Package ‘matlab’

June 24, 2014

Version 1.0.2
Date 2014-06-19
Title MATLAB emulation package
Author P. Roebuck
Maintainer P. Roebuck <proebuck@mdanderson.org>
Description Emulate MATLAB code using R
Depends R (>= 2.15)
Imports methods
URL http://cran.r-project.org/package=matlab
License Artistic-2.0
Copyright file COPYRIGHTS
LazyLoad true

R topics documented:

matlab-package ....................................................... 2
ceil ................................................................. 3
cell ................................................................. 4
colorbar ............................................................ 4
eye ................................................................. 6
factors .............................................................. 7
fileparts ............................................................ 8
filesep .............................................................. 9
find ................................................................. 9
fix ................................................................. 10
fliplr ............................................................. 11
fullfile .......................................................... 12
hilb .............................................................. 13
MATLAB Emulation Functions

Description

Wrapper functions and variables used to replicate MATLAB function calls as best possible to simplify porting.

Details

<table>
<thead>
<tr>
<th>Package</th>
<th>matlab</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Package</td>
</tr>
<tr>
<td>Version</td>
<td>1.0.2</td>
</tr>
<tr>
<td>Date</td>
<td>2014-06-19</td>
</tr>
<tr>
<td>License</td>
<td>Artistic-2.0</td>
</tr>
</tbody>
</table>
They are no more complete than absolutely necessary and are quite possibly broken for fringe cases. For a complete list of functions, use `library(help="matlab")`. For a high-level summary of the changes for each revision, use `file.show(system.file("NEWS", package="matlab"))`.

**Note**

In certain cases, these may not correspond exactly with MATLAB API as sometimes it just wasn’t possible.

**Author(s)**

P. Roebuck <proebuck@mdanderson.org>

---

**Description**

Rounds to the nearest integer.

**Usage**

`ceil(x)`

**Arguments**

- `x` numeric to be rounded

**Details**

Simply invokes `ceiling` for those more used to C library API name.

**Value**

Returns numeric vector containing smallest integers not less than the corresponding elements of argument `x`.

**Author(s)**

P. Roebuck <proebuck@mdanderson.org>

**See Also**

`fix, Round`

**Examples**

`ceil(c(0.9, 1.3, 2.4))`
cell

MATLAB cell function

Description
Create cell array.

Usage
```
cell(...)
```

Arguments
```
... numeric dimensions for the result
```

Value
Returns list consisting of empty matrices. Defaults to square if dimension argument resolves to a single value.

Author(s)
P. Roebuck <proebuck@mdanderson.org>

See Also
```
ones, zeros
```

Examples
```
cell(3)
cell(c(3, 3)) # same thing
cell(3, 3) # same thing
cell(size(matrix(NA, 3, 3))) # same thing
```

colorbar

MATLAB colorbar function

Description
Displays colorbar showing the color scale.

Usage
```
colorbar(C, location=c("EastOutside", "WestOutside", "NorthOutside", "SouthOutside"), ...)
```
Arguments

C numeric vector or matrix representing data values
location character scalar indicating desired orientation with respect to the axes
...

graphical parameters for image may also be passed as arguments to this method

Details

The values of the elements of C are indices into the current palette that determine the color of each patch.

This implementation differs a bit from its MATLAB counterpart in that the values must be passed explicitly.

Author(s)

P. Roebuck <proebuck@mdanderson.org>

See Also

imagesc, jet.colors, layout.par

Examples

doPlot <- function(C,
  cb.loc=c("EastOutside",
    "WestOutside",
    "NorthOutside",
    "SouthOutside"),
  ...
) {
  saved.par <- par(no.readonly=TRUE)
  on.exit(par(saved.par))
  layout.EO <- function() {
    ## divide the device into one row and nine columns
    ## allocate figure 1 the first eight columns
    ## allocate figure 2 the last column
    layout(matrix(c(1, 1, 1, 1, 1, 1, 1, 1, 2), ncol=9))
  }
  layout.WO <- function() {
    ## divide the device into one row and nine columns
    ## allocate figure 1 the last eight columns
    ## allocate figure 2 the first column
    layout(matrix(c(2, 1, 1, 1, 1, 1, 1, 1, 1), ncol=9))
  }
  layout.NO <- function() {
    ## divide the device into six rows and one column
    ## allocate figure 1 the last five rows
    ## allocate figure 2 the first row
    layout(matrix(c(2, 1, 1, 1, 1), nrow=6))
eye

MATLAB eye function

Description

Create an identity matrix.

