN" ...

Details

Gross domestic savings (percent of GDP), Gross domestic savings (NY.GDS.TOTL.ZS) are calculated as GDP less final consumption expenditure (total consumption)., "World Bank national accounts data, and OECD National Accounts data files."

Source

http://databank.worldbank.org/data/reports.aspx?source=world-development-indicators

References

The World Bank: Dataset name: Gross domestic savings (percent of GDP).

Examples

```
data(gds)
plot(gds)
## maybe str(gds) ; plot(gds) ...
```

---

pedroni99 *panel cointegration tests - bivatiate case*

Description

Computation of the Pedroni (1999) panel cointegration test statistics. All statistics are asymptotically normal. Reported are their empirical values and their standardized values (as suggested in Pedroni, 1999).

Usage

```
pedroni99(Y, X, kk = 0, type.stat = 1, ka = 2)
```

Arguments

Y	The 'dependent' variable in the cointegration regression. Must be a matrix (TxN), 'time' in rows, 'individuals' in columns. No missing values are allowed.
X	The 'independent' variable in the cointegration regression. Must be a matrix (TxN), 'time' in rows, 'individuals' in columns. No missing values are allowed.
kk	Parameter for the Newey-West (1994) long term variance estimation (number of lags). Can be a vector, with a different value for each individual series, or a scalar. By default it is set to 'round(4 * (T/100)^(2/9))'.
type.stat	Type of the main regresion: 1 - 'none', 2 - 'intercept', 3 - 'intercept and time trend'.
ka	Number of lags for the ADF type regression on residuals, for the parametric statistics.

Details

The function closely follows the instructions in Pedroni (1999). Calculated and reported are the 7 statistics on page 660 in Pedroni (1999) for the bivariate case. Also reported are their standardized values, as described on page 665 and by use of the adjustment terms in Table 2, page 666, op.cit. H0 is 'no cointegration'.

Value

CALL	The result of 'match.call()'.
METHOD	Title of the test.
STATISTIC	The 7 test statistics in Pedroni (1999), in two columns - for the empirical and the standardized values.

Note

Under H0 ('no cointegration') the autoregressive coefficients, gamma_i = 1 for all i, versus H1: gamma_i < 1 for all i.

The standardized values of the test statistics are asymptotically normal (0,1) under H0.

Author(s)

Georgi Marinov

References

Newey, Whitney K.; West, Kenneth D. (1994). "Automatic lag selection in covariance matrix estimation". Review of Economic Studies 61 (4): 631-654.

Pedroni, Peter, 1999. "Critical Values for Cointegration Tests in Heterogeneous Panels with Multiple Regressors," Oxford Bulletin of Economics and Statistics, Department of Economics, University of Oxford, vol. 61(0), pages 653-70, Special I.

