R topics documented:

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>frobenius.prod</td>
<td>15</td>
</tr>
<tr>
<td>H.matrices</td>
<td>16</td>
</tr>
<tr>
<td>hadamard.prod</td>
<td>17</td>
</tr>
<tr>
<td>hankel.matrix</td>
<td>18</td>
</tr>
<tr>
<td>hilbert.matrix</td>
<td>19</td>
</tr>
<tr>
<td>hilbert.schmidt.norm</td>
<td>20</td>
</tr>
<tr>
<td>inf.norm</td>
<td>21</td>
</tr>
<tr>
<td>is.diagonal.matrix</td>
<td>22</td>
</tr>
<tr>
<td>is.idempotent.matrix</td>
<td>23</td>
</tr>
<tr>
<td>is.indefinite</td>
<td>24</td>
</tr>
<tr>
<td>is.negative.definite</td>
<td>26</td>
</tr>
<tr>
<td>is.negative.semi.definite</td>
<td>27</td>
</tr>
<tr>
<td>is.non.singular.matrix</td>
<td>29</td>
</tr>
<tr>
<td>is.positive.definite</td>
<td>30</td>
</tr>
<tr>
<td>is.positive.semi.definite</td>
<td>31</td>
</tr>
<tr>
<td>is.singular.matrix</td>
<td>33</td>
</tr>
<tr>
<td>is.skew.symmetric.matrix</td>
<td>34</td>
</tr>
<tr>
<td>is.square.matrix</td>
<td>35</td>
</tr>
<tr>
<td>is.symmetric.matrix</td>
<td>36</td>
</tr>
<tr>
<td>K.matrix</td>
<td>37</td>
</tr>
<tr>
<td>L.matrix</td>
<td>38</td>
</tr>
<tr>
<td>lower.triangle</td>
<td>39</td>
</tr>
<tr>
<td>lu.decomposition</td>
<td>40</td>
</tr>
<tr>
<td>matrix.inverse</td>
<td>41</td>
</tr>
<tr>
<td>matrix.power</td>
<td>42</td>
</tr>
<tr>
<td>matrix.rank</td>
<td>43</td>
</tr>
<tr>
<td>matrix.trace</td>
<td>44</td>
</tr>
<tr>
<td>maximum.norm</td>
<td>45</td>
</tr>
<tr>
<td>N.matrix</td>
<td>46</td>
</tr>
<tr>
<td>one.norm</td>
<td>47</td>
</tr>
<tr>
<td>pascal.matrix</td>
<td>48</td>
</tr>
<tr>
<td>set.submatrix</td>
<td>49</td>
</tr>
<tr>
<td>shift.down</td>
<td>50</td>
</tr>
<tr>
<td>shift.left</td>
<td>51</td>
</tr>
<tr>
<td>shift.right</td>
<td>52</td>
</tr>
<tr>
<td>shift.up</td>
<td>53</td>
</tr>
<tr>
<td>spectral.norm</td>
<td>54</td>
</tr>
<tr>
<td>stirling.matrix</td>
<td>55</td>
</tr>
<tr>
<td>svd.inverse</td>
<td>56</td>
</tr>
<tr>
<td>symmetric.pascal.matrix</td>
<td>57</td>
</tr>
<tr>
<td>T.matrices</td>
<td>58</td>
</tr>
<tr>
<td>toeplitz.matrix</td>
<td>59</td>
</tr>
<tr>
<td>u.vectors</td>
<td>60</td>
</tr>
<tr>
<td>upper.triangle</td>
<td>61</td>
</tr>
<tr>
<td>vandermonde.matrix</td>
<td>62</td>
</tr>
<tr>
<td>vec</td>
<td>63</td>
</tr>
<tr>
<td>vech</td>
<td>63</td>
</tr>
<tr>
<td>%s%</td>
<td>64</td>
</tr>
</tbody>
</table>
The commutation matrix

Index 66

---

**commutation.matrix**

**Commutation matrix for r by c numeric matrices**

**Description**

This function returns a square matrix of order \( p = r \times c \) that, for an \( r \times c \) matrix \( A \), transforms \( \text{vec}(A) \) to \( \text{vec}(A') \) where prime denotes transpose.

**Usage**

```r
commutation.matrix(r, c=r)
```

**Arguments**

- \( r \)  
  a positive integer integer row dimension

- \( c \)  
  a positive integer integer column dimension

**Details**

This function is a wrapper function that uses the function \( K.matrix \) to do the actual work. The \( r \times c \) matrices \( H_{i,j} \) constructed by the function \( h.matrices \) are combined using direct product to generate the commutation product with the following formula:

\[
K_{r,c} = \sum_{i=1}^{r} \sum_{j=1}^{c} (H_{i,j} \otimes H'_{i,j})
\]

**Value**

An order \((r \times c)\) matrix.

**Note**

If either argument is less than 2, then the function stops and displays an appropriate error message. If either argument is not an integer, then the function stops and displays an appropriate error message.

**Author(s)**

Frederick Novomestky <fnovomes@poly.edu>

**References**


**See Also**

*H.matrices, K.matrix*
Examples

```r
K <- commutation.matrix( 3, 4 )
A <- matrix( seq( 1, 12, 1 ), nrow=3, byrow=TRUE )
vecA <- vec( A )
vecAt <- vec( t(A) )
print( K %*% vecA )
print( vecAt )
```

**creation.matrix**  

**Creation Matrix**

**Description**

This function returns the order n creation matrix, a square matrix with the sequence 1, 2, ..., n - 1 on the sub-diagonal below the principal diagonal.

**Usage**

`creation.matrix(n)`

**Arguments**

- `n` a positive integer greater than 1

**Details**

The order n creation matrix is also called the derivation matrix and is used in numerical mathematics and physics. It arises in the solution of linear dynamical systems. The form of the matrix is

```
\[
\begin{pmatrix}
0 & 0 & \cdots & 0 & 0 \\
1 & 0 & \cdots & 0 & 0 \\
0 & 2 & \cdots & 0 & 0 \\
0 & 0 & 3 & \cdots & 0 \\
\vdots & \vdots & \ddots & \ddots & \ddots \\
0 & 0 & 0 & \cdots & n - 1 \\
\end{pmatrix}
\]
```

**Value**

An order n matrix.

**Note**

If the argument n is not an integer that is greater than 1, the function presents an error message and stops.

**Author(s)**

Frederick Novomestky <fnovomes@poly.edu>
D.matrix

References


Examples

```r
H <- creation.matrix( 10 )
print( H )
```

---

D.matrix  Duplication matrix

Description

This function constructs the linear transformation D that maps vech(A) to vec(A) when A is a symmetric matrix.

Usage

D.matrix(n)

Arguments

n  a positive integer value for the order of the underlying matrix

Details

Let $T_{i,j}$ be an $n \times n$ matrix with 1 in its $(i,j)$ element $1 \leq i, j \leq n.$ and zeroes elsewhere. These matrices are constructed by the function `T.matrices`. The formula for the transpose of matrix D is $D' = \sum_{j=1}^{n} \sum_{i=j}^{n} u_{i,j} (\text{vec } T_{i,j})' \text{ where } u_{i,j}$ is the column vector in the order $\frac{1}{2} n (n+1)$ identity matrix for column $k = (j-1) n + i - \frac{1}{2} j (j-1)$. The function `u.vectors` generates these vectors.

Value

It returns an $n^2 \times \frac{1}{2} n (n+1)$ matrix.

Author(s)

Frederick Novomestky <fnovomes@poly.edu>

References


See Also

T.matrices, u.vectors

Examples

```r
D <- D.matrix(3)
A <- matrix(c(1, 2, 3,
           2, 3, 4,
           3, 4, 5), nrow=3, byrow=TRUE)
vecA <- vec(A)
vechA <- vech(A)
y <- D %*% vechA
print(y)
print(vecA)
```


direct.prod Direct prod of two arrays

direct.prod(x, y)

Arguments

- `x`: a numeric matrix or vector
- `y`: a numeric matrix or vector

Details

If either `x` or `y` is a vector, it is converted to a matrix. Suppose that `x` is an \(m \times n\) matrix and `y` is an \(p \times q\) matrix. Then, the function returns the matrix

\[
\begin{bmatrix}
  x_{1,1} & y & x_{1,2} & \cdots & x_{1,n} & y \\
  x_{2,1} & y & x_{2,2} & \cdots & x_{2,n} & y \\
  \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
  x_{m,1} & y & x_{m,2} & \cdots & x_{m,n} & y
\end{bmatrix}
\]

Value

A numeric matrix.

Author(s)

Frederick Novomestky <fnovomes@poly.edu>, Kurt Hornik <Kurt.Hornik@wu-wien.ac.at>
direct.sum

References


Examples

```r
x <- matrix( seq( 1, 4 ) )
y <- matrix( seq( 5, 8 ) )
print( direct.prod( x, y ) )
```

---

direct.sum  

*Direct sum of two arrays*

Description

This function computes the direct sum of two arrays. The arrays can be numerical vectors or matrices. The result is the block diagonal matrix

\[
\begin{bmatrix}
x & 0 \\
0 & y
\end{bmatrix}
\]

Usage

```r
direct.sum( x, y )
```

Arguments

- **x**: a numeric matrix or vector
- **y**: a numeric matrix or vector

Details

If either `x` or `y` is a vector, it is converted to a matrix. The result is a block diagonal matrix

\[
\begin{bmatrix}
x & 0 \\
0 & y
\end{bmatrix}
\]

Value

A numeric matrix.