Usage

eye(m, n)

Arguments

m, n numeric scalar specifying dimensions for the result

Value

Returns matrix of order 1. Defaults to square if second dimension argument n not provided.

Author(s)

P. Roebuck <proebuck@mdanderson.org>
See Also
ones, zeros

Examples

```matlab
eye(3)
```

---

### Description

Performs prime factorization.

### Usage

```matlab
factors(n)
```

### Arguments

- `n`: numeric scalar specifying composite number to be factored

### Details

Computes the prime factors of `n` in ascending order, each one as often as its multiplicity requires, such that `n == prod(factors(n))`.

### Value

Returns vector containing the prime factors of `n`.

### Note

The corresponding MATLAB function is called 'factor', but was renamed here to avoid conflict with R's compound object class.

### Author(s)

H. Borchers <hwborchers@googlemail.com>, P. Roebuck <proebuck@mdanderson.org>

### See Also

isprime, primes

### Examples

```matlab
factors(1002001)  # 7 7 11 11 13 13
factors(65537)    # is prime
## Eulers calculation
factors(2^32 + 1) # 641 67/zero.noslash/zero.noslash417
```
Description

Return filename parts.

Usage

fileparts(pathname)

Arguments

pathname character string representing pathname to be parsed

Details

Determines the path, filename, extension, and version for the specified file. The returned ext contains a dot (.) before the file extension. The returned versn is always an empty string as the field is provided for compatibility with its namesake’s results.

Value

Returns a list with components:

pathstr character string representing directory path
name character string representing base of file name
ext character string representing file extension
versn character string representing version. Unused

Note

Returns same insane results as does its namesake when handling relative directories, UNIX hidden files, and tilde expansion. Hidden files are returned with name containing a zero length vector and ext containing the actual name. For best results, use this routine to process files, not directories.

Author(s)

P. Roebuck <proebuck@mdanderson.org>

See Also

fullfile

Examples

## Rename dot-txt file as dot-csv
ans <- fileparts("/home/luser/foo.txt")
fullfile(ans$pathstr, paste(ans$name, "csv", sep=".")) # /home/luser/foo.csv
filesep

MATLAB filesep function

Description

Returns the character that separates directory names in filenames.

Usage

filesep

Details

Variable that contains the value of .Platform$file.sep.

Value

Returns character representing this platform's file separator.

Note

Implemented as an R variable rather than a function such that it more closely resembles normal MATLAB usage.

Author(s)

P. Roebuck <proebuck@mdanderson.org>

See Also

fileparts, fullfile, pathsep

find

MATLAB find function

Description

Finds indices of elements.

Usage

find(x)

Arguments

x expression to evaluate
Details
If expression is not logical, finds indices of nonzero elements of argument \( x \).

Value
Returns indices of corresponding elements matching the expression \( x \).

Author(s)
P. Roebuck <proebuck@mdanderson.org>

Examples
\[
\begin{align*}
\text{find}(-3:3 & \geq 0) \\
\text{find}(c(0, 1, 0, 2, 3))
\end{align*}
\]

Description
Rounds toward zero.

Usage
\[
\text{fix}(A)
\]

Arguments
- \( A \) numeric to be rounded

Details
Simply invokes \texttt{trunc}.

Value
Returns vector containing integers by truncating the corresponding values of argument \( A \) toward zero.