See Also

pedroni99m

Examples

```
data(gdi)
data(gds)
# An illustration for the (non-existent) Feldstein-Horioka paradox.
pedroni99(gdi,gds)

## The function is currently defined as
function (Y, X, kk = 0, type.stat = 1, ka = 2)
{
    ff <- function(Y1, X1) {
        NN = ncol(X1)
        sapply(1:NN, function(l) {
```

```
            lm(Y1[, l] ~ X1[, l] - 1)$residuals
        })
    }
    ff1 <- function(Y1, X1) {
        NN = ncol(X1)
        sapply(1:NN, function(l) {
            lm(Y1[, l] ~ X1[, l])$residuals
        })
    }
    ff2 <- function(Y1, X1) {
        NN = ncol(X1)
        trend = 1:nrow(X1)
        sapply(1:NN, function(l) {
            lm(Y1[, l] ~ X1[, l] + trend)$residuals
        })
    }
    nw <- function(xx, ki) {
        tt = length(xx)
        (1/tt) * sum(sapply(1:ki, function(s) {
            (1 - s/(ki + 1)) * sum(xx[(s + 1):tt] * xx[1:(tt -
                s)])
        }))
    }
adfl<-function (ee, lags) {
nn<-length(ee)
z<-ee[(lags+1):nn]
zl<-ee[lags:(nn-1)]
zd<-matrix(cbind(rep(z,lags)),ncol=lags)
ii<-embed(1:nn,lags)
ii<-ii[-(nrow(ii)),]
zd<-zd-ee[ii]
zd<-zd[,-1]
z<-ee[(lags+1):nn]
zl<-ee[lags:(nn-1)]
return(lm(z ~ zl + zd -1)$residuals)
}
    Y <- as.matrix(Y)
    X <- as.matrix(X)
    if (any((dim(Y) != dim(X)))) {
        stop("Y and X are not compatible.")
    }
    na.fail(Y)
    na.fail(X)
    TD = nrow(X)
    N = ncol(X)
    if (is.vector(kk) && length(kk) == N) {
        k = kk
    }
    else if (kk > 0) {
        k = rep(round(kk), N)
    }
    else {
        i = round(4 * (TD/100)^(2/9))
```

```
    k = rep(i, N)
}
if (ka < 2) {
    ka = 2
    warning("Parameter 'ka' was changed to 2.")
}
ka <- as.vector(ka)
if (length(ka) != N) {
    ka <- rep(ka[1], N)
}
stats <- matrix(nrow = 7, ncol = 2)
rownames(stats) <- c("nipanel", "rhopanel", "tpanelnonpar",
    "tpanelpar", "rhogroup", "tgroupnonpar", "tgrouppar")
colnames(stats) <- c("empirical", "standardized")
statsm <- cbind(c(6.982, -6.388, -1.662, -1.662, -9.889,
    -1.992, -1.992), c(11.754, -9.495, -2.177, -2.177, -12.938,
    -2.453, -2.453), c(21.162, -14.011, -2.648, -2.648, -17.359,
    -2.872, -2.872))
rownames(statsm) <- c("nipanel", "rhopanel", "tpanel", "tpanelp",
    "rhogroup", "tgroup", "tgroupp")
colnames(statsm) <- c("none", "intercept", "trend")
statsv <- cbind(c(81.145, 64.288, 1.559, 1.559, 41.943, 0.649,
    0.649), c(104.546, 57.61, 0.964, 0.964, 51.49, 0.618,
    0.618), c(160.249, 64.219, 0.69, 0.69, 66.387, 0.555,
    0.555))
rownames(statsv) <- c("nipanel", "rhopanel", "tpanel", "tpanelp",
    "rhogroup", "tgroup", "tgroupp")
colnames(statsv) <- c("none", "intercept", "trend")
e <- matrix(ncol = N, nrow = TD)
if (type.stat == 2) {
    e <- ff1(Y, X)
}
else if (type.stat == 3) {
    e <- ff2(Y, X)
}
else {
    e <- ff(Y, X)
    type.stat = 1
}
De <- diff(e)
estar <- e
Destar <- diff(estar)
DX <- diff(X)
DY <- diff(Y)
eta <- matrix(ncol = ncol(DX), nrow = nrow(DX))
eta <- ff(DY, DX)
L11hat2 <- sapply(1:N, function(i) {
    (1/nrow(eta)) * sum(eta[, i]^2) + 2 * nw(eta[, i], k[i])
})
mu <- matrix(ncol = ncol(DX), nrow = nrow(DX))
mu <- ff(e[2:TD, ], e[1:(TD - 1), ])
lambdahat <- sapply(1:N, function(i) {
    nw(mu[, i], k[i])
```

```
})
mustar <- matrix(ncol = ncol(DX), nrow = nrow(DX))
mustar <- sapply(1:N, function(i) {
    adfl(e[, i], ka[i])
})
shatstar2 <- sapply(1:N, function(i) {
    (1/nrow(mustar)) * sum(mustar[, i]^2)
})
stildestar2 <- (1/N) * sum(shatstar2)
shat2 <- sapply(1:N, function(i) {
    (1/nrow(mu)) * sum(mu[, i]^2)
})
sigmahat2 <- shat2 + 2 * lambdahat
sigmatilde2 <- (1/N) * sum(L11hat2^(-2) * sigmahat2)
nipa <- sum(sapply(1:N, function(i) {
    sum((L11hat2[i]^(-2)) * (e[1:(TD - 1), i]^2))
}))
lel <- sum(sapply(1:N, function(i) {
    (L11hat2[i]^(-2)) * sum(sapply(1:(nrow(De)), function(ttt) {
        (e[(ttt), i] * De[ttt, i] - lambdahat[i])
    }))
}))
nipanel <- (TD^2) * (N^(3/2)) * nipa^(-1)
stats[1, 1] <- nipanel
rhopanel <- TD * (N^(1/2)) * (nipa^(-1)) * lel
stats[2, 1] <- rhopanel
tpanelnonpar <- ((sigmatilde2 * nipa)^(-1/2)) * lel
stats[3, 1] <- tpanelnonpar
tpanelpar <- ((stildestar2 * sum(sapply(1:N, function(i) {
    sum((L11hat2[i]^(-2)) * estar[1:(nrow(estar) - 1), i]^2)
})))^(-1/2)) * sum(sapply(1:N, function(i) {
    sum(sapply(1:(nrow(Destar)), function(ttt) {
        (L11hat2[i]^(-2)) * (estar[ttt, i] * Destar[ttt,
            i])
    }))
}))
stats[4, 1] <- tpanelpar
rhogroup <- TD * (N^(-1/2)) * sum(sapply(1:N, function(i) {
    ((sum(e[1:(nrow(e) - 1), i]^2))^(-1)) * sum(sapply(1:(nrow(De)),
        function(ttt) {
            (e[ttt, i] * De[ttt, i] - lambdahat[i])
        }))
}))
stats[5, 1] <- rhogroup
tgroupnonpar <- (N^(-1/2)) * sum(sapply(1:N, function(i) {
    ((sigmahat2[i] * sum(e[1:(nrow(e) - 1), i]^2))^(-1/2)) *
        sum(sapply(1:(nrow(De)), function(ttt) {
            (e[(ttt), i] * De[ttt, i] - lambdahat[i])
        }))
}))
stats[6, 1] <- tgroupnonpar
tgrouppar <- (N^(-1/2)) * sum(sapply(1:N, function(i) {
    (sum(shat2[i] * estar[1:(nrow(estar) - 1), i]^2))^(-1/2) *
```

```
            sum(estar[1:(nrow(estar) - 1), i] * Destar[1:(nrow(estar) -
                1), i])
    }))
    stats[7, 1] <- tgrouppar
    stats[, 2] <- sapply(1:7, function(i) {
        (stats[i, 1] - statsm[i, type.stat] * sqrt(N))/sqrt(statsv[i,
            type.stat])
    })
    list(CALL = match.call(), METHOD = "Pedroni(1999) panel tests for cointegration",
        STATISTIC = stats)
}
```