Author(s)

Frederick Novomestky <fnovomes@poly.edu>, Kurt Hornik <Kurt.Hornik@wu-wien.ac.at>

References

duplication.matrix

Examples

x <- matrix( seq(1, 4) )
y <- matrix( seq(5, 8) )
print( direct.sum( x, y ) )

duplication.matrix  
Duplication matrix for n by n matrices

Description

This function returns a matrix with n * n rows and n * (n + 1) / 2 columns that transforms vech(A) to vec(A) where A is a symmetric n by n matrix.

Usage

duplication.matrix(n=1)

Arguments

n  
Row and column dimension

Details

This function is a wrapper function for the function D.matrix. Let $T_{i,j}$ be an $n \times n$ matrix with 1 in its $(i,j)$ element $1 \leq i,j \leq n$ and zeroes elsewhere. These matrices are constructed by the function T.matrices. The formula for the transpose of matrix $D$ is $D' = \sum_{j=1}^{n} \sum_{i=j}^{n} u_{i,j} \left( vec T_{i,j} \right)'$ where $u_{i,j}$ is the column vector in the order $1 2 n (n + 1)$ identity matrix for column $k = (j - 1) n + i - \frac{1}{2} j (j - 1)$. The function u.vectors generates these vectors.

Value

It returns an $n^2 \times \frac{1}{2} n (n + 1)$ matrix.

Author(s)

Frederick Novomestky <fnovomes@poly.edu>, Kurt Hornik <Kurt.Hornik@wu-wien.ac.at>

References


See Also

D.matrix, vec, vech
Examples

```r
d <- duplication.matrix( 3 )
A <- matrix( c( 1, 2, 3, 
              2, 3, 4, 
              3, 4, 5 ), nrow=3, byrow=TRUE )
vecA <- vec( A )
vecA <- vech( A )
y <- d %*% vecA
print( y )
print( vecA )
```

---

**E.matrices**

*List of E Matrices*

**Description**

This function constructs and returns a list of lists. The component of each sublist is a square matrix derived from the column vectors of an order \( n \) identity matrix.

**Usage**

`E.matrices(n)`

**Arguments**

- `n` a positive integer for the order of the identity matrix

**Details**

Let \( \mathbf{I}_n = [ \mathbf{e}_1 \ \mathbf{e}_2 \ \cdots \ \mathbf{e}_n ] \) be the order \( n \) identity matrix with corresponding unit vectors \( \mathbf{e}_i \) with one in its \( i \)th position and zeros elsewhere. The \( n \times n \) matrix \( \mathbf{E}_{i,j} \) is computed from the unit vectors \( \mathbf{e}_i \) and \( \mathbf{e}_j \) as \( \mathbf{E}_{i,j} = \mathbf{e}_i \mathbf{e}_j' \). These matrices are stored as components in a list of lists.

**Value**

A list with \( n \) components

1. A sublist of \( n \) components
2. A sublist of \( n \) components
   ...
\( n \) A sublist of \( n \) components

Each component \( j \) of sublist \( i \) is a matrix \( \mathbf{E}_{i,j} \)

**Note**

The argument \( n \) must be an integer value greater than or equal to 2.
Author(s)
Frederick Novomestky <fnovomes@poly.edu>

References


Examples
```r
e <- eMatricesH S I
```

---

### elimination.matrix

Elimination matrix for lower triangular matrices

#### Description
This function returns a matrix with \( n \times (n + 1) / 2 \) rows and \( N \times n \) columns which for any lower triangular matrix \( A \) transforms \( \text{vec}(A) \) into \( \text{vech}(A) \)

#### Usage
```r
elimination.matrix(n)
```

#### Arguments
- `n` row or column dimension

#### Details
This function is a wrapper function to the function `L.matrix`. The formula used to compute the \( L \) matrix which is also called the elimination matrix is

\[
L = \sum_{j=1}^{n} \sum_{i=j}^{n} u_{i,j} (\text{vec} E_{i,j})' \]

\( u_{i,j} \) are the order \( n \times (n + 1) / 2 \) vectors constructed by the function `u.vectors`. \( E_{i,j} \) are the \( n \times n \) matrices constructed by the function `eMatrices`.

#### Value
An \( \left[ \frac{1}{2} n (n + 1) \right] \times n^2 \) matrix.

#### Note
If the argument is not an integer, the function displays an error message and stops. If the argument is less than two, the function displays an error message and stops.
**entrywise.norm**

**Author(s)**

Frederick Novomestky <fnovomes@poly.edu>

**References**


**See Also**

`E.matrices`, `L.matrix`, `u.vectors`

**Examples**

```r
L <- elimination.matrix(4)
A <- lower.triangle(matrix(seq(1, 16, 1), nrow=4, byrow=TRUE))
vecA <- vec(A)
vechA <- vech(A)
y <- L %*% vecA
print(y)
print(vechA)
```

---

**entrywise.norm**  
*Compute the entrywise norm of a matrix*

**Description**

This function returns the $\|x\|_p$ norm of the matrix $x$.

**Usage**

```r
entrywise.norm(x, p)
```

**Arguments**

- `x`  
a numeric vector or matrix
- `p`  
a real value for the power

**Details**

Let $x$ be an $m \times n$ numeric matrix. The formula used to compute the norm is $\|x\|_p = \left( \sum_{i=1}^{m} \sum_{j=1}^{n} |x_{i,j}|^p \right)^{1/p}$.

**Value**

A numeric value.
Note
If argument x is not numeric, the function displays an error message and terminates. If argument x
is neither a matrix nor a vector, the function displays an error message and terminates. If argument
p is zero, the function displays an error message and terminates.

Author(s)
Frederick Novomestky <fnovomes@poly.edu>

References
Industrial and Applied Mathematics.
University Press.

See Also
one.norm, inf.norm

Examples
```r
A <- matrix(c(3, 5, 7, 2, 6, 4, 0, 2, 8), nrow=3, ncol=3, byrow=TRUE)
print(entrywise.norm(A, 2))
```

fibonacci.matrix  Fibonacci Matrix

Description
This function constructs the order n + 1 square Fibonacci matrix which is derived from a Fibonacci
sequence.

Usage
fibonacci.matrix(n)

Arguments
- `n`: a positive integer value

Details
Let \( \{f_0, f_1, \ldots, f_n\} \) be the set of \( n + 1 \) Fibonacci numbers where \( f_0 = f_1 = 1 \) and \( f_j = f_{j-1} + f_{j-2}, \quad 2 \leq j \leq n \). The order \( n + 1 \) Fibonacci matrix \( F \) has as typical element \( F_{i,j} = \begin{cases} f_{i-j+1} & i - j + 1 \geq 0 \\ 0 & i - j + 1 < 0 \end{cases} \).
**frobenius.matrix**

**Value**

An order $n + 1$ matrix

**Note**

If the argument $n$ is not a positive integer, the function presents an error message and stops.

**Author(s)**

Frederick Novomestky <fnovomes@poly.edu>

**References**


**Examples**

```r
F <- fibonacci.matrix(10)
print(F)
```

---

**frobenius.matrix**  
**Frobenius Matrix**

**Description**

This function returns an order $n$ Frobenius matrix that is useful in numerical mathematics.

**Usage**

`frobenius.matrix(n)`

**Arguments**

- `n`: a positive integer value greater than 1

**Details**

The Frobenius matrix is also called the companion matrix. It arises in the solution of systems of linear first order differential equations. The formula for the order $n$ Frobenius matrix is $F = \begin{bmatrix} 0 & 0 & \cdots & 0 & (-1)^{n-1} \binom{n}{0} \\ 1 & 0 & \cdots & 0 & (-1)^{n-2} \binom{n}{1} \\ 0 & 1 & \cdots & 0 & (-1)^{n-3} \binom{n}{2} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & (-1)^0 \binom{n}{n-1} \end{bmatrix}$. 


Value

An order \( n \) matrix

Note

If the argument \( n \) is not a positive integer that is greater than 1, the function presents an error message and stops.