Author(s)
P. Roebuck <proebuck@mdanderson.org>

See Also
\texttt{ceil}, \texttt{Round}
Examples

fix(c(1.3, 2.5, 3.7))

\[ \text{fliplr} \quad \text{MATLAB matrix flip functions} \]

Description

Flips matrices either left-right or up-down.

Usage

fliplr(object)
flipud(object)

Arguments

object vector or matrix to be flipped

Details

These are S4 generic functions.

Value

Return value is the same type as argument object.

Author(s)

P. Roebuck <proebuck@mdanderson.org>

See Also

rot90

Examples

fliplr(1:9)
flipud(1:9) # same as previous since vectors have no orientation in R
fliplr(matrix(1:9, 3, 3, byrow=TRUE))
flipud(matrix(1:9, 3, 3, byrow=TRUE))
**fullfile**

*MATLAB fullfile function*

---

**Description**

Contracts path to a file from components in platform-independent manner

**Usage**

`fullfile(...)`

**Arguments**

... character strings representing path components

**Details**

Builds a full filename from the directories and filename specified. This is conceptually equivalent to

```
paste(dir1, dir2, dir3, filename, sep=filesep)
```

with care taken to handle cases when directories begin or end with a separator.

**Value**

Returns character vector of arguments concatenated term-by-term and separated by file separator if all arguments have a positive length; otherwise, an empty character vector.

**Author(s)**

P. Roebuck <proebuck@mdanderson.org>

**See Also**

`fileparts`, `filesep`

**Examples**

`fullfile("", "etc", "profile")` # /etc/profile
**Description**

Create a Hilbert matrix.

**Usage**

hilb(n)

**Arguments**

n numeric scalar specifying dimensions for the result

**Details**

The Hilbert matrix is a notable example of a poorly conditioned matrix. Its elements are

\[ H[i, j] = \frac{1}{i + j - 1} \]

**Value**

Returns an \( n \times n \) matrix constructed as described above.

**Author(s)**

H. Borchers <hwborchers@googlemail.com>, P. Roebuck <proebuck@mdanderson.org>

**Examples**

hilb(3)

---

**Description**

Scales image data to the full range of the current palette and displays the image.

**Usage**

imagesc(x=seq(ncol(C)), y=seq(nrow(C)), C, col=jet.colors(12), ...)

---
isempty

Arguments

x, y   locations of grid lines at which the values in C are measured. These must be finite, non-missing and in (strictly) ascending order. By default, the dimensions of C are used.

C   numeric matrix representing data to be plotted. Note that x can be used instead of C for convenience.

col   vector of colors used to display image data

...   graphical parameters for image may also be passed as arguments to this method

Details

Each element of C corresponds to a rectangular area in the image. The values of the elements of C are indices into the current palette that determine the color of each patch.

The method interprets the matrix data as a table of \( f(x[i], y[j]) \) values, so that the x axis corresponds to column number and the y axis to row number, with row 1 at the top, i.e., the same as the conventional printed layout of a matrix.

Author(s)

P. Roebuck <proebuck@mdanderson.org>

See Also

image, jet.colors, par

Examples

values <- matrix(c(seq(1, 5, by=1),
                  seq(2, 10, by=2),
                  seq(3, 15, by=3)), nrow=3, byrow=TRUE)
imagesc(values, xlab="cols", ylab="rows", col=jet.colors(16))

isempty

MATLAB isempty function

Description

Determine if object is empty.

Usage

isempty(A)

Arguments

A   object to evaluate
Details
   An empty object has at least one dimension of size zero.

Value
   Returns TRUE if x is an empty object; otherwise, FALSE.

Author(s)
   P. Roebuck <proebuck@mdanderson.org>

Examples
   isempty(1:3) # FALSE
   isempty(array(NA, c(2, 0, 2))) # TRUE

isprime MATLAB isprime function

Description
   Array elements that are prime numbers.

Usage
   isprime(x)

Arguments
   x numeric vector or matrix containing nonnegative integer values

Value
   Returns an array (or vector) the same size as x containing logical 1 (true) for the elements of x which are prime, and logical 0 (false) otherwise.

