

# Package ‘DCCA’

January 20, 2025

**Version** 0.1.1

**Date** 2019-12-18

**Title** Detrended Fluctuation and Detrended Cross-Correlation Analysis

**Depends** R (>= 3.5.0)

**Imports** checkmate

**Suggests** lattice

## Description

A collection of functions to perform Detrended Fluctuation Analysis (DFA) and Detrended Cross-Correlation Analysis (DCCA).

This package implements the results presented in Prass, T.S. and Pumi, G. (2019). ``On the behavior of the DFA and DCCA in trend-stationary processes" <[arXiv:1910.10589](https://arxiv.org/abs/1910.10589)>.

**License** GPL (>= 3)

**Encoding** UTF-8

**LazyData** true

**NeedsCompilation** yes

**Author** Taiane Schaedler Prass [aut, cre]

(<<https://orcid.org/0000-0003-3136-909X>>),

Guilherme Pumi [aut] (<<https://orcid.org/0000-0002-6256-3170>>)

**Maintainer** Taiane Schaedler Prass <[taianeprass@gmail.com](mailto:taianeprass@gmail.com)>

**Repository** CRAN

**Date/Publication** 2020-01-01 15:10:02 UTC

## Contents

covF2dfa . . . . .	2
covFdcca . . . . .	3
EF2dfa . . . . .	5
EFdcca . . . . .	6
F2dfa . . . . .	7
Fdcca . . . . .	9
Jn . . . . .	11

Kkronm . . . . .	12
Km . . . . .	14
Pm . . . . .	15
Qm . . . . .	16
rhodcca . . . . .	17
rhoE . . . . .	18

<b>Index</b>	<b>21</b>
--------------	-----------

---

covF2dfa	<i>Autocovariance function of the detrended variance</i>
----------	--

---

### Description

Calculates the autocovariance of the detrended variance.

### Usage

```
covF2dfa(m = 3, nu = 0, h = 0, overlap = TRUE, G, Cumulants = NULL)
```

### Arguments

m	an integer or integer valued vector indicating the size of the window for the polinomial fit. $\min(m)$ must be greater or equal than $nu$ or else it will return an error.
nu	a non-negative integer denoting the degree of the polinomial fit applied on the integrated series.
h	an integer or integer valued vector indicating the lags for which the autocovariance function is to be calculated.
overlap	logical: if true (the default), overlapping boxes are used for calculations. Otherwise, non-overlapping boxes are applied.
G	the autocovariance matrix for the original time series. The dimension of $G$ must be $(\max(m) + \max(h) + 1)$ by $(\max(m) + \max(h) + 1)$ if <code>overlap = TRUE</code> and $(\max(m) + \max(h))(\max(h) + 1)$ by $(\max(m) + \max(h))(\max(h) + 1)$ otherwise.
Cumulants	The matrix containing the joint cumulants for lags. Dimension must be $(\max(m) + 1) * nrow(G)$ . If not provided, it is assumed that the cumulants are all zero.

### Value

A matrix with the autocovariance of lag  $h$ , for each value of  $m$  provided. This matrix is obtained from expressions (21) for  $h = 0$  and (22) for  $h > 0$  in Prass and Pumi (2019).

### Author(s)

Taiane Schaedler Prass

## References

Prass, T.S. and Pumi, G. (2019). On the behavior of the DFA and DCCA in trend-stationary processes <arXiv:1910.10589>.