Author(s)

Frederick Novomestky <fnovomes@poly.edu>

References


Examples

\[
F <- \text{frobenius.matrix}(10)
\]
\[
\text{print}(F)
\]

```
> Frobenius.norm

Compute the Frobenius norm of a matrix

Description

This function returns the Frobenius norm of the matrix \( x \).

Usage

\[
\text{frobenius.norm}(x)
\]

Arguments

\( x \) a numeric vector or matrix

Details

The formula used to compute the norm is \( \| x \|_2 \). Note that this is the entrywise norm with exponent 2.

Value

A numeric value.

Author(s)

Frederick Novomestky <fnovomes@poly.edu>
**frobenius.prod**

**References**


**See Also**

`entrywise.norm`

**Examples**

```r
A <- matrix(c(3, 5, 7, 2, 6, 4, 0, 2, 8), nrow=3, ncol=3, byrow=TRUE)
print(frobenius.norm(A))
```

---

**Description**

This function returns the Frobenius inner product of two matrices, `x` and `y`, with the same row and column dimensions.

**Usage**

`frobenius.prod(x, y)`

**Arguments**

- `x`: a numeric matrix or vector object
- `y`: a numeric matrix or vector object

**Details**

The Frobenius inner product is the element-by-element sum of the Hadamard or Shur product of two numeric matrices. Let `x` and `y` be two `m × n` matrices. Then Frobenious inner product is computed as $\sum_{i=1}^{m} \sum_{j=1}^{n} x_{i,j} y_{i,j}$.

**Value**

A numeric value.
Note

The function converts vectors to matrices if necessary. The function stops running if x or y is not numeric and an error message is displayed. The function also stops running if x and y do not have the same row and column dimensions and an error message is displayed.

Author(s)

Frederick Novomestky <fnovomes@poly.edu>

References


See Also

hadamard.prod

Examples

```r
x <- matrix(c(1, 2, 3, 4), nrow=2, byrow=TRUE)
y <- matrix(c(2, 4, 6, 8), nrow=2, byrow=TRUE)
z <- frobenius.prod(x, y)
print(z)
```

Description

This function constructs and returns a list of lists. The component of each sublist is derived from column vectors in an order r and order c identity matrix.

Usage

`H.matrices(r, c = r)`

Arguments

- `r` : a positive integer value for an order r identity matrix
- `c` : a positive integer value for an order c identity matrix

Details

Let $I_r = [ \begin{array}{ccccc} a_1 & a_2 & \cdots & a_r \end{array} ]$ be the order $r$ identity matrix with corresponding unit vectors $a_i$ with one in its $i$th position and zeros elsewhere. Let $I_c = [ \begin{array}{ccccc} b_1 & b_2 & \cdots & b_c \end{array} ]$ be the order $c$ identity matrix with corresponding unit vectors $b_i$ with one in its $i$th position and zeros elsewhere. The $r \times c$ matrix $H_{i,j} = a_i b'_j$ is used in the computation of the commutation matrix.
Value
A list with \( r \) components

1 A sublist of \( c \) components
2 A sublist of \( c \) components
...
\( r \) A sublist of \( c \) components

Each component \( j \) of sublist \( i \) is a matrix \( H_{i,j} \)

Note
The argument \( n \) must be an integer value greater than or equal to two.

Author(s)
Frederick Novomestky \(<\text{fnovomes@poly.edu}>\)

References

Examples
\[
\begin{align*}
H.2.3 & \leftarrow \text{h.matrices}(2,3) \\
H.3 & \leftarrow \text{h.matrices}(3)
\end{align*}
\]

---

**hadamard.prod**

*Hadamard product of two matrices*

Description
This function returns the Hadamard or Shur product of two matrices, \( x \) and \( y \), that have the same row and column dimensions.

Usage
\[
\text{hadamard.prod}(x, y)
\]

Arguments
\[
\begin{align*}
x & \text{ a numeric matrix or vector object} \\
y & \text{ a numeric matrix or vector object}
\end{align*}
\]
Details

The Hadamard product is an element-by-element product of the two matrices. Let \( x \) and \( y \) be two \( m \times n \) numeric matrices. The Hadamard product is \( x \circ y = \begin{bmatrix} x_{1,1} y_{1,1} & x_{1,2} y_{1,2} & \cdots & x_{1,n} y_{1,n} \\ x_{2,1} y_{2,1} & x_{2,2} y_{2,2} & \cdots & x_{2,n} y_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m,1} y_{m,1} & x_{m,2} y_{m,2} & \cdots & x_{m,n} y_{m,n} \end{bmatrix} \).

It uses the \( \ast \) operation in R.

Value

A matrix.

Note

The function converts vectors to matrices if necessary. The function stops running if \( x \) or \( y \) is not numeric and an error message is displayed. The function also stops running if \( x \) and \( y \) do not have the same row and column dimensions and an error message is displayed.

Author(s)

Frederick Novomestky <fnovomes@poly.edu>

References


Examples

```r
x <- matrix(c(1, 2, 3, 4), nrow=2, byrow=TRUE)
y <- matrix(c(2, 4, 6, 8), nrow=2, byrow=TRUE)
z <- hadamard.prod(x, y)
print(z)
```

---

**hankel.matrix**

**Hankel Matrix**

Description

This function constructs an order \( n \) Hankel matrix from the values in the order \( n \) vector \( x \). Each row of the matrix is a circular shift of the values in the previous row.

Usage

`hankel.matrix(n, x)`
hilbert.matrix

Arguments

- n: a positive integer value for order of matrix greater than 1
- x: a vector of values used to construct the matrix

Details

A Hankel matrix is a square matrix with constant skew diagonals. The determinant of a Hankel matrix is called a catalecticant. Hankel matrices are formed when the hidden Mark model is sought from a given sequence of data.

Value

An order n matrix.

Note

If the argument n is not a positive integer, the function presents an error message and stops. If the length of x is less than n, the function presents an error message and stops.

Author(s)

Frederick Novomestky <fnovomes@poly.edu>

References


Examples

```r
H <- hankel.matrix(4, seq(1, 7))
print(H)
```

hilbert.matrix

Hilbert matrices

Description

This function returns an n by n Hilbert matrix.

Usage

`hilbert.matrix(n)`

Arguments

- n: Order of the Hilbert matrix
Details
A Hilbert matrix is an order n square matrix of unit fractions with elements defined as \( H_{i,j} = \frac{1}{i+j-1} \).

Value
A matrix.

Note
If the argument is less than or equal to zero, the function displays an error message and stops. If the argument is not an integer, the function displays an error message and stops.

Author(s)
Frederick Novomestky <fnovomes@poly.edu>

References

Examples
\begin{verbatim}
H <- hilbert.matrix( 4 )
print( H )
\end{verbatim}

hilbert.schmidt.norm
Compute the Hilbert-Schmidt norm of a matrix

Description
This function returns the Hilbert-Schmidt norm of the matrix \( x \).

Usage
hilbert.schmidt.norm(x)

Arguments
x a numeric vector or matrix

Details
The formula used to compute the norm is \( \| x \|_2 \). This is merely the entrywise norm with exponent 2.
inf.norm

Value
A numeric value.

Author(s)
Frederick Novomestky <fnovomes@poly.edu>

References

See Also
entrywise.norm

Examples
A <- matrix( c( 3, 5, 7, 2, 6, 4, 0, 2, 8 ), nrow=3, ncol=3, byrow=TRUE )
print( hilbert.schmidt.norm( A ) )

inf.norm

Compute the infinity norm of a matrix

Description
This function returns the \( \| x \|_\infty \) norm of the matrix \( x \).

Usage
inf.norm(x)

Arguments
x a numeric vector or matrix

Details
Let \( x \) be an \( m \times n \) numeric matrix. The formula used to compute the norm is
\[
\| x \|_\infty = \max_{1 \leq i \leq m} \sum_{j=1}^{n} |x_{i,j}|.
\]
This is merely the maximum absolute row sum of the \( m \times n \) matrix.

Value
A numeric value.
is.diagonal.matrix

Author(s)
Frederick Novomestky <fnovomes@poly.edu>

References

See Also
one.norm

Examples
```r
A <- matrix(c(3, 5, 7, 2, 6, 4, 0, 2, 8), nrow=3, ncol=3, byrow=TRUE)
print(Inf.norm(A))
```

Description
This function returns TRUE if the given matrix argument x is a square numeric matrix and that the off-diagonal elements are close to zero in absolute value to within the given tolerance level. Otherwise, a FALSE value is returned.

Usage
```r
is.diagonal.matrix(x, tol = 1e-08)
```

Arguments
- `x` a numeric square matrix
- `tol` a numeric tolerance level usually left out

Value
A TRUE or FALSE value.

Author(s)
Frederick Novomestky <fnovomes@poly.edu>
is.idempotent.matrix

References


Examples

```r
A <- diag(1, 3)
is.diagonal.matrix(A)
B <- matrix(c(1, 2, 3, 4), nrow=2, byrow=TRUE)
is.diagonal.matrix(B)
C <- matrix(c(1, 0, 0, 0), nrow=2, byrow=TRUE)
is.diagonal.matrix(C)
```

is.idempotent.matrix  Test for idempotent square matrix

Description

This function returns a TRUE value if the square matrix argument x is idempotent, that is, the product of the matrix with itself is the matrix. The equality test is performed to within the specified tolerance level. If the matrix is not idempotent, then a FALSE value is returned.

Usage