Author(s)
   H. Borchers <hwborchers@googlemail.com>, P. Roebuck <proebuck@mdanderson.org>

See Also
   factors, primes

Examples
   x <- c(2, 3, 0, 6, 10)
   ans <- isprime(x) ## 1, 1, 0, 0, 0
   as.logical(ans)       ## true, true, false, false, false
jet.colors  

MATLAB jet function

Description

Creates a vector of \( n \) colors beginning with dark blue, ranging through shades of blue, cyan, green, yellow and red, and ending with dark red.

Usage

jet.colors(n)

Arguments

\( n \) numeric scalar specifying number of colors to be in the palette

Value

Returns vector of \( n \) color names. This can be used either to create a user-defined color palette for subsequent graphics, a \texttt{col=} specification in graphics functions, or in \texttt{par}.

Author(s)

P. Roebuck <proebuck@mdanderson.org>

See Also

\texttt{palette}, \texttt{par}, \texttt{rgb}

Examples

```r
require(graphics)
x <- 1:16
pie(x, col=jet.colors(length(x)))
```

linspace  

MATLAB linspace function

Description

Generate linearly spaced vectors.

Usage

linspace(a, b, n=100)
logspace

Arguments

- **a**: numeric scalar specifying starting point
- **b**: numeric scalar specifying ending point
- **n**: numeric scalar specifying number of points to be generated

Details

Similar to colon operator but gives direct control over the number of points. Note also that although MATLAB doesn’t specifically document this, the number of points generated is actually `floor(n)`.

Value

Returns vector containing containing *n* points linearly spaced between *a* and *b* inclusive. If *n* < 2, the result will be the ending point *b*.

Author(s)

P. Roebuck <proebuck@mdanderson.org>

See Also

- **logspace**

Examples

```matlab
linspace(1, 10, 4)
```

---

**Description**

Generate logarithmically spaced vectors.

**Usage**

```matlab
logspace(a, b, n=50)
```

**Arguments**

- **a**: numeric scalar specifying exponent for starting point
- **b**: numeric scalar specifying exponent for ending point
- **n**: numeric scalar specifying number of points to be generated

**Details**

Useful for creating frequency vectors, it is a logarithmic equivalent of `linspace`. 
Value

Returns vector containing containing \( n \) points logarithmically spaced between decades \( 10^a \) and \( 10^b \). For \( n < 2 \), \( b \) is returned.

Author(s)

P. Roebuck <proebuck@mdanderson.org>

See Also

linspace

Examples

logspace(1, pi, 36)

---

**magic**

MATLAB magic function

Description

Create a magic square.

Usage

magic(n)

Arguments

\( n \) numeric scalar specifying dimensions for the result

Details

The value of the characteristic sum for a magic square of order \( n \) is \( \text{sum}(1 : n^2)/n \). The order \( n \) must be a scalar greater than or equal to 3; otherwise, the result will be either a nonmagic square, or else the degenerate magic squares 1 and '[]'.

Value

Returns an \( n \)-by-\( n \) matrix constructed from the integers 1 through \( N^2 \) with equal row and column sums.

Note

A magic square, scaled by its magic sum, is doubly stochastic.

Author(s)

P. Roebuck <proebuck@mdanderson.org>
meshgrid

See Also

ones, zeros

Examples

magic(3)

meshgrid MATLAB meshgrid functions

Description

Generate X and Y matrices for three-dimensional plots.

Usage

meshgrid(x, y, z, nargout=2)

Arguments

x, y, z numeric vectors of values
nargout numeric scalar that determines number of dimensions to return

Details

In the first example below, the domain specified by vectors x and y are transformed into two arrays which can be used to evaluate functions of two variables and three-dimensional surface plots. The rows of the output array x are copies of the vector x; columns of the output array y are copies of the vector y.

The second example below is syntactic sugar for specifying meshgrid(x, x).

The third example below produces three-dimensional arrays used to evaluate functions of three variables and three-dimensional volumetric plots.

Value

Returns list containing either two or three matrices depending on the value of nargout.

x, y, z output matrices

Note

Limited to two- or three-dimensional Cartesian space.