## Examples

```
## Not run:
ms = seq(3,100,1)
hs = seq(0,50,1)
overlap = TRUE
nu = 0
m_max = (max(ms)+1)*(max(hs)+1) - max(ms)*max(hs)*as.integer(overlap)

theta = c(c(1,(20:1)/10), rep(0, m_max - 20))
Gamma1 = diag(m_max+1)
Gamma2 = matrix(0, ncol = m_max+1, nrow = m_max+1)
Gamma12 = matrix(0, ncol = m_max+1, nrow = m_max+1)
for(t in 1:(m_max+1)){
  for(h in 0:(m_max+1-t)){
    Gamma2[t,t+h] = sum(theta[1:(length(theta)-h)]*theta[(1+h):length(theta)])
    Gamma2[t+h,t] = Gamma2[t,t+h]
    Gamma12[t,t+h] = theta[h+1]
  }
}

covdfa1 = covF2dfa(m = ms, nu = 0, h = hs,
                  overlap = TRUE, G = Gamma1, Cumulants = NULL)

covdfa2 = covF2dfa(m = ms, nu = 0, h = hs,
                  overlap = TRUE, G = Gamma2, Cumulants = NULL)

cr = rainbow(100)
plot(ms, covdfa1[,1], type = "l", ylim = c(0,20),
     xlab = "m", ylab = expression(gamma[DFA](h)), col = cr[1])
for(i in 2:ncol(covdfa1)){
  points(ms, covdfa1[,i], type = "l", col = cr[i])
}

lattice::wireframe(covdfa1, drape = TRUE,
                  col.regions = rev(rainbow(150))[50:150],
                  zlab = expression(gamma[DFA]), xlab = "m", ylab = "h")

## End(Not run)
```

---

 covFdcca

*Autocovariance function of the detrended cross-covariance*


---

## Description

Calculates the autocovariance of the detrended cross-covariance.

**Usage**

```
covFdcca(m = 3, nu = 0, h = 0, overlap = TRUE, G1, G2, G12, Cumulants = NULL)
```

**Arguments**

m	an integer or integer valued vector indicating the size of the window for the polinomial fit. $\min(m)$ must be greater or equal than $nu$ or else it will return an error.
nu	a non-negative integer denoting the degree of the polinomial fit applied on the integrated series.
h	an integer or integer valued vector indicating the lags for which the autocovariance function is to be calculated. Negative values are not allowed.
overlap	logical: if true (the default), overlapping boxes are used for calculations. Otherwise, non-overlapping boxes are applied.
G1, G2	the autocovariance matrices for the original time series. The dimension of $G1$ and $G2$ must be compatible with the highest values in vectors $m$ and $h$ . More specifically, the dimension of $G1$ and $G2$ is $(\max(m) + \max(h) + 1)$ by $(\max(m) + \max(h) + 1)$ if <code>overlap = TRUE</code> and $\dim(G1) = \dim(G2) = (\max(m) + \max(h))(\max(h) + 1)$ by $(\max(m) + \max(h))(\max(h) + 1)$ otherwise.
G12	the cross-covariance matrix for the original time series. The dimension of $G12$ must be compatible with the highest values in vectors $m$ and $h$ . If <code>overlap = TRUE</code> , $\dim(G12) = [(\max(m) + 1) * (\max(h) + 1) - \max(m) * \max(h)]$ by $[(\max(m) + 1) * (\max(h) + 1) - \max(m) * \max(h)]$ and $\dim(G12) = [(\max(m) + 1) * (\max(h) + 1)]$ by $[\max(m) + 1] * [\max(h) + 1]$ , otherwise
Cumulants	The matrix of cumulants. If not provided, it is assumed that the cumulants are all zero.

**Value**

A matrix of dimension  $\text{length}(h)$  by  $\text{length}(m)$  with the autocovariance of lag  $h$  (rows), for each value of  $m$  (columns) provided. This matrix is obtained from expressions (24) for  $h = 0$  and (25) for  $h > 0$  in Prass and Pumi (2019).

**Author(s)**

Taiane Schaedler Prass

**References**

Prass, T.S. and Pumi, G. (2019). On the behavior of the DFA and DCCA in trend-stationary processes <arXiv:1910.10589>.

**Examples**

```

## Not run:
ms = seq(3,100,1)
hs = seq(0,50,1)
overlap = TRUE
nu = 0
m_max = (max(ms)+1)*(max(hs)+1) - max(ms)*max(hs)*as.integer(overlap)

theta = c(c(1,(20:1)/10), rep(0, m_max - 20))
Gamma1 = diag(m_max+1)
Gamma2 = matrix(0, ncol = m_max+1, nrow = m_max+1)
Gamma12 = matrix(0, ncol = m_max+1, nrow = m_max+1)
for(t in 1:(m_max+1)){
  for(h in 0:(m_max+1-t)){
    Gamma2[t,t+h] = sum(theta[1:(length(theta)-h)]*theta[(1+h):length(theta)])
    Gamma2[t+h,t] = Gamma2[t,t+h]
    Gamma12[t,t+h] = theta[h+1]
  }
}

covdcca = covFdcca(m = ms, nu = 0, h = hs,
                  G1 = Gamma1, G2 = Gamma2, G12 = Gamma12)

## End(Not run)

```

---

EF2dfa

*Expected value of the detrended variance*


---

**Description**

Calculates the expected value of the detrended variance.