```r
is.idempotent.matrix(x, tol = 1e-08)
```

Arguments

- `x`: a numeric square matrix
- `tol`: a numeric tolerance level usually left out

Details

Idempotent matrices are used in econometric analysis. Consider the problem of estimating the regression parameters of a standard linear model \( y = X \beta + e \) using the method of least squares. \( y \) is an order \( m \) random vector of dependent variables. \( X \) is an \( m \times n \) matrix whose columns are columns of observations on one of the \( n - 1 \) independent variables. The first column contains \( m \) ones. \( e \) is an order \( m \) random vector of zero mean residual values. \( \beta \) is the order \( n \) vector of regression parameters. The objective function that is minimized in the method of least squares is \( (y - X \beta)'(y - X \beta) \). The solution to this quadratic programming problem is \( \hat{\beta} = \left( X' X \right)^{-1} X' \hat{y} \). The corresponding estimator for the residual vector is \( \hat{e} = y - X \hat{\beta} = \left[ I - X (X' X)^{-1} X' \right] y = M y \). \( M \) and \( X (X' X)^{-1} X' \) are idempotent. Idempotency of \( M \) enters into the estimation of the variance of the estimator.
is.indefinite

Value
A TRUE or FALSE value.

Author(s)
Frederick Novomestky <fnovomes@poly.edu>

References

Examples
```r
A <- diag(1, 3)
is.idempotent.matrix(A)
B <- matrix(c(1, 2, 3, 4), nrow=2, byrow=FALSE)
is.idempotent.matrix(B)
C <- matrix(c(1, 0, 0, 0), nrow=2, byrow=TRUE)
is.idempotent.matrix(C)
```

is.indefinite Test matrix for positive indefiniteness

Description
This function returns TRUE if the argument, a square symmetric real matrix x, is indefinite. That is, the matrix has both positive and negative eigenvalues.

Usage
```r
is.indefinite(x, tol=1e-8)
```

Arguments

<table>
<thead>
<tr>
<th>x</th>
<th>a matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>tol</td>
<td>a numeric tolerance level</td>
</tr>
</tbody>
</table>

Details
For an indefinite matrix, the matrix should have both positive and negative eigenvalues. The R function `eigen` is used to compute the eigenvalues. If any of the eigenvalues is absolute value less than the given tolerance, that eigenvalue is replaced with zero. If the matrix has both positive and negative eigenvalues, it is declared to be indefinite.
is.indefinite

Value

TRUE or FALSE.

Author(s)

Frederick Novomestky <fnovomes@poly.edu>

References


See Also

is.positive.definite, is.positive.semi.definite, is.negative.definite, is.negative.semi.definite

Examples

```r
# identity matrix is always positive definite
I <- diag(1, 3)
is.indefinite(I)

# positive definite matrix
A <- matrix(c(2, -1, 0, -1, 2, -1, 0, -1, 2), nrow=3, byrow=TRUE)
is.indefinite(A)

# positive semi-definite matrix
eigenvalues are 3.4142136 2.0000000 0.585786
B <- matrix(c(2, -1, 2, -1, 2, -1, 2, -1, 2), nrow=3, byrow=TRUE)
is.indefinite(B)

# negative definite matrix
eigenvalues are -3.4142136 -2.0000000 0.5857864
C <- matrix(c(-2, 1, 0, 1, -2, 1, 0, 1, -2), nrow=3, byrow=TRUE)
is.indefinite(C)

# negative semi-definite matrix
eigenvalues are -4.732051 1.267949 8.881784e-16
D <- matrix(c(-2, 1, -2, 1, -2, 1, -2, 1, -2), nrow=3, byrow=TRUE)
is.indefinite(D)

# indefinite matrix
eigenvalues are 3.828427 1.000000 -1.828427
E <- matrix(c(1, 2, 0, 2, 1, 2, 0, 2, 1), nrow=3, byrow=TRUE)
```
is.negative.definite

Description

This function returns TRUE if the argument, a square symmetric real matrix x, is negative definite.

Usage

is.negative.definite(x, tol=1e-8)

Arguments

x  
a matrix

tol  
a numeric tolerance level

Details

For a negative definite matrix, the eigenvalues should be negative. The R function eigen is used to compute the eigenvalues. If any of the eigenvalues in absolute value is less than the given tolerance, that eigenvalue is replaced with zero. If any of the eigenvalues is greater than or equal to zero, then the matrix is not negative definite. Otherwise, the matrix is declared to be negative definite.

Value

TRUE or FALSE.

Author(s)

Frederick Novomestky <fnovomes@poly.edu>

References


See Also

is.positive.definite, is.positive.semi.definite, is.negative.semi.definite, is.indefinite
Examples

### identity matrix is always positive definite
I <- diag(1, 3)
is.negative.definite(I)

### positive definite matrix
### eigenvalues are 3.4142136 2.0000000 0.585786
###
A <- matrix(c(2, -1, 0, -1, 2, -1, 0, -1, 2), nrow=3, byrow=TRUE)
is.negative.definite(A)

### positive semi-definite matrix
### eigenvalues are 4.732051 1.267949 8.881784e-16
###
B <- matrix(c(2, -1, 2, -1, 2, -1, 2, -1, 2), nrow=3, byrow=TRUE)
is.negative.definite(B)

### negative definite matrix
### eigenvalues are -0.5857864 -2.0000000 -3.4142136
###
C <- matrix(c(-2, 1, 0, -2, 1, 0, -2, 1, 0), nrow=3, byrow=TRUE)
is.negative.definite(C)

### negative semi-definite matrix
### eigenvalues are 1.894210e-16 -1.267949 -4.732051
###
D <- matrix(c(-2, 1, -2, 1, -2, 1, -2, 1, -2), nrow=3, byrow=TRUE)
is.negative.definite(D)

### indefinite matrix
### eigenvalues are 3.828427 1.000000 -1.828427
###
E <- matrix(c(1, 2, 0, 2, 1, 2, 0, 2, 1), nrow=3, byrow=TRUE)
is.negative.definite(E)

---

is.negative.semi.definite

Test matrix for negative semi definiteness

Description

This function returns TRUE if the argument, a square symmetric real matrix x, is negative semi-negative.

Usage

is.negative.semi.definite(x, tol=1e-8)
is.negative.semi.definite

Arguments

- `x` a matrix
- `tol` a numeric tolerance level

Details

For a negative semi-definite matrix, the eigenvalues should be non-positive. The R function `eigen` is used to compute the eigenvalues. If any of the eigenvalues in absolute value is less than the given tolerance, that eigenvalue is replaced with zero. Then, if any of the eigenvalues is greater than zero, the matrix is not negative semi-definite. Otherwise, the matrix is declared to be negative semi-definite.

Value

TRUE or FALSE.

Author(s)

Frederick Novomestky <fnovomes@poly.edu>

References


See Also

`is.positive.definite`, `is.positive.semi.definite`, `is.negative.definite`, `is.indefinite`

Examples

```r
### identity matrix is always positive definite
I <- diag(1, 3)
is.negative.semi.definite(I)

### positive definite matrix
eigenvalues are 3.4142136 2.0000000 0.585786
A <- matrix(c(2, -1, 0, -1, 2, -1, 0, -1, 2), nrow=3, byrow=TRUE)
is.negative.semi.definite(A)

### positive semi-definite matrix
eigenvalues are 4.732051 1.267949 0.881784e-16
B <- matrix(c(2, -1, 2, -1, 2, -1, 2, -1, 2), nrow=3, byrow=TRUE)
is.negative.semi.definite(B)

### negative definite matrix
eigenvalues are -0.5857864 -2.0000000 -3.4142136
```
C <- matrix( c( -2, 1, 0, -2, 1, 0, 1, -2 ), nrow=3, byrow=TRUE )
is.negative.semi.definite( C )
###
### negative semi-definite matrix
### eigenvalues are 1.894210e-16 1.267949 4.732051
###
D <- matrix( c( -2, 1, -2, 1, -2, -2, 1, -2 ), nrow=3, byrow=TRUE )
is.negative.semi.definite( D )
###
### indefinite matrix
### eigenvalues are 3.828427 1.000000 1.828427
###
E <- matrix( c( 1, 2, 0, 2, 1, 2, 0, 2 ), nrow=3, byrow=TRUE )
is.negative.semi.definite( E )

is.non.singular.matrix

Test if matrix is non-singular

Description

This function returns TRUE is the matrix argument is non-singular and FALSE otherwise.

Usage

is.non.singular.matrix(x, tol = 1e-08)

Arguments

x a numeric square matrix
tol a numeric tolerance level usually left out

Details

The determinant of the matrix x is first computed. If the absolute value of the determinant is greater than or equal to the given tolerance level, then a TRUE value is returned. Otherwise, a FALSE value is returned.

Value

TRUE or FALSE value.

Author(s)

Frederick Novomestky <fnovomes@poly.edu>
is.positive.definite

References


See Also

* is.singular.matrix

Examples

```r
A <- diag(1, 3)
is.non.singular.matrix(A)
B <- matrix(c(0, 0, 3, 4), nrow=2, byrow=TRUE)
is.non.singular.matrix(B)
```

is.positive.definite Test matrix for positive definiteness

Description

This function returns TRUE if the argument, a square symmetric real matrix x, is positive definite.

Usage

```r
is.positive.definite(x, tol=1e-8)
```

Arguments

- `x`: a matrix
- `tol`: a numeric tolerance level

Details

For a positive definite matrix, the eigenvalues should be positive. The R function `eigen` is used to compute the eigenvalues. If any of the eigenvalues in absolute value is less than the given tolerance, that eigenvalue is replaced with zero. If any of the eigenvalues is less than or equal to zero, then the matrix is not positive definite. Otherwise, the matrix is declared to be positive definite.

Value

TRUE or FALSE.

Author(s)

Frederick Novomestky <fnovomes@poly.edu>
is.positive.semi.definite

References


See Also

is.positive.semi.definite, is.negative.definite, is.negative.semi.definite, is.indefinite

Examples