---

pedroni99m *panel cointegration tests - multivatiate case*

Description

Computation of the Pedroni (1999) panel cointegration test statistics. All statistics are asymptotically normal. Reported are their empirical values and their standardized values (as suggested in Pedroni, 1999).

Usage

```
pedroni99m(X, kk = 0, type.stat = 1, ka = 2)
```

Arguments

`X`	The data to be tested for cointegration. Must be a 'cube', an array (TxNxM) with multiple 'sheets', the first 'sheet' is the 'dependent' variable, 'independent' variables are the rest. The first dimension is 'time', the second is 'individuals' and the third is 'variables'. No missing values are allowed.
`kk`	Parameter for the Newey-West (1994) long term variance estimation (number of lags). Can be a vector, with a different value for each individual series, or a scalar. By default it is set to 'round(4 * (T/100)^(2/9))'.
`type.stat`	Type of the main regresion: 1 - 'none', 2 - 'intercept', 3 - 'intercept and time trend'.
`ka`	Number of lags for the ADF type regression on residuals, for the parametric statistics.

Details

The function closely follows the instructions in Pedroni (1999). Calculated and reported are the 7 statistics on page 660 in Pedroni (1999) for the multivariate case. Also reported are their standardized values, as described on page 665 and by use of the adjustment terms in Table 2, page 666, op.cit. H0 is 'no cointegration'.

Value

CALL	The result of 'match.call()'.
METHOD	Title of the test.
STATISTIC	The 7 test statistics in Pedroni (1999), in two columns - for the empirical and the standardized values.

Note

Under H0 ('no cointegration') the autoregressive coefficients, gamma_i = 1 for all i, versus H1: gamma_i < 1 for all i.