**Usage**

```
EF2dfa(m = 3, nu = 0, G, K = NULL)
```

**Arguments**

m	an integer or integer valued vector indicating the size of the window for the polinomial fit. $\min(m)$ must be greater or equal than $nu$ or else it will return an error.
nu	a non-negative integer denoting the degree of the polinomial fit applied on the integrated series.
G	the autocovariance matrix for the original time series. The dimension of $G$ must be $(\max(m) + 1)$ by $(\max(m) + 1)$ .
K	optional: the matrix $K$ . If this matrix is provided and $m$ is an integer, then $nu$ is ignored.

**Value**

A vector of size  $length(m)$  containing the expected values of the detrended variance corresponding to the values of  $m$  provided. This is expression (20) in Prass and Pumi (2019).

**Author(s)**

Taiane Schaedler Prass

**References**

Prass, T.S. and Pumi, G. (2019). On the behavior of the DFA and DCCA in trend-stationary processes <arXiv:1910.10589>.

**Examples**

```
m = 3
K = Km(m = m, nu = 0)
G = diag(m+1)
EF2dfa(G = G, K = K)
# same as
EF2dfa(m = 3, nu = 0, G = G)

# An AR(1) example
phi = 0.4
n = 500
burn.in = 50
eps = rnorm(n + burn.in)
z.temp = numeric(n + burn.in)
z.temp[1] = eps[1]
for(i in 2:(n + burn.in)){
  z.temp[i] = phi*z.temp[i-1] + eps[i]
}
z = z.temp[(burn.in + 1):(n + burn.in)]

F2.dfa = F2dfa(z, m = 3:100, nu = 0, overlap = TRUE)
plot(3:100, F2.dfa, type="o", xlab = "m")
```

---

EFdcca

*Expected value of the detrended cross-covariance*

---

**Description**

Calculates the expected value of the detrended cross-covariance given a cross-covariance matrix.

**Usage**

```
EFdcca(m = 3, nu = 0, G, K = NULL)
```

**Arguments**

m	an integer or integer valued vector indicating the size of the window for the polinomial fit. $\min(m)$ must be greater or equal than $nu$ or else it will result in an error.
nu	a non-negative integer denoting the degree of the polinomial fit applied on the integrated series.
G	the cross-covariance matrix for the original time series. The dimension of $G$ must be $(\max(m) + 1)$ by $(\max(m) + 1)$ .
K	optional: the matrix $K$ . If this matrix and $m$ are provided, then $nu$ is ignored.

**Value**

a size  $\text{length}(m)$  vector containing the expected values of the detrended cross-covariance corresponding to the values of  $m$  provided. This is expression (23) in Prass and Pumi (2019).

**Author(s)**

Taiane Schaedler Prass

**References**

Prass, T.S. and Pumi, G. (2019). On the behavior of the DFA and DCCA in trend-stationary processes <arXiv:1910.10589>.

**Examples**

```
m = 3
K = Km(m = m, nu = 0)
G = diag(m+1)
EFdcca(G = G, K = K)
# same as
EFdcca(m = 3, nu = 0, G = G)
```

---

F2dfa

*Detrended Variance*


---

**Description**

Calculates the detrended variance based on a given time series.

**Usage**

```
F2dfa(y, m = 3, nu = 0, overlap = TRUE)
```

**Arguments**

y	vector corresponding to the time series data.
m	an integer or integer valued vector indicating the size (or sizes) of the window for the polinomial fit. $\min(m)$ must be greater or equal than <i>nu</i> or else it will return an error.
nu	a non-negative integer denoting the degree of the polinomial fit applied on the integrated series.
overlap	logical: if true (the default), uses overlapping windows. Otherwise, non-overlapping boxes are applied.