```r
### identity matrix is always positive definite
I <- diag(1, 3)
is.positive.definite(I)
### positive definite matrix
### eigenvalues are 3.4142136 2.0000000 0.585786
###
A <- matrix(c(2, -1, 0, -1, 2, -1, 0, -1, 2), nrow=3, byrow=TRUE)
is.positive.definite(A)
###
### positive semi-definite matrix
### eigenvalues are 4.732051 1.267949 8.881784e-16
###
B <- matrix(c(2, -1, 2, -1, 2, -1, 2, -1, 2), nrow=3, byrow=TRUE)
is.positive.definite(B)
###
### negative definite matrix
### eigenvalues are -0.5857864 -2.0000000 -3.4142136
###
C <- matrix(c(-2, 1, 0, 1, -2, 1, 0, 1, -2), nrow=3, byrow=TRUE)
is.positive.definite(C)
###
### negative semi-definite matrix
### eigenvalues are 1.894210e-16 -1.267949 -4.732051
###
D <- matrix(c(-2, 1, -2, 1, -2, 1, -2, 1, -2), nrow=3, byrow=TRUE)
is.positive.definite(D)
###
### indefinite matrix
### eigenvalues are 3.828427 1.000000 -1.828427
###
E <- matrix(c(1, 2, 0, 2, 1, 2, 0, 2, 1), nrow=3, byrow=TRUE)
is.positive.definite(E)
```

is.positive.semi.definite

Test matrix for positive semi-definiteness
Description

This function returns TRUE if the argument, a square symmetric real matrix x, is positive semi-definite.

Usage

is.positive.semi.definite(x, tol=1e-8)

Arguments

x a matrix

tol a numeric tolerance level

Details

For a positive semi-definite matrix, the eigenvalues should be non-negative. The R function eigen is used to compute the eigenvalues. If any of the eigenvalues is less than zero, then the matrix is not positive semi-definite. Otherwise, the matrix is declared to be positive semi-definite.

Value

TRUE or FALSE.

Author(s)

Frederick Novomestky <fnovomes@poly.edu>

References


See Also

is.positive.definite, is.negative.definite, is.negative.semi.definite, is.indefinite

Examples

### identity matrix is always positive definite
I <- diag(1, 3)
is.positive.semi.definite( I )
### positive definite matrix
### eigenvalues are 3.4142136 2.0000000 0.585786
###
A <- matrix( c(2, -1, 0, -1, 2, -1, 0, -1, 2 ), nrow=3, byrow=TRUE )
is.positive.semi.definite( A )
###
### positive semi-definite matrix
is.singular.matrix

### eigenvalues are 4.732051 1.267949 8.881784e-16
###
B <- matrix(c( 2, -1, 2, -1, 2, -1, 2, -1, 2, -1, 2, -1, 2, -1, 2 ), nrow=3, byrow=TRUE )
is.positive.semi.definite( B )
###
### negative definite matrix
### eigenvalues are -0.5857864 -2.0000000 -3.4142136
###
C <- matrix(c( -2, 1, 0, 1, -2, 1, 0, 1, -2 ), nrow=3, byrow=TRUE )
is.positive.semi.definite( C )
###
### negative semi-definite matrix
### eigenvalues are 1.894210e-16 -1.267949 -4.732051
###
D <- matrix(c( -2, 1, -2, 1, -2, 1, -2, 1, -2 ), nrow=3, byrow=TRUE )
is.positive.semi.definite( D )
###
### indefinite matrix
### eigenvalues are 3.828427 1.000000 -1.828427
###
E <- matrix(c( 1, 2, 0, 2, 1, 2, 0, 2, 1 ), nrow=3, byrow=TRUE )
is.positive.semi.definite( E )

---

is.singular.matrix  
*Test for singular square matrix*

**Description**

This function returns TRUE is the matrix argument is singular and FALSE otherwise.

**Usage**

```r
is.singular.matrix(x, tol = 1e-08)
```

**Arguments**

- `x`: a numeric square matrix
- `tol`: a numeric tolerance level usually left out

**Details**

The determinant of the matrix x is first computed. If the absolute value of the determinant is less than the given tolerance level, then a TRUE value is returned. Otherwise, a FALSE value is returned.

**Value**

A TRUE or FALSE value.
is.skew.symmetric.matrix

**Author(s)**
Frederick Novomestky <fnovomes@poly.edu>

**References**

**See Also**
is.non.singular.matrix

**Examples**
```r
A <- diag(1, 3)
is.skew.symmetric.matrix(A)
B <- matrix(c(0, 0, 3, 4), nrow=2, byrow=TRUE)
is.skew.symmetric.matrix(B)
```

**Description**
This function returns **TRUE** if the matrix argument `x` is a skew symmetric matrix, i.e., the transpose of the matrix is the negative of the matrix. Otherwise, **FALSE** is returned.

**Usage**
is.skew.symmetric.matrix(x, tol = 1e-08)

**Arguments**
- `x`: a numeric square matrix
- `tol`: a numeric tolerance level usually left out

**Details**
Let `x` be an order `n` matrix. If every element of the matrix `x + x'` in absolute value is less than the given tolerance, then the matrix argument is declared to be skew symmetric.

**Value**
A **TRUE** or **FALSE** value.
is.square.matrix

Author(s)
Frederick Novomestky <fnovomes@poly.edu>

References

Examples
A <- diag( 1, 3 )
is.skew.symmetric.matrix( A )
B <- matrix( c( 0, -2, -1, -2, 0, -4, 1, 4, 0 ), nrow=3, byrow=TRUE )
is.skew.symmetric.matrix( B )
C <- matrix( c( 0, 2, 1, 2, 0, 4, 1, 4, 0 ), nrow=3, byrow=TRUE )
is.skew.symmetric.matrix( C )

Description
The function returns TRUE if the argument is a square matrix and FALSE otherwise.

Usage
is.square.matrix(x)

Arguments
x a matrix

Value
TRUE or FALSE

Author(s)
Frederick Novomestky <fnovomes@poly.edu>

References
is.symmetric.matrix

Examples

A <- matrix( seq(1, 12, 1), nrow=3, byrow=TRUE )
is.symmetric.matrix( A )
B <- matrix( seq(1, 16, 1), nrow=4, byrow=TRUE )
is.symmetric.matrix( B )

Description

This function returns TRUE if the argument is a numeric symmetric square matrix and FALSE otherwise.

Usage

is.symmetric.matrix(x)

Arguments

x

an R object

Value

TRUE or FALSE.

Note

If the argument is not a numeric matrix, the function displays an error message and stops. If the argument is not a square matrix, the function displays an error message and stops.

Author(s)

Frederick Novomestky <fnovomes@poly.edu>

References


See Also

is.square.matrix

Examples

A <- matrix( c(1, 2, 3, 4), nrow=2, byrow=TRUE )
is.symmetric.matrix( A )
B <- matrix( c(1, 2, 2, 1), nrow=2, byrow=TRUE )
is.symmetric.matrix( B )
Description

This function returns a square matrix of order \( p = r \times c \) that, for an \( r \) by \( c \) matrix \( A \), transforms \( \text{vec}(A) \) to \( \text{vec}(A') \) where prime denotes transpose.

Usage

\[
\text{kmatrix}(r, c = r)
\]

Arguments

\[
\begin{align*}
r & \quad \text{a positive integer row dimension} \\
c & \quad \text{a positive integer column dimension}
\end{align*}
\]

Details

The \( r \times c \) matrices \( H_{i,j} \) constructed by the function \text{hmatrices} are combined using direct product to generate the commutation product with the formula

\[
K_{r,c} = \sum_{i=1}^{r} \sum_{j=1}^{c} (H_{i,j} \otimes H'_{i,j})
\]

Value

An order \((r \times c)\) matrix.

Note

If either argument is less than 2, then the function stops and displays an appropriate error message. If either argument is not an integer, then the function stops and displays an appropriate error message.

Author(s)

Frederick Novomestky <fnovomestky@polyn.edu>

References


See Also

\text{hmatrices}
Examples

```r
K <- K.matrix(3, 4)
A <- matrix(seq(1, 12, 1), nrow=3, byrow=TRUE)
vecA <- vec(A)
vecAt <- vec(t(A))
y <- K %*% vecA
print(y)
print(vecAt)
```

### L.matrix

**Construct L Matrix**

**Description**

This function returns a matrix with \( n \times (n + 1) / 2 \) rows and \( N \times n \) columns which for any lower triangular matrix \( A \) transforms \( \text{vec}(A) \) into \( \text{vech}(A) \).

**Usage**