The standardized values of the test statistics are asymptotically normal (0,1) under H0.

Author(s)

Georgi Marinov

References

Newey, Whitney K.; West, Kenneth D. (1994). "Automatic lag selection in covariance matrix estimation". Review of Economic Studies 61 (4): 631-654.

Pedroni, Peter, 1999. "Critical Values for Cointegration Tests in Heterogeneous Panels with Multiple Regressors," Oxford Bulletin of Economics and Statistics, Department of Economics, University of Oxford, vol. 61(0), pages 653-70, Special I.

See Also

pedroni99

Examples

```
xx<-array(cumsum(rnorm(10000)),dim=c(100,20,5))
pedroni99m(xx)

## The function is currently defined as
function (X, kk = 0, type.stat = 1, ka = 2)
{
    ffm <- function(Y2, X2) {
        NN = ncol(X2)
        sapply(1:NN, function(l) {
            lm(Y2[, l] ~ X2[, l, ] - 1)$residuals
        })
    }
    ff1m <- function(Y2, X2) {
        NN = ncol(X2)
        sapply(1:NN, function(l) {
            lm(Y2[, l] ~ X2[, l, ])$residuals
        })
    }
```

```
    ff2m <- function(Y2, X2) {
        NN = ncol(X2)
        trend = 1:nrow(X2)
        sapply(1:NN, function(l) {
            lm(Y2[, l] ~ X2[, l, ] + trend)$residuals
        })
    }
    ffmm <- function(Y1, X1) {
        NN = ncol(X1)
        sapply(1:NN, function(l) {
            lm(Y1[, l] ~ X1[, l] - 1)$residuals
        })
    }
    nwm <- function(xx, ki) {
        tt = length(xx)
        (1/tt) * sum(sapply(1:ki, function(s) {
            (1 - s/(ki + 1)) * sum(xx[(s + 1):tt] * xx[1:(tt -
                s)])
        }))
    }
adflm<-function (ee, lags) {
nn<-length(ee)
z<-ee[(lags+1):nn]
zl<-ee[lags:(nn-1)]
zd<-matrix(cbind(rep(z,lags)),ncol=lags)
ii<-embed(1:nn,lags)
ii<-ii[-(nrow(ii)),]
zd<-zd-ee[ii]
zd<-zd[,-1]
z<-ee[(lags+1):nn]
zl<-ee[lags:(nn-1)]
return(lm(z ~ zl + zd -1)$residuals)
}
    na.fail(X)
    Y <- as.matrix(X[, , 1])
    XX <- X[, , (2:dim(X)[3])]
    TD <- dim(X)[1]
    N <- dim(X)[2]
    M <- dim(X)[3]
    if (is.vector(kk) && length(kk) == N) {
        k = kk
    }
    else if (kk > 0) {
        k = rep(round(kk), N)
    }
    else {
        i = round(4 * (TD/100)^(2/9))
        k = rep(i, N)
    }
    if (ka < 2) {
        ka = 2
        warning("Parameter 'ka' was changed to 2.")
    }
```

```
ka <- as.vector(ka)
if (length(ka) != N) {
    ka <- rep(ka[1], N)
}
stamm <- array(dim = c(7, 3, 6))
stamm[, , 1] <- cbind(c(6.982, -6.388, -1.662, -1.662, -9.889,
    -1.992, -1.992), c(11.754, -9.495, -2.177, -2.177, -12.938,
    -2.453, -2.453), c(21.162, -14.011, -2.648, -2.648, -17.359,
    -2.872, -2.872))
stamm[, , 2] <- cbind(c(10.402, -10.191, -2.156, -2.156,
    -13.865, -2.44, -2.44), c(15.197, -13.256, -2.567, -2.567,
    -16.888, -2.827, -2.827), c(24.556, -17.6, -2.967, -2.967,
    -21.116, -3.179, -3.179))
stamm[, , 3] <- cbind(c(14.254, -14.136, -2.571, -2.571,