Author(s)

P. Roebuck <proebuck@mdanderson.org>
Examples

meshgrid(1:3, 10:14)  # example 1
meshgrid(1:3)        # example 2
meshgrid(5:8, 10:14, 2:3, 3)  # example 3

mod MATLAB mod/rem functions

Description

Provides modulus and remainder after division.

Usage

mod(x, y)
rem(x, y)

Arguments

x, y numeric vectors or objects

Value

Returns vector containing result of the element by element operations.

Author(s)

P. Roebuck <proebuck@mdanderson.org>

Examples

## same results with x, y having the same sign
mod(5, 3)
rem(5, 3)

## same results with x, y having different signs
mod(-5, 3)
rem(-5, 3)
Description

Creates a vector of colors equivalent to MATLAB’s default colors to use for multiline plots.

Usage

multiline.plot.colors()

Details

This is equivalent to the MATLAB command

```
get(gca, ColorOrder)
```

Value

Returns vector of color names. This can be used either to create a user-defined color palette for
subsequent graphics, a col= specification in graphics functions, or in par.

Note

Method should be considered experimental and will most likely be removed and replaced with
similar functionality in the near future.

Author(s)

P. Roebuck <proebuck@mdanderson.org>

See Also

`palette`, `par`, `rgb`

Examples

```
require(graphics)
x <- matrix(1:16, nrow=2, byrow=TRUE)
matplot(x, type="l", col=multiline.plot.colors())
```
ndims MATLAB ndims function

Description

Provides number of array dimensions.

Usage

ndims(A)

Arguments

A object of which to determine the number of dimensions

Details

Simply invokes \texttt{length(size(A))}.

Value

Returns the number of dimensions in the array A.

Note

The number of dimensions is always greater than or equal to 2. Initial implementation returned \texttt{length}.

Author(s)

P. Roebuck <proebuck@mdanderson.org>

See Also

size

Examples

\texttt{ndims(2:9)} # 2
\texttt{ndims(magic(4))} # 2
\texttt{ndims(array(1:8, c(2,2,2)))} # 3
nextpow2  MATLAB nextpow2 function

Description

Smallest power of 2 greater than or equal to its argument.

Usage

nextpow2(x)

Arguments

x  numeric or complex value(s).

Details

Computes the smallest power of two that is greater than or equal to the absolute value of x. (That is, \( p \) that satisfies \( 2^p \geq \text{abs}(x) \)). For negative or complex values, the absolute value will be taken.

Value

Returns numeric result containing integer \( p \) as described above. Nonscalar input returns an element-by-element result (of same size/dimensions as its input).

Author(s)

H. Borchers <hwborchers@googlemail.com>, P. Roebuck <proebuck@mdanderson.org>

See Also

pow2

Examples

nextpow2(10)  #  4
nextpow2(1:10) #  0 1 2 3 3 3 4 4
nextpow2(-2*10) # 10
nextpow2(.Machine$double.eps) # -52
nextpow2(c(0.5, 0.25, 0.125)) # -1 -2 -3
MATLAB numel function

**Description**

Provides number of elements in array \( A \) or subscripted array expression.

**Usage**

\[
\text{numel}(A, \text{varargin})
\]

**Arguments**

- **A**: object of which to determine the number of elements
- **varargin**: unimplemented

**Value**

Returns \( \text{prod(size}(A)) \).

**Author(s)**

P. Roebuck <proebuck@mdanderson.org>

**See Also**

- `prod`, `size`

**Examples**

\[
\text{numel}(2:9) \# 8 \\
\text{numel(magic}(4)) \# 16
\]

MATLAB ones/zeros functions

**Description**

Create a matrix consisting of all ones or zeros.

**Usage**

\[
\text{ones}(\ldots) \\
\text{zeros}(\ldots)
\]
**padarray**

**Arguments**

... numeric dimensions for the result

**Value**

Returns matrix consisting only of ones (or zeros). Defaults to square if dimension argument resolves to a single value.