```r
L.matrix(n)
```

**Arguments**

- `n` a positive integer order for the associated matrix \( A \)

**Details**

The formula used to compute the \( L \) matrix which is also called the elimination matrix is:

\[
L = \sum_{j=1}^{n} \sum_{i=j}^{n} u_{i,j} (\text{vec} E_{i,j})'
\]

\( u_{i,j} \) are the \( n \times 1 \) vectors constructed by the function `u.vectors`. \( E_{i,j} \) are the \( n \times n \) matrices constructed by the function `E.matrices`.

**Value**

An \( \left[ \frac{1}{2} n (n + 1) \right] \times n^2 \) matrix.

**Note**

If the argument is not an integer, the function displays an error message and stops. If the argument is less than two, the function displays an error message and stops.

**Author(s)**

Frederick Novomestky <fnovomes@poly.edu>
**lower.triangle**

**References**


**See Also**

- elimination.matrix
- E.matrices
- u.vectors

**Examples**

```r
L <- L.matrix(4)
A <- lower.triangle( matrix( seq(1,16), nrow=4, byrow=TRUE ) )
vecA <- vec( A )
vechA <- vech( A )
y <- L %*% vecA
print( y )
print( vechA )
```

---

**lower.triangle**

*Lower triangle portion of a matrix*

**Description**

Returns the lower triangle including the diagonal of a square numeric matrix.

**Usage**

```r
lower.triangle(x)
```

**Arguments**

- `x` a matrix

**Value**

A matrix.

**Author(s)**

Frederick Novomestky <fnovomes@poly.edu>

**References**

lu.decomposition

See Also

is.square.matrix

Examples

```r
B <- matrix( seq(1, 16, 1), nrow=4, byrow=TRUE )
lower.triangle( B )
```

Description

This function performs an LU decomposition of the given square matrix argument the results are returned in a list of named components. The Doolittle decomposition method is used to obtain the lower and upper triangular matrices.

Usage

```r
lu.decomposition(x)
```

Arguments

- `x` a numeric square matrix

Details

The Doolittle decomposition without row exchanges is performed generating the lower and upper triangular matrices separately rather than in one matrix.

Value

A list with two named components.

- `L` The numeric lower triangular matrix
- `U` The number upper triangular matrix

Author(s)

Frederick Novomestky <fnovomes@poly.edu>

References


matrix.inverse

**Examples**

```r
A <- matrix(c(1, 2, 2, 1), nrow=2, byrow=TRUE)
luA <- lu.decomposition(A)
L <- lu$L
U <- lu$U
print(L)
print(U)
print(L %*% U)
print(A)
B <- matrix(c(2, -1, -2, -4, 6, 3, -4, -2, 8), nrow=3, byrow=TRUE)
luB <- lu.decomposition(B)
L <- lu$L
U <- lu$U
print(L)
print(U)
print(L %*% U)
print(B)
```

---

**Description**

This function returns the inverse of a square matrix computed using the R function `solve`.

**Usage**

```r
matrix.inverse(x)
```

**Arguments**

- `x`: a square numeric matrix

**Value**

A matrix.

**Author(s)**

Frederick Novomestky <fnovomes@poly.edu>

**References**

Examples

```r
A <- matrix( c(1, 2, 2, 1), nrow=2, byrow=TRUE)
p<- (A )
invA <- matrix.inverse(A )
p<- ( A %**% invA )
p<- ( invA %**% A )
```

matrix.power  Matrix Raised to a Power

Description

This function computes the k-th power of order n square matrix x If k is zero, the order n identity matrix is returned. argument k must be an integer.

Usage

```
matrix.power(x, k)
```

Arguments

- `x`: a numeric square matrix
- `k`: a numeric exponent

Details

The matrix power is computed by successive matrix multiplications. If the exponent is zero, the order n identity matrix is returned. If the exponent is negative, the inverse of the matrix is raised to the given power.

Value

An order n matrix.

Author(s)

Frederick Novomestky <fnovomes@poly.edu>

References

matrix.rank

Examples

A <- matrix(c(1, 2, 2, 1), nrow=2, byrow=TRUE)
matrix.power(A, -2)
matrix.power(A, -1)
matrix.power(A, 0)
matrix.power(A, 1)
matrix.power(A, 2)

matrix.rank

Rank of a square matrix

Description

This function returns the rank of a square numeric matrix based on the selected method.

Usage

matrix.rank(x, method = c("qr", "chol"))

Arguments

x a matrix
method a character string that specifies the method to be used

Details

If the user specifies "qr" as the method, then the QR decomposition function is used to obtain the rank. If the user specifies "chol" as the method, the rank is obtained from the attributes of the value returned.

Value

An integer.

Note

If the argument is not a square numeric matrix, then the function presents an error message and stops.

Author(s)

Frederick Novomestky <fnovomes@poly.edu>

References

See Also

is.square.matrix

Examples

A <- diag( seq( 1, 4, 1 ) )
matrix.rank( A )
B <- matrix( seq( 1, 16, 1 ), nrow=4, byrow=TRUE )
matrix.rank( B )

matrix.trace( A )

Description

This function returns the trace of a given square numeric matrix.

Usage

matrix.trace(x)

Arguments

x  a matrix

Value

A numeric value which is the sum of the values on the diagonal.

Note

If the argument x is not numeric, the function presents an error message and terminates. If the argument x is not a square matrix, the function presents an error message and terminates.

Author(s)

Frederick Novomestky <fnovomes@poly.edu>

References


Examples

A <- matrix( seq( 1, 16, 1 ), nrow=4, byrow=TRUE )
matrix.trace( A )
maximum.norm  

Maximum norm of matrix

Description

This function returns the max norm of a real matrix.

Usage

maximum.norm(x)

Arguments

x  
a numeric matrix or vector

Details

Let \( x \) be an \( m \times n \) real matrix. The max norm returned is \( \|x\|_{\text{max}} = \max_{i,j} |x_{i,j}| \).

Value

A numeric value.

Author(s)

Frederick Novomestky <fnovomes@poly.edu>

References


See Also

inf.norm, one.norm

Examples

A <- matrix(c(3, 5, 7, 2, 6, 4, 0, 2, 8), nrow=3, ncol=3, byrow=TRUE)
m = maximum.norm(A)
**N.matrix**

**Construct N Matrix**

**Description**

This function returns the order n square matrix that is the sum of an implicit commutation matrix and the order n identity matrix quantity divided by two.

**Usage**

\[
\text{N.matrix}(n)
\]

**Arguments**

\[
n \quad \text{A positive integer matrix order}
\]

**Details**

Let \( K_n \) be the order \( n \) implicit commutation matrix (i.e., \( K_{n,n} \), and \( I_n \) the order \( n \) identity matrix. The formula for the matrix is

\[
N = \frac{1}{2} (K_n + I_n).
\]

**Value**

An order \( n \) matrix.

**Note**

If the argument is not an integer, the function displays an error message and stops. If the argument is less than two, the function displays an error message and stops.

**Author(s)**

Frederick Novomestky <fnovomes@poly.edu>

**References**


**See Also**

\[ \text{K.matrix} \]

**Examples**

\[
N <- \text{N.matrix}(3)
\]

\[
\text{print}(N)
\]
one.norm  

Compute the one norm of a matrix

Description
This function returns the $\|x\|_1$ norm of the matrix $x$.

Usage
one.norm(x)

Arguments
x a numeric vector or matrix

Details
Let $x$ be an $m \times n$ matrix. The formula used to compute the norm is $\|x\|_1 = \max_{1 \leq j \leq n} \sum_{i=1}^{m} |x_{i,j}|$. This is merely the maximum absolute column sum of the $m \times n$ matrix.

Value
A numeric value.

Author(s)
Frederick Novomestky <fnovomes@poly.edu>

References


See Also
inf.norm

Examples
A <- matrix( c(3, 5, 7, 2, 6, 4, 0, 2, 8), nrow=3, ncol=3, byrow=TRUE )
one.norm( A )
Description

This function returns an n by n Pascal matrix.

Usage

pascal.matrix(n)

Arguments

n Order of the matrix

Details

In mathematics, particularly matrix theory and combinatorics, the Pascal matrix is a lower triangular matrix with binomial coefficients in the rows. It is easily obtained by performing an LU decomposition on the symmetric Pascal matrix of the same order and returning the lower triangular matrix.

Value

An order n matrix.

Note

If the argument n is not a positive integer, the function presents an error message and stops.

Author(s)

Frederick Novomestky <fnovomes@poly.edu>

References


See Also

lu.decomposition, symmetric.pascal.matrix
Examples

```
P <- pascal.matrix( 4 )
print( P )
```

---

**set.submatrix**

*Store matrix inside another matrix*

**Description**

This function returns a matrix which is a copy of matrix x into which the contents of matrix y have been inserted at the given row and column.

**Usage**

```
set.submatrix(x, y, row, col)
```

**Arguments**

- **x**: a matrix
- **y**: a matrix
- **row**: an integer row number
- **col**: an integer column number

**Value**

A matrix.

**Note**

If the argument x is not a numeric matrix, then the function presents an error message and stops. If the argument y is not a numeric matrix, then the function presents an error message and stops. If the argument row is not a positive integer, then the function presents an error message and stops. If the argument col is not a positive integer, then the function presents an error message and stops. If the target row range does not overlap with the row range of argument x, then the function presents an error message and stops. If the target col range does not overlap with the col range of argument x, then the function presents an error message and stops.

**Author(s)**

Frederick Novomestky <fnovomes@poly.edu>

**Examples**