**Value**

A vector of size  $\text{length}(m)$  containing the detrended variance considering windows of size  $m + 1$ , for each  $m$  supplied.

**Author(s)**

Taiane Schaedler Prass

**References**

Prass, T.S. and Pumi, G. (2019). On the behavior of the DFA and DCCA in trend-stationary processes <arXiv:1910.10589>.

**Examples**

```
# Simple usage
y = rnorm(100)
F2.dfa = F2dfa(y, m = 3, nu = 0, overlap = TRUE)
F2.dfa

vF2.dfa = F2dfa(y, m = 3:5, nu = 0, overlap = TRUE)
vF2.dfa

#####
# AR(1) example showing how the DFA varies with phi

phi = (1:8)/10
n = 300
z = matrix(nrow = n, ncol = length(phi))
for(i in 1:length(phi)){
  z[,i] = arima.sim(model = list(ar = phi[i]), n)
}

ms = 3:50
F2.dfa = matrix(ncol = length(phi), nrow = length(ms))

for(j in 1:length(phi)){
  F2.dfa[,j] = F2dfa(z[,j], m = ms , nu = 0, overlap = TRUE)
```



```

}

cr = rainbow(length(phi))
plot(ms, F2.dfa[,1], type = "o", xlab = "m", col = cr[1],
      ylim = c(0,max(F2.dfa)), ylab = "F2.dfa")
for(j in 2:length(phi)){
  points(ms, F2.dfa[,j], type = "o", col = cr[j])
}
legend("topleft", lty = 1, legend = phi, col = cr, bty = "n", title = expression(phi), pch=1)

#####
# An MA(2) example showcasing why overlapping windows are usually advantageous
n = 300
ms = 3:50
theta = c(0.4,0.5)

# Calculating the expected value of the DFA in this scenario
m_max = max(ms)
vtheta = c(c(1,theta, rep(0, m_max - length(theta))))
G = matrix(0, ncol = m_max+1, nrow = m_max+1)
for(t in 1:(m_max+1)){
  for(h in 0:(m_max+1-t)){
    G[t,t+h] = sum(vtheta[1:(length(vtheta)-h)]*vtheta[(1+h):length(vtheta)])
    G[t+h,t] = G[t,t+h]
  }
}

EF2.dfa = EF2dfa(m = ms, nu = 0, G = G)

z = arima.sim(model = list(ma = theta), n)

ms = 3:50
OF2.dfa = F2dfa(z, m = ms, nu = 0, overlap = TRUE)
NOF2.dfa = F2dfa(z, m = ms, nu = 0, overlap = FALSE)

plot(ms, OF2.dfa, type = "o", xlab = "m", col = "blue",
      ylim = c(0,max(OF2.dfa,NOF2.dfa,EF2.dfa)), ylab = "F2.dfa")
points(ms, NOF2.dfa, type = "o", col = "darkgreen")
points(ms, EF2.dfa, type = "o", col = "red")
legend("bottomright", legend = c("overlapping","non-overlapping","expected"),
      col = c("blue", "darkgreen","red"), lty= 1, bty = "n", pch=1)

```

**Description**

Calculates the detrended cross-covariance between two time series  $y_1$  and  $y_2$ .

**Usage**

```
Fdcca(y1, y2, m = 3, nu = 0, overlap = TRUE)
```

**Arguments**

<code>y1, y2</code>	vectors corresponding to the time series data. If $length(y1)$ and $length(y2)$ differ, the longer time series is coerced to match the length of the shorter.
<code>m</code>	an integer or integer valued vector indicating the size (or sizes) of the window for the polynomial fit. $min(m)$ must be greater or equal than $nu$ or else it will return an error.
<code>nu</code>	a non-negative integer denoting the degree of the polynomial fit applied on the integrated series.
<code>overlap</code>	logical: if true (the default), uses overlapping windows. Otherwise, non-overlapping boxes are applied.

**Value**

A vector of size  $length(m)$  containing the detrended cross-covariance considering windows of size  $m + 1$ , for each  $m$  supplied.

**Author(s)**

Taiane Schaedler Prass

**References**

Prass, T.S. and Pumi, G. (2019). On the behavior of the DFA and DCCA in trend-stationary processes <arXiv:1910.10589>.