    -17.834, -2.819, -2.819), c(18.91, -17.163, -2.93, -2.93,
    -20.841, -3.157, -3.157), c(28.046, -21.287, -3.262,
    -3.262, -24.93, -3.464, -3.464))
stamm[, , 4] <- cbind(c(18.198, -18.042, -2.926, -2.926,
    -21.805, -3.151, -3.151), c(22.715, -21.013, -3.241,
    -3.241, -24.775, -3.452, -3.452), c(31.738, -25.13, -3.545,
    -3.545, -28.849, -3.737, -3.737))
stamm[, , 5] <- cbind(c(22.169, -21.985, -3.244, -3.244,
    -25.75, -3.45, -3.45), c(26.603, -24.944, -3.531, -3.531,
    -28.72, -3.726, -3.726), c(35.537, -28.981, -3.806, -3.806,
    -32.716, -3.986, -3.986))
stamm[, , 6] <- cbind(c(26.12, -25.889, -3.533, -3.533, -29.627,
    -3.723, -3.723), c(30.457, -28.795, -3.795, -3.795, -32.538,
    -3.976, -3.976), c(39.231, -32.756, -4.047, -4.047, -36.494,
    -4.217, -4.217))
rownames(stamm) <- c("nipanel", "rhopanel", "tpanelnonpar",
    "tpanelpar", "rhogroup", "tgroupnonpar", "tgrouppar")
colnames(stamm) <- c("none", "intercept", "trend")
stavv <- array(dim = c(7, 3, 6))
stavv[, , 1] <- cbind(c(81.145, 64.288, 1.559, 1.559, 41.943,
    0.649, 0.649), c(104.546, 57.61, 0.964, 0.964, 51.49,
    0.618, 0.618), c(160.249, 64.219, 0.69, 0.69, 66.387,
    0.555, 0.555))
stavv[, , 2] <- cbind(c(140.804, 89.962, 1.286, 1.286, 57.801,
    0.6, 0.6), c(151.094, 81.772, 0.923, 0.923, 67.123, 0.585,
    0.585), c(198.167, 83.815, 0.686, 0.686, 81.832, 0.548,
    0.548))
stavv[, , 3] <- cbind(c(182.45, 103.176, 1.028, 1.028, 72.097,
    0.567, 0.567), c(190.661, 99.331, 0.843, 0.843, 81.835,
    0.56, 0.56), c(239.425, 103.905, 0.688, 0.688, 97.362,
    0.543, 0.543))
stavv[, , 4] <- cbind(c(217.784, 120.787, 0.928, 0.928, 88.611,
    0.559, 0.559), c(231.864, 119.546, 0.8, 0.8, 98.278,
    0.553, 0.553), c(276.997, 124.613, 0.686, 0.686, 113.145,
    0.538, 0.538))
stavv[, , 5] <- cbind(c(256.53, 132.499, 0.82, 0.82, 103.371,
    0.544, 0.544), c(270.451, 134.341, 0.75, 0.75, 113.131,
    0.542, 0.542), c(310.982, 138.227, 0.654, 0.654, 127.989,
    0.53, 0.53))
```

```
stavv[, , 6] <- cbind(c(277.429, 143.561, 0.75, 0.75, 117.059,
    0.53, 0.53), c(293.431, 144.615, 0.685, 0.685, 126.059,
    0.525, 0.525), c(348.217, 154.378, 0.638, 0.638, 140.756,
    0.518, 0.518))
rownames(stavv) <- c("nipanel", "rhopanel", "tpanelnonpar",
    "tpanelpar", "rhogroup", "tgroupnonpar", "tgrouppar")
colnames(stavv) <- c("none", "intercept", "trend")
statsm <- matrix(nrow = 7, ncol = 2)
rownames(statsm) <- c("nipanel", "rhopanel", "tpanelnonpar",
    "tpanelpar", "rhogroup", "tgroupnonpar", "tgrouppar")
colnames(statsm) <- c("empirical", "standardized")
e <- matrix(ncol = N, nrow = TD)
if (type.stat == 2) {
    e <- ff1m(Y, XX)
}
else if (type.stat == 3) {
    e <- ff2m(Y, XX)
}
else {
    e <- ffm(Y, XX)
    type.stat = 1
}
De <- diff(e)
estar <- e
Destar <- diff(estar)
DXX <- array(dim = c((dim(XX)[1] - 1), dim(XX)[2], dim(XX)[3]))