```
x <- matrix( seq( 1, 16, 1 ), nrow=4, byrow=TRUE )
y <- matrix( seq( 1, 4, 1 ), nrow=2, byrow=TRUE )
z <- set.submatrix( x, y, 3, 3 )
```
Description

This function returns a matrix that has had its rows shifted downwards filling the above rows with the given fill value.

Usage

\[
\text{shift.down}(A, \text{rows} = 1, \text{fill} = 0)
\]

Arguments

- \(A\): a matrix
- \(\text{rows}\): the number of rows to be shifted
- \(\text{fill}\): the fill value which as a default is zero

Value

A matrix.

Note

If the argument \(A\) is not a numeric matrix, then the function presents an error message and stops. If the argument \(\text{rows}\) is not a positive integer, then the function presents an error message and stops.

Author(s)

Frederick Novomestky <fnovomes@poly.edu>

Examples

\[
A \leftarrow \text{matrix}(\text{seq}(1, 16, 1), \text{nrow}=4, \text{byrow}=\text{TRUE})
\]
\[
\text{shift.down}(A, 1)
\]
\[
\text{shift.down}(A, 3)
\]
shift.left

Shift a matrix n columns to the left

Description

This function returns a matrix that has been shifted n columns to the left filling the subsequent columns with the given fill value.

Usage

shift.left(A, cols = 1, fill = 0)

Arguments

A  a matrix
cols  integer number of columns to be shifted to the left
fill  the fill value which has a default zero

Value

A matrix.

Note

If the argument A is not a numeric matrix, then the function presents an error message and stops. If the argument cols is not a positive integer, then the function presents an error message and stops.

Author(s)

Frederick Novomestky <fnovomes@poly.edu>

Examples

A <- matrix( seq( 1, 12, 1 ), nrow=3, byrow=TRUE )
shift.left( A, 1 )
shift.left( A, 2 )
**shift.right**   

*Shift matrix n columns to the right*

**Description**

This function returns a matrix that has been shifted to the right n columns filling the previous columns with the given fill value.

**Usage**

\[
\text{shift.right}(A, \text{cols} = 1, \text{fill} = 0)
\]

**Arguments**

- **A**  
  a matrix
- **cols**  
  integer number of columns to be shifted to the right
- **fill**  
  the fill which as default value zero

**Value**

A matrix.

**Note**

If the argument A is not a numeric matrix, then the function presents an error message and stops. If the argument rows is not a positive integer, then the function presents an error message and stops.

**Author(s)**

Frederick Novomestky <fnovomes@poly.edu>

**Examples**

```R
A <- matrix( seq( 1, 16, 1 ), nrow=4, byrow=TRUE )
shift.right( A, 1 )
shift.right( A, 2 )
```
**Description**

This function returns a matrix where the argument has been shifted up the given number of rows filling the bottom rows with the given fill value.

**Usage**

```r
shift.up(A, rows = 1, fill = 0)
```

**Arguments**

- `A` a matrix
- `rows` integer number of rows
- `fill` fill value which is the default value of zero

**Value**

A matrix.

**Note**

If the argument A is not a numeric matrix, then the function presents an error message and stops. If the argument rows is not a positive integer, then the function presents an error message and stops.

**Author(s)**

Frederick Novomestky <fnovomes@poly.edu>

**Examples**

```r
A <- matrix( seq( 1, 16, 1 ), nrow=4, byrow=TRUE )
shift.up( A, 1 )
shift.up( A, 3 )
```
spectral.norm  

**Spectral norm of matrix**

**Description**

This function returns the spectral norm of a real matrix.

**Usage**

```r
spectral.norm(x)
```

**Arguments**

- `x`: a numeric matrix or vector

**Details**

Let \( x \) be an \( m \times n \) real matrix. The function computes the order \( n \) square matrix \( A = x' x \). The R function `eigen` is applied to this matrix to obtain the vector of eigenvalues \( \lambda = [\lambda_1 \lambda_2 \cdots \lambda_n] \). By construction the eigenvalues are in descending order of value so that the largest eigenvalue is \( \lambda_1 \). Then the spectral norm is \( \|x\|_2 = \sqrt{\lambda_1} \). If \( x \) is a vector, then \( L_2 = \sqrt{A} \) is returned.

**Value**

A numeric value.

**Note**

If the argument \( x \) is not numeric, an error message is displayed and the function terminates. If the argument is neither a matrix nor a vector, an error message is displayed and the function terminates. If the product matrix \( x' x \) is negative definite, an error message displayed and the function terminates.

**Author(s)**

Frederick Novomestky <fnovomes@poly.edu>

**References**


**stirling.matrix**

**Examples**

```r
x <- matrix( c( 2, 4, 2, 1, 3, 1, 5, 2, 1, 2, 3, 3 ), nrow=4, ncol=4, byrow=TRUE )
spectral.norm( x )
```

---

**stirling.matrix**  
*Stirling Matrix*

**Description**

This function constructs and returns a Stirling matrix which is a lower triangular matrix containing the Stirling numbers of the second kind.

**Usage**

```r
stirling.matrix(n)
```

**Arguments**

- `n`  
  A positive integer value

**Details**

The Stirling numbers of the second kind, $S^j_i$, are used in combinatorics to compute the number of ways a set of $i$ objects can be partitioned into $j$ non-empty subsets $j \leq i$. The numbers are also denoted by $\left\{ \begin{array}{c} i \\ j \end{array} \right\}$. Stirling numbers of the second kind can be computed recursively with the equation $S^j_{i+1} = S^j_{i-1} + j S^j_i$, $1 \leq i \leq n-1$, $1 \leq j \leq i$. The initial conditions for the recursion are $S^1_1 = 1$, $0 \leq i \leq n$ and $S^0_j = S^0_0 = 0$, $0 \leq j \leq n$. The resultant numbers are organized in an order $n+1$ matrix:

$$
\begin{bmatrix}
S^0_0 & 0 & 0 & \cdots & 0 \\
0 & S^1_1 & 0 & \cdots & 0 \\
0 & S^1_1 & S^2_2 & \cdots & 0 \\
\cdots & \cdots & \cdots & \cdots & \cdots \\
0 & S^n_1 & S^n_2 & \cdots & S^n_n \\
\end{bmatrix}
$$

**Value**

An order $n+1$ lower triangular matrix.

**Note**

If the argument `n` is not a positive integer, the function presents an error message and stops.

**Author(s)**

Frederick Novomestky <fnovomes@poly.edu>
svd.inverse

References


Examples

```r
S <- stirling.matrix( 10 )
print( S )
```

svd.inverse

*SVD Inverse of a square matrix*

Description

This function returns the inverse of a matrix using singular value decomposition. If the matrix is a square matrix, this should be equivalent to using the `solve` function. If the matrix is not a square matrix, then the result is the Moore-Penrose pseudo inverse.

Usage

`svd.inverse(x)`

Arguments

- `x`: a numeric matrix

Value

A matrix.

Author(s)

Frederick Novomestky <fnovomes@poly.edu>

References


Examples

```r
A <- matrix( c( 1, 2, 2, 1 ), nrow=2, byrow=TRUE )
invA <- svd.inverse( A )
print( A )
print( invA )
print( A %% invA )
B <- matrix( c( -1, 2, 2 ), nrow=1, byrow=TRUE )
invB <- svd.inverse( B )
print( B )
```
Description

This function returns an n by n symmetric Pascal matrix.

Usage

symmetric.pascal.matrix(n)

Arguments

n Order of the matrix

Details

In mathematics, particularly matrix theory and combinatorics, the symmetric Pascal matrix is a square matrix from which you can derive binomial coefficients. The matrix is an order n symmetric matrix with typical element given by \( S_{i,j} = \frac{n!}{r! \cdot (n-r)!} \) where \( n = i + j - 2 \) and \( r = i - 1 \). The binomial coefficients are elegantly recovered from the symmetric Pascal matrix by performing an LU decomposition as \( S = L \cdot U \).

Value

An order n matrix.

Note

If the argument n is not a positive integer, the function presents an error message and stops.

Author(s)

Frederick Novomestky <fnovomes@poly.edu>

References


Examples

S <- symmetric.pascal.matrix( 4 )
print(S)
List of T Matrices

Description

This function constructs a list of lists. The number of components in the high level list is n. Each of
the n components is also a list. Each sub-list has n components each of which is an order n square
matrix.

Usage

T.matrices(n)

Arguments

n
a positive integer value for the order of the matrices

Details

Let \( E_{i,j} \) \( i = 1, \ldots, n \); \( j = 1, \ldots, n \) be a representative order \( n \) matrix created with function
E.matrices. The order \( n \) matrix \( T_{i,j} \) is defined as follows:
\[
T_{i,j} = \begin{cases}
E_{i,j} & i = j \\
E_{i,j} + E_{j,i} & i \neq j
\end{cases}
\]

Value

A list of \( n \) components.

1 A list of \( n \) components

2 A list of \( n \) components

... 

n A list of \( n \) components

Each component \( j \) of sublist \( i \) is a matrix \( T_{i,j} \)

Note

The argument \( n \) must be an integer value greater than or equal to 2.

Author(s)

Frederick Novomestky <fnovomes@poly.edu>

References

Magnus, J. R. and H. Neudecker (1980). The elimination matrix, some lemmas and applications,

See Also

E.matrices

Examples

T <- T.matrices(3)

toeplitz.matrix Toeplitz Matrix

Description

This function constructs an order n Toeplitz matrix from the values in the order 2 * n - 1 vector x.

Usage

toeplitz.matrix(n, x)

Arguments

n a positive integer value for order of matrix greater than 1
x a vector of values used to construct the matrix

Details

The element $t[i,j]$ in the Toeplitz matrix is $x[i-j+n]$.

Value

An order n matrix.

Note

If the argument n is not a positive integer, the function presents an error message and stops. If the length of x is not equal to 2 * n - 1, the function presents an error message and stops.

Author(s)

Frederick Novomestky <fnovomes@poly.edu>

References


Examples

T <- toeplitz.matrix(4, seq(1, 7))
print(T)
**u.vectors**  
*u vectors of an identity matrix*

**Description**

This function constructs an order \( n \times \frac{(n + 1)}{2} \) identity matrix and an order matrix \( u \) that that maps the ordered pair of indices \((i,j)\ i=j, ..., n; j=1, ..., n\) to a column in this identity matrix.

**Usage**