**Examples**

```
# Simple usage
y1 = rnorm(100)
y2 = rnorm(100)
F.dcca = Fdcca(y1, y2, m = 3, nu = 0, overlap = TRUE)
F.dcca

# A simple example where y1 and y2 are independent.

ms = 3:50
F.dcca1 = Fdcca(y1, y2, m = ms, nu = 0, overlap = TRUE)
F.dcca2 = Fdcca(y1, y2, m = ms, nu = 0, overlap = FALSE)

plot(ms, F.dcca1, type = "o", xlab = "m", col = "blue",
      ylim = c(min(F.dcca1, F.dcca2), max(F.dcca1, F.dcca2)),
      ylab = expression(F[DCCA]))
points(ms, F.dcca2, type = "o", col = "red")
legend("bottomright", legend = c("overlapping", "non-overlapping"),
      col = c("blue", "red"), lty= 1, bty = "n", pch=1)
```

```

# A more elaborated example where y1 and y2 display cross-correlation for non-null lags.
# This example also showcases why overlapping windows are usually advantageous.
# The data generating process is the following:
# y1 is i.i.d. Gaussian while y2 is an MA(2) generated from y1.

n = 500
ms = 3:50
theta = c(0.4,0.5)

# Calculating the expected value of the DCCA in this scenario
m_max = max(ms)
vtheta = c(1,theta, rep(0, m_max - length(theta)))
G12 = matrix(0, ncol = m_max+1, nrow = m_max+1)
for(t in 1:(m_max+1)){
  for(h in 0:(m_max+1-t)){
    G12[t,t+h] = vtheta[h+1]
  }
}

EF.dcca = EFdcca(m = ms, nu = 0, G = G12)

# generating the series and calculating the DCCA
burn.in = 100
eps = rnorm(burn.in)

y1 = rnorm(n)
y2 = arima.sim(model = list(ma = theta), n, n.start = burn.in, innov = y1, start.innov = eps)

ms = 3:50
OF.dcca = Fdcca(y1, y2, m = ms, nu = 0, overlap = TRUE)
NOF.dcca = Fdcca(y1, y2, m = ms, nu = 0, overlap = FALSE)

plot(ms, OF.dcca, type = "o", xlab = "m", col = "blue",
      ylim = c(min(NOF.dcca,OF.dcca,EF.dcca),max(NOF.dcca,OF.dcca,EF.dcca)),
      ylab = expression(F[DCCA]))
points(ms, NOF.dcca, type = "o", col = "darkgreen")
points(ms, EF.dcca, type = "o", col = "red")
legend("bottomright", legend = c("overlapping","non-overlapping","expected"),
      col = c("blue", "darkgreen","red"), lty= 1, bty = "n", pch=1)

```

---

Jn

*Matrix J*


---

### Description

Creates a  $n$  by  $n$  lower triangular matrix with all non-zero entries equal to one.

### Usage

Jn( $n = 2$ )

**Arguments**

n number of rows and columns in the J matrix.

**Value**

an  $n$  by  $n$  lower triangular matrix with all non-zero entries equal to one. This is an auxiliary function.

**Examples**

```
J = Jn(n = 3)
J
```

---

 Kkronm

*The product of Kronecker Product of some Arrays*


---

**Description**

This is an auxiliary function and requires some context to be used adequately. It computes equation (19) in Prass and Pumi (2019), returning a square matrix defined by

$$K* = (Jm \% x \% J*)'(Q \% x \% Q)(Jm \% x \% J*)$$

where:

- $J$  is an  $(m + 1) * (h + 1) - m * h * s$  by  $(m + 1) * (h + 1) - m * h * s$  lower triangular matrix with all non-zero entries equal to one, with  $s = 1$  if `overlap = TRUE` and  $s = 0$ , otherwise;
- $Jm$  corresponds to the first  $m + 1$  rows and columns of  $J$ ;
- $J*$  corresponds to the last  $m + 1$  rows of  $J$ ;
- $Q = I - P$ , where  $P$  is the  $m + 1$  by  $m + 1$  projection matrix into the subspace generated by degree  $nu + 1$  polynomials.