DXX[, , 1:dim(XX)[3]] <- sapply(1:dim(XX)[3], function(i) {
    DXX[, , i] <- diff(XX[, , i])
})
DY <- diff(Y)
eta <- ffm(DY, DXX)
L11hat2 <- sapply(1:N, function(i) {
    (1/nrow(eta)) * sum(eta[, i]^2) + 2 * nwm(eta[, i], k[i])
})
mu <- matrix(ncol = ncol(DY), nrow = nrow(DY))
mu <- ffmm(e[2:TD, ], e[1:(TD - 1), ])
lambdahat <- sapply(1:N, function(i) {
    nwm(mu[, i], k[i])
})
mustar <- matrix(ncol = ncol(DY), nrow = nrow(DY))
mustar <- sapply(1:N, function(i) {
    adflm(e[, i], ka[i])
})
shatstar2 <- sapply(1:N, function(i) {
    (1/nrow(mustar)) * sum(mustar[, i]^2)
})
stildestar2 <- (1/N) * sum(shatstar2)
shat2 <- sapply(1:N, function(i) {
    (1/nrow(mu)) * sum(mu[, i]^2)
})
sigmahat2 <- shat2 + 2 * lambdahat
sigmatilde2 <- (1/N) * sum(L11hat2^(-2) * sigmahat2)
nipa <- sum(sapply(1:N, function(i) {
```

```
        sum((L11hat2[i]^(-2)) * (e[1:(TD - 1), i]^2))
    }))
    lel <- sum(sapply(1:N, function(i) {
        (L11hat2[i]^(-2)) * sum(sapply(2:(nrow(De)), function(ttt) {
            (e[(ttt - 1), i] * De[ttt, i] - lambdahat[i])
        }))
    }))
    nipanel <- (TD^2) * (N^(3/2)) * nipa^(-1)
    statsm[1, 1] <- nipanel
    rhopanel <- TD * (N^(1/2)) * (nipa^(-1)) * lel
    statsm[2, 1] <- rhopanel
    tpanelnonpar <- ((sigmatilde2 * nipa)^(-1/2)) * lel
    statsm[3, 1] <- tpanelnonpar
    tpanelpar <- ((stildestar2 * sum(sapply(1:N, function(i) {
        sum((L11hat2[i]^(-2)) * estar[1:(nrow(estar) - 1), i]^2)
    })))^(-1/2)) * sum(sapply(1:N, function(i) {
        sum(sapply(2:(nrow(Destar)), function(ttt) {
            (L11hat2[i]^(-2)) * (estar[(ttt - 1), i] * Destar[ttt,
                i])
        }))
    }))
    statsm[4, 1] <- tpanelpar
    rhogroup <- TD * (N^(-1/2)) * sum(sapply(1:N, function(i) {
        ((sum(e[1:(nrow(e) - 1), i]^2))^(-1)) * sum(sapply(2:(nrow(De)),
            function(ttt) {
                (e[(ttt - 1), i] * De[ttt, i] - lambdahat[i])
            }))
    }))
    statsm[5, 1] <- rhogroup
    tgroupnonpar <- (N^(-1/2)) * sum(sapply(1:N, function(i) {
        ((sigmahat2[i] * sum(e[1:(nrow(e) - 1), i]^2))^(-1/2)) *
            sum(sapply(2:(nrow(De)), function(ttt) {
                (e[(ttt - 1), i] * De[ttt, i] - lambdahat[i])
            }))
    }))
    statsm[6, 1] <- tgroupnonpar
    tgrouppar <- (N^(-1/2)) * sum(sapply(1:N, function(i) {
        (sum(shat2[i] * estar[1:(nrow(estar) - 1), i]^2))^(-1/2) *
            sum(sapply(2:nrow(Destar), function(tt1) {
                estar[(tt1 - 1), i] * Destar[tt1, i]
            }))
    }))
    statsm[7, 1] <- tgrouppar
    statsm[, 2] <- sapply(1:7, function(i) {
        (statsm[i, 1] - stamm[i, type.stat, M] * sqrt(N))/sqrt(stavv[i,
            type.stat, M])
    })
    list(CALL = match.call(), METHOD = "Pedroni(1999) panel tests for cointegration",
        STATISTIC = statsm)
}
```

Index