**Author(s)**

P. Roebuck <proebuck@mdanderson.org>

**See Also**

eye

**Examples**

```r
ones(3)
ones(c(3, 3)) # same thing
ones(3, 3) # same thing
ones(size(matrix(NA, 3, 3))) # same thing
zeros(3)
```

---

**Description**

Pad array.

**Usage**

```r
padarray(A, padsize, padval=0, direction=c("both", "pre", "post"))
```

**Arguments**

- `A`: vector, matrix, or array to be padded
- `padsize`: integer vector specifying both amount of padding and the dimension along which to add it
- `padval`: scalar value specifying pad value, which defaults to 0. Instead, it may specify the method used to determine pad values. Valid values for the method are:
  - "circular": pad with circular repetition of elements within the dimension
  - "replicate": pad by repeating border elements of array
  - "symmetric": pad array with mirror reflections of itself
direction character string specifying direction to apply padding. Valid values are:

"both" pad before first element and after last array element along each dimension
"pre" pad after last array element along each dimension
"post" pad before first array element along each dimension

Details

This is an S4 generic function.

Value

Return value is the same type as argument A with requested padding.

Author(s)

P. Roebuck <proebuck@mdanderson.org>

Examples

```
padarray(1:4, c(/zero.noslash, 2)) # /zero.noslash /zero.noslash [1 2 3 4] /zero.noslash /zero.noslash
padarray(1:4, c(/zero.noslash, 2), -1) # -1 -1 [1 2 3 4] -1 -1
padarray(1:4, c(0, 2), -1, "post") # [1 2 3 4] -1 -1
padarray(1:4, c(0, 3), "symmetric", "pre") # 3 2 1 [1 2 3 4]
padarray(letters[1:5], c(/zero.noslash, 3), "replicate") # a a a [a b c d e] e e e
padarray(letters[1:5], c(0, 3), "circular", "post") # [a b c d e] a b c
```

Description

Generate Pascal matrix.

Usage

```
pascal(n, k=0)
```

Arguments

- `n` numeric scalar specifying order
- `k` numeric scalar specifying desired option. Valid values are 0, 1, or 2

MATLAB pascal function

```
pascal(n, k=0)
```

Mex files that implement the function can be downloaded from

http://www.mathworks.com/matlabcentral/fileexchange/26188-forcell-arrays

```
pascal(n, k=0)
```

Usage

```
pascal(n, k=0)
```

Arguments

- `n` numeric scalar specifying order
- `k` numeric scalar specifying desired option. Valid values are 0, 1, or 2
Details
Specifying \( k = 0 \) returns symmetric positive definite matrix with integer entries taken from Pascal’s triangle.
Specifying \( k = 1 \) returns the lower triangular Cholesky factor (up to the signs of the columns) of the Pascal matrix.
Specifying \( k = 2 \) returns a cube root of the identity matrix.

Value
Returns matrix of order \( n \) according to specified option \( k \).

Author(s)
P. Roebuck <proebuck@mdanderson.org>

Examples

\[
\text{pascal}(4) \\
\text{pascal}(3, 2)
\]

---

**pathsep**  
*MATLAB pathsep function*

Description
Returns the character that separates directory names in a list such as the PATH environment variable.

Usage

\[
\text{pathsep}
\]

Details
Variable that contains the value of \$.Platform$path.sep.

Value
Returns character representing this platform’s path separator.

Note
Implemented as an R variable rather than a function such that it more closely resembles normal MATLAB usage.

Author(s)
P. Roebuck <proebuck@mdanderson.org>
See Also

`filesep`

---

**pow2**

**MATLAB pow2 function**

**Description**

Power with base 2.

**Usage**

`pow2(f, e)`

**Arguments**

- `f` numeric vector of factors
- `e` numeric vector of exponents for base 2

**Details**

Computes the expression \( f \times 2^e \) for corresponding elements of `f` and `e`. If `e` is missing, it sets `e` to `f` and `f` to 1. Imaginary parts of complex values are ignored unless `e` is missing.