**Usage**

```
Kkronm(m = 3, nu = 0, h = 0, overlap = TRUE, K = NULL)
```

**Arguments**

m a positive integer indicating the size of the window for the polinomial fit.

nu a non-negative integer denoting the degree of the polinomial fit applied on the integrated series.

h an integer indicating the lag.

overlap logical: if true (the default), overlapping boxes are used for calculations. Otherwise, non-overlapping boxes are applied.

K optional: the matrix defined by  $K = J'QJ$ . This is used to calculate  $K* = (Jm \% x \% J*)'(Q \% x \% Q)(Jm \% x \% J*)$ . For details see (19) in Prass and Pumi (2019). If this matrix is provided  $mu$  is ignored.

**Value**

an  $(m + 1)[(m + 1) * (h + 1) - m * h * s]$  by  $(m + 1)[(m + 1) * (h + 1) - m * h * s]$  matrix, where  $s = 1$  if `overlap = TRUE` and  $s = 0$ , otherwise. This matrix corresponds to equation (19) in Prass and Pumi (2019).

**Author(s)**

Taiane Schaedler Prass

**References**

Prass, T.S. and Pumi, G. (2019). On the behavior of the DFA and DCCA in trend-stationary processes <arXiv:1910.10589>.

**See Also**

`Jn` which creates the matrix  $J$ , `Qm` which creates  $Q$  and `Km` which creates  $K$ .

**Examples**

```
m = 3
h = 1
J = Jn(n = m+1+h)
Q = Qm(m = m, nu = 0)

# using K
K = Km(J = J[1:(m+1),1:(m+1)], Q = Q)
Kkron0 = Kkronm(K = K, h = h)

# using m and nu
Kkron = Kkronm(m = m, nu = 0, h = h)

# using kronecker product from R
K = Km(J = J[1:(m+1),1:(m+1)], Q = Q)
Kh = rbind(matrix(0, nrow = h, ncol = m+1+h),
            cbind(matrix(0, nrow = m+1, ncol = h), K))
KkronR = K %x% Kh

# using the definition K* = (Jm %x% J)'(Q %x% Q)(Jm %x% J)
J_m = J[1:(m+1),1:(m+1)]
J_h = J[(h+1):(m+1+h),1:(m+1+h)]
KkronD = t(J_m %x% J_h)%x%(Q %x% Q)%x%(J_m %x% J_h)

# comparing the results
sum(abs(Kkron0 - Kkron))
sum(abs(Kkron0 - KkronR))
sum(abs(Kkron0 - KkronD)) # difference due to rounding error

## Not run:
# Function Kkronm is computationally faster than a pure implementation in R:
```

```

m = 100
h = 1
J = Jn(n = m+1)
Q = Qm(m = m, nu = 0)

# using Kkronm
t1 = proc.time()
Kkron = Kkronm(m = m, nu = 0, h = 1)
t2 = proc.time()
# elapsed time:
t2-t1

# Pure R implementation:
K = Km(J = J, Q = Q)
Kh = rbind(matrix(0, nrow = h, ncol = m+1+h),
            cbind(matrix(0, nrow = m+1, ncol = h), K))
t3 = proc.time()
KkronR = K %% Kh
t4 = proc.time()
# elapsed time
t4-t3

## End(Not run)

```

---

Km

*Matrix K*


---

### Description

This is an auxiliary function which computes expression (18) in Prass and Pumi (2019). It creates an  $m + 1$  by  $m + 1$  matrix defined by  $K = J'QJ$  where  $J$  is a  $m + 1$  by  $m + 1$  lower triangular matrix with all non-zero entries equal to one and  $Q$  is a  $m + 1$  by  $m + 1$  given by  $Q = I - P$  where  $P$  is the projection matrix into the subspace generated by degree  $nu + 1$  polynomials and  $I$  is the  $m + 1$  by  $m + 1$  identity matrix.

### Usage

```
Km(m = 3, nu = 0, J = NULL, Q = NULL)
```

### Arguments

m	a positive integer greater or equal than $nu$ indicating the size of the window for the polynomial fit.
nu	a non-negative integer denoting the degree of the polynomial fit applied on the integrated series.
J, Q	optional: the matrices such that $K = J'QJ$ . If both matrices are provided, $m$ and $nu$ are ignored.