```r
u.vectors(n)
```

**Arguments**

- `n`  
a positive integer value for the order of underlying matrices

**Details**

The function firsts constructs an identity matrix of order \( \frac{1}{2}n(n + 1) \). \( u_{i,j} \) is the column vector in the order \( \frac{1}{2}n(n + 1) \) identity matrix for column \( k = (j - 1)n + i - \frac{1}{2}j(j - 1) \).

**Value**

A list with two named components

- `k`  
order \( n \) square matrix that maps each ordered pair \((i,j)\) to a column in the identity matrix

- `I`  
order \( \frac{1}{2}n(n + 1) \) identity matrix

**Note**

If the argument is not an integer, the function displays an error message and stops. If the argument is less than two, the function displays an error message and stops.

**Author(s)**

Frederick Novomestky <fnovomes@poly.edu>

**References**


**Examples**

```r
u <- u.vectors(3)
```
**upper.triangle**

Upper triangle portion of a matrix

---

**Description**

Returns the lower triangle including the diagonal of a square numeric matrix.

**Usage**

`upper.triangle(x)`

**Arguments**

- `x`: a matrix

**Value**

A matrix.

**Author(s)**

Frederick Novomestky <fnovomes@poly.edu>

**References**


**See Also**

`is.square.matrix`

**Examples**

```r
A <- matrix( seq( 1, 9, 1 ), nrow=3, byrow=TRUE )
upper.triangle( A )
```
Description
This function returns an m by n matrix of the powers of the alpha vector.

Usage
vandermonde.matrix(alpha, n)

Arguments
alpha A numerical vector of values
n The column dimension of the Vandermonde matrix

Details
In linear algebra, a Vandermonde matrix is an \( m \times n \) matrix with terms of a geometric progression of an \( m \times 1 \) parameter vector \( \alpha = [ \alpha_1, \alpha_2, \ldots, \alpha_m ]' \) such that \( V(\alpha) = \begin{bmatrix} 1 & \alpha_1 & \alpha_1^2 & \cdots & \alpha_1^{n-1} \\ 1 & \alpha_2 & \alpha_2^2 & \cdots & \alpha_2^{n-1} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 1 & \alpha_m & \alpha_m^2 & \cdots & \alpha_m^{n-1} \end{bmatrix} \).

Value
A matrix.

Author(s)
Frederick Novomestky <fnovomes@poly.edu>

References

Examples
alpha <- c( .1, .2, .3, .4 )
V <- vandermonde.matrix( alpha, 4 )
print( V )
vec

Vectorize a matrix

Description
This function returns a column vector that is a stack of the columns of x, an m by n matrix.

Usage
vec(x)

Arguments
x a matrix

Value
A matrix with m n rows and one column.

Author(s)
Frederick Novomestky <fnovomes@poly.edu>

References

Examples
x <- matrix( seq(1, 16, 1), nrow=4, byrow=TRUE )
print(x)
vecx <- vec(x)
print(vecx)

vech

Vectorize a matrix

Description
This function returns a stack of the lower triangular matrix of a square matrix as a matrix with 1 column and n * (n + 1) / 2 rows

Usage
vech(x)
Arguments

x a matrix

Value

A matrix with \( \frac{1}{2}n(n + 1) \) rows and one column.

Author(s)

Frederick Novomestky <fnovomes@poly.edu>

References


See Also

is.square.matrix

Examples

```r
x <- matrix( seq(1, 16, 1), nrow=4, byrow=TRUE )
print(x)
y <- vech(x)
print(y)
```

---

**Direct sum of two arrays**

Description

This function computes the direct sum of two arrays. The arrays can be numerical vectors or matrices. The result is the block diagonal matrix.

Usage

```
x%+%y
```

Arguments

- x a numeric matrix or vector
- y a numeric matrix or vector

Details

If either x or y is a vector, it is converted to a matrix. The result is a block diagonal matrix

\[
\begin{bmatrix}
  x & 0 \\
  0 & y
\end{bmatrix}
\]
Value
A numeric matrix.

Author(s)
Frederick Novomestky <fnovomes@poly.edu>, Kurt Hornik <Kurt.Hornik@wu-wien.ac.at>

References

Examples
```r
x <- matrix( seq( 1, 4 ) )
y <- matrix( seq( 5, 8 ) )
print( x %s% y )
```
Index

*Topic math

%S%, 64

commutation.matrix, 3
creation.matrix, 4
D.matrix, 5
direct.prod, 6
direct.sum, 7
duplication.matrix, 8
E.matrices, 9
elimination.matrix, 10
entrywise.norm, 11
frobenius.matrix, 12
frobenius.norm, 14
frobenius.prod, 15
H.matrices, 16
hadamard.prod, 17
hankel.matrix, 18
hilbert.matrix, 19
hilbert.schmidt.norm, 20
inf.norm, 21
is.diagonal.matrix, 22
is.idempotent.matrix, 23
is.indefinite, 24
is.negative.definite, 26
is.negative.semi.definite, 27
is.non.singular.matrix, 29
is.positive.definite, 30
is.positive.semi.definite, 31
is.singular.matrix, 33
is.skew.symmetric.matrix, 34
is.square.matrix, 35
is.symmetric.matrix, 36
K.matrix, 37
L.matrix, 38
lower.triangle, 39
lu.decomposition, 40
matrix.inverse, 41
matrix.power, 42
matrix.rank, 43
matrix.trace, 44
maximum.norm, 45
N.matrix, 46
one.norm, 47
pascal.matrix, 48
set.submatrix, 49
shift.down, 50
shift.left, 51
shift.right, 52
shift.up, 53
spectral.norm, 54
stirling.matrix, 55
svd.inverse, 56
symmetric.pascal.matrix, 57
T.matrices, 58
toeplitz.matrix, 59
u.vectors, 60
upper.triangle, 61
vandermonde.matrix, 62
vec, 63
vech, 63

%S%, 64

commutation.matrix, 3
creation.matrix, 4
D.matrix, 5, 8
direct.prod, 6
direct.sum, 7
duplication.matrix, 8
E.matrices, 9, 11, 39, 59
elimination.matrix, 10, 39
entrywise.norm, 11, 15, 21
frobenius.matrix, 12
frobenius.norm, 14
frobenius.prod, 15
INDEX

H. matrices, 3, 16, 37
hadamard.prod, 16, 17
hankel.matrix, 18
hilbert.matrix, 19
hilbert.schmidt.norm, 20
toeplitz.matrix, 59

inf.norm, 12, 21, 45, 47
is.diagonal.matrix, 22
is.idempotent.matrix, 23
is.indefinite, 24, 26, 28, 31, 32
is.negative.definite, 25, 26, 28, 31, 32
is.negative.semi.definite, 25, 26, 27, 31, 32
is.non.singular.matrix, 29, 34
is.positive.definite, 25, 26, 28, 30, 32
is.positive.semi.definite, 25, 26, 28, 31, 32
is.singular.matrix, 30, 33
is.skew.symmetric.matrix, 34
is.square.matrix, 35, 36, 40, 44, 61, 64
is.symmetric.matrix, 36

K. matrix, 3, 37, 46

L. matrix, 11, 38
lower.triangle, 39
lu.decomposition, 40, 48

matrix.inverse, 41
matrix.power, 42
matrix.rank, 43
matrix.trace, 44
maximum.norm, 45

N. matrix, 46
one.norm, 12, 22, 45, 47

pascal.matrix, 48

set.submatrix, 49
shift.down, 50
shift.left, 51
shift.right, 52
shift.up, 53
spectral.norm, 54
stirling.matrix, 55
svd.inverse, 56
symmetric.pascal.matrix, 48, 57

T. matrices, 6, 58

u.vectors, 6, 11, 39, 60
upper.triangle, 61
vandermonde.matrix, 62
vec, 8, 63
vech, 8, 63