**Value**

Returns numeric vector constructed as described above.

**Author(s)**

H. Borchers <hwborchers@googlemail.com>, P. Roebuck <proebuck@mdanderson.org>

**See Also**

`nextpow2`

**Examples**

```
pow2(c(0, 1, 2, 3)) # 1 2 4 8
pow2(c(0, -1, 2, 3), c(0,1,-2,3)) # 0.0 -2.0 0.5 24.0
pow2(1i) # 0.7692389+0.6389613i

# For IEEE arithmetic...
pow2(1/2, 1) # 1
pow2(pi/4, 2) # pi
pow2(-3/4, 2) # -3
pow2(1/2, -51) # .Machine$double.eps
pow2(1/2, -1021) # .Machine$double.xmin
```
primes

MATLAB primes function

Description

Generate a list of prime numbers.

Usage

primes(n)

Arguments

n  scalar numeric specifying largest prime number desired.

Details

Generates the list of prime numbers less than or equal to \( n \) using a variant of the basic "Sieve of Eratosthenes" algorithm. This approach is reasonably fast, but requires a copious amount of memory when \( n \) is large. A prime number is one that has no other factors other than 1 and itself.

Value

Returns numeric vector containing prime numbers less than or equal to argument \( n \).

Author(s)

H. Borchers <hwborchers@googlemail.com>, P. Roebuck <proebuck@mdanderson.org>

See Also

isprime, factors

Examples

primes(1000)
length(primes(1e6))  # 78498 prime numbers less than one million
## Not run:
length(primes(1e7))  # 664579 prime numbers less than ten million
length(primes(1e8))  # 5761455 prime numbers less than one hundred million
## End(Not run)
**MATLAB repmat function**

**Description**

Replicate and tile a matrix.

**Usage**

`repmat(A, ...)`

**Arguments**

- `A`: vector or matrix to be tiled. Must be numeric, logical, complex or character.
- `...`: numeric dimensions for the result

**Value**

Returns matrix with value `A` tiled to the number of dimensions specified. Defaults to square if dimension argument resolves to a single value.

**Author(s)**

P. Roebuck `<proebuck@mdanderson.org>`

**See Also**

`ones`, `zeros`

**Examples**

```matlab
repmat(1, 3)       # same as ones(3)
repmat(1, c(3, 3)) # same thing
repmat(1, 3, 3)    # same thing
repmat(1, size(matrix(NA, 3, 3))) # same thing
repmat(matrix(1:4, 2, 2), 3)
```
**reshape**

**MATLAB reshape function**

**Description**

Reshape matrix or array.

**Usage**

`reshape(A, ...)`

**Arguments**

- `A` matrix or array containing the original data
- `...` numeric dimensions for the result

**Details**

In the first example below, an \(m\)-by-\(n\) matrix is created whose elements are taken column-wise from \(A\). An error occurs if \(A\) does not have \(m \times n\) elements.

In the second example below, an \(n\)-dimensional array with the same elements as \(A\) but reshaped to have the size \(m\)-by-\(n\)-by-\(p\). The product of the specified dimensions must be the same as \(\text{prod(size}(A))\).

In the third example below, an \(n\)-dimensional array with the same elements as \(A\) but reshaped to \(\text{siz}\), a vector representing the dimensions of the reshaped array. The quantity \(\text{prod(siz)}\) must be the same as \(\text{prod(size}(A))\).

**Value**

Returns matrix (or array) of requested dimensions containing the elements of \(A\).

**Author(s)**

P. Roebuck <proebuck@mdanderson.org>

**Examples**

```r
Xmat.2d <- matrix(1:12, nrow=4, ncol=3)
reshape(Xmat.2d, 6, 2) # example 1
reshape(Xmat.2d, c(6, 2)) # same thing
Xarr.3d <- reshape(Xmat.2d, c(6, 2, 1)) # example 2
reshape(Xmat.2d, size(Xarr.3d)) # example 3
```
**Description**

Create the Rosser matrix, a classic symmetric eigenvalue test problem.

**Usage**

```matlab
rosser()
```

**Details**

The returned matrix has the following features:

- a double eigenvalue
- three nearly equal eigenvalues
- dominant eigenvalues of opposite sign
- a zero eigenvalue
- a small, nonzero eigenvalue

**Value**

Returns an 8-by-8 matrix with integer elements.

**Author(s)**

P. Roebuck &lt;proebuck@mdanderson.org&gt;

**Examples**

```matlab
rosser()
```

---

**Description**

Rotates matrix counterclockwise k*90 degrees.

**Usage**

```matlab
rot90(A, k=1)
```
Arguments
A matrix to be rotated
k numeric scalar specifying the number of times to rotate (1..4)

Details
Rotating 4 times (360 degrees) returns the original matrix unchanged.

Value
Returns matrix corresponding to argument A having been rotated argument k number of times.

Author(s)
P. Roebuck <proebuck@mdanderson.org>

See Also
fliplr, flipud

Examples
rot90(matrix(1:4, 2, 2))

size MATLAB size function

Description
Provides dimensions of X.

Usage
size(X, dimen)

Arguments
X vector, matrix, or array object
dimen numeric scalar specifies particular dimension

Details
This is an S4 generic function. Vector will be treated as a single row matrix. Stored value is equivalent to dim.
Value
Returns object of class size_t containing the dimensions of input argument X if invoked with a single argument. Returns integer value of specified dimension if invoked with two arguments. If dimen specifies a higher dimension than exists, returns 1 representing the singleton dimension.

Note
Handling of vectors is different than in initial implementation. Initial implementation returned length.

Author(s)
P. Roebuck <proebuck@mdanderson.org>

Examples
size(2:9) # 1 8
size(matrix(1:8, 2, 4)) # 2 4
size(matrix(1:8, 2, 4), 2) # 4
size(matrix(1:8, 2, 4), 3) # 1

size_t-class

Class “size_t”

Description
This class represents the dimensions of another R object

Objects from the Class
Objects can be created by calls of the form new("size_t", ...). Use of generator method is preferred.

Slots
.Data: object of class "integer" containing size values

Extends
Class "integer", from data part. Class "vector", by class "integer". Class "numeric", by class "integer".

Note
Internal class supporting size.

Author(s)
P. Roebuck <proebuck@mdanderson.org>
**std** *MATLAB std function*

**Description**
Computes the standard deviation of the values of x.

**Usage**

    std(x, flag=0)

**Arguments**

- **x** numeric vector or matrix
- **flag** numeric scalar. If 0, selects unbiased algorithm. If 1, selects biased algorithm (currently unsupported).

**Details**
Simply invokes `sd`.

**Value**
Return value depends on argument x. If vector, returns the standard deviation. If matrix, returns vector containing the standard deviation of each column.

**Author(s)**
P. Roebuck <proebuck@mdanderson.org>

**Examples**

    std(1:2) ^ 2

---

**strcmp** *MATLAB strcmp function*

**Description**
Compare strings.

**Usage**

    strcmp(S, T)
Arguments
S, T character vectors to evaluate

Details
Comparisons are case-sensitive and any leading and trailing blanks in either of the strings are explicitly included in the comparison.

Value
Returns TRUE if S is identical to T; otherwise, FALSE.

Note
Value returned is the opposite of the C language convention.

Author(s)
P. Roebuck <proebuck@mdanderson.org>

Examples
```r
strcomp("foo", "bar")  # FALSE
strcomp(c("yes", "no"), c("yes", "no"))  # TRUE
```

Description
Provides sum of elements.

Usage
```r
sum(x, na.rm=FALSE)
```

Arguments
x numeric or logical to be summed
na.rm logical scalar. If TRUE, remove missing values

Details
This is an S4 generic function.

Value
Return value depends on argument x. If vector, returns the same as sum. If matrix, returns vector containing the sum of each column.
**tic**

**Author(s)**

P. Roebuck <proebuck@mdanderson