**Value**

an  $m + 1$  by  $m + 1$  matrix corresponding to expression (18) in Prass and Pumi (2019).

**Author(s)**

Taiane Schaedler Prass

**References**

Prass, T.S. and Pumi, G. (2019). On the behavior of the DFA and DCCA in trend-stationary processes <arXiv:1910.10589>.

**See Also**

[Jn](#) which creates the matrix  $J$ , [Qm](#) which creates  $Q$  and [Pm](#) which creates  $P$ .

**Examples**

```
K = Km(m = 3, nu = 0)
K
# same as
m = 3
J = Jn(n = m+1)
Q = Qm(m = m, nu = 0)
K = Km(J = J, Q = Q)
K
```

---

Pm

*Projection Matrix P*

---

**Description**

Creates the  $m + 1$  by  $m + 1$  projection matrix defined by  $P = D(D'D)^{-1}D'$  where  $D$  is the design matrix associated to a polynomial regression of degree  $nu + 1$ .

**Usage**

```
Pm(m = 2, nu = 0)
```

**Arguments**

`nu` the degree of the polynomial fit.  
`m` a positive integer satisfying  $m \geq nu$  indicating the size of the window for the polynomial fit.

**Details**

To perform matrix inversion, the code makes use of the routine DGETRI in LAPACK, which applies an LU decomposition approach to obtain the inverse matrix. See the LAPACK documentation available at <http://www.netlib.org/lapack>.

**Value**

an  $m + 1$  by  $m + 1$  matrix.

**Author(s)**

Taiane Schaedler Prass

**Examples**

```
P = Pm(m = 5, nu = 0)
P

n = 10
t = 1:n
D = cbind(rep(1,n),t,t^2)

# Calculating in R
PR = D%%solve(t(D)%*%D)%*%t(D)
# Using the provided function
P = Pm(m = n-1, nu = 1)

# Difference:
sum(abs(P-PR))
```

---

Qm

*Projection Matrix Q*

---

**Description**

Creates the  $m + 1$  by  $m + 1$  projection matrix defined by  $Q = I - P$  where  $I$  is the the  $m + 1$  by  $m + 1$  identity matrix and  $P$  is the  $m + 1$  by  $m + 1$  projection matrix into the space generated by polynomials of degree  $nu + 1$ .

**Usage**

```
Qm(m = 2, nu = 0, P = NULL)
```



**Arguments**

nu	the degree of the polynomial fit.
m	a positive integer satisfying $m \geq nu$ indicating the size of the window for the polynomial fit.
P	optional: the projection matrix such that $Q = I - P$ (see function <code>Pm</code> ). If this matrix is provided $m$ and $nu$ are ignored.

**Value**

an  $m + 1$  by  $m + 1$  matrix.

**See Also**

`Pm` which generates the projection matrix  $P$ .

**Examples**

```
Q = Qm(m = 3, nu = 0)
Q
# same as
P = Pm(m = 3, nu = 0)
Q = Qm(P = P)
Q
```

---

 rhodcca

*Detrended Cross-correlation coefficient*


---

**Description**

Calculates the detrended cross-correlation coefficient for two time series  $y1$  and  $y2$ .

**Usage**

```
rhodcca(y1, y2, m = 3, nu = 0, overlap = TRUE)
```

**Arguments**

$y1, y2$	vectors corresponding to the time series data. If $length(y1)$ and $length(y2)$ differ, the longer time series is coerced to match the length of the shorter.
m	an integer value or a vector of integer values indicating the size of the window for the polynomial fit. $min(m)$ must be greater or equal than $nu$ or else it will return an error.
nu	the degree of the polynomial fit
overlap	logical: if true (the default), uses overlapping windows. Otherwise, non-overlapping boxes are applied.

**Value**

A list containing the following elements, calculated considering windows of size  $m + 1$ , for each  $m$  supplied:

F2dfa1, F2dfa2    The detrended variances for  $y_1$  and  $y_2$ , respectively.  
 Fdcca            The detrended cross-covariance.  
 rhodcca         The detrended cross-correlation coefficient.

**Note**

The time series  $y_1$  and  $y_2$  must have the same sample size.

**Author(s)**

Taiane Schaedler Prass

**References**

Prass, T.S. and Pumi, G. (2019). On the behavior of the DFA and DCCA in trend-stationary processes <arXiv:1910.10589>.

**See Also**

[F2dfa](#) which calculated the DFA and [Fdcca](#) which calculated the DCCA of two given time series.

**Examples**

```
y1 = rnorm(100)
y2 = rnorm(100)
rho.dccam1 = rhodcca(y1, y2, m = 3, nu = 0, overlap = TRUE)
rho.dccam1

rho.dccam2 = rhodcca(y1, y2, m = c(3,6,8), nu = 0, overlap = TRUE)
rho.dccam2
```

---

rhoE

*The limit value of the detrended cross-covariance*

---

**Description**

Calculates the theoretical counterpart of the cross-correlation coefficient. This is expression (11) in Prass and Pumi (2019). For trend-stationary processes under mild assumptions, this is equivalent to the limit of the detrended cross correlation coefficient calculated with window of size  $m + 1$  as  $m$  tends to infinity (see theorem 3.2 in Prass and Pumi, 2019).

**Usage**

```
rhoE(m = 3, nu = 0, G1, G2, G12, K = NULL)
```

**Arguments**

m	an integer or integer valued vector indicating the size (or sizes) of the window for the polinomial fit. $\min(m)$ must be greater or equal than $nu$ or else it will return an error.
nu	a non-negative integer denoting the degree of the polinomial fit applied on the integrated series.
G1, G2	the autocovariance matrices for the original time series. Both are $\max(m) + 1$ by $\max(m) + 1$ matrices.
G12	the cross-covariance matrix for the original time series. The dimension of $G12$ must be $\max(m) + 1$ by $\max(m) + 1$ .
K	optional: the matrix $K$ . See the details.

**Details**

The optional argument  $K$  is an  $m + 1$  by  $m + 1$  matrix defined by  $K = J'QJ$ , where  $J$  is a  $m + 1$  by  $m + 1$  lower triangular matrix with all non-zero entries equal to one and  $Q$  is a  $m + 1$  by  $m + 1$  given by  $Q = I - P$  where  $P$  is the projection matrix into the subspace generated by degree  $nu + 1$  polynomials and  $I$  is the  $m + 1$  by  $m + 1$  identity matrix.  $K$  is equivalent to expression (18) in Prass and Pumi (2019). If this matrix is provided and  $m$  is an integer, then  $nu$  are ignored.

**Value**

A list containing the following elements, calculated considering windows of size  $m + 1$ , for each  $m$  supplied:

EF2dfa1, EF2dfa2	the expected values of the detrended variances.
EFdcca	the expected value of the detrended cross-covariance.
rhoE	the vector with the theoretical counterpart of the cross-correlation coefficient.

**Author(s)**

Taiane Schaedler Prass

**References**

Prass, T.S. and Pumi, G. (2019). On the behavior of the DFA and DCCA in trend-stationary processes <arXiv:1910.10589>.

**See Also**

[Km](#) which creates the matrix  $K$ , [Jn](#) which creates the matrix  $J$ , [Qm](#) which creates  $Q$  and [Pm](#) which creates  $P$ .

**Examples**

```
m = 3
K = Km(m = m, nu = 0)
G1 = G2 = diag(m+1)
G12 = matrix(0, ncol = m+1, nrow = m+1)
rhoE(G1 = G1, G2 = G2, G12 = G12, K = K)
# same as
rhoE(m = 3, nu = 0, G1 = G1, G2 = G2, G12 = G12)
```

# Index

covF2dfa, [2](#)  
covFdcca, [3](#)

EF2dfa, [5](#)  
EFdcca, [6](#)

F2dfa, [7](#), [18](#)  
Fdcca, [9](#), [18](#)

Jn, [11](#), [13](#), [15](#), [19](#)

Kkronm, [12](#)  
Km, [13](#), [14](#), [19](#)

Pm, [15](#), [15](#), [17](#), [19](#)

Qm, [13](#), [15](#), [16](#), [19](#)

rhodcca, [17](#)  
rhoE, [18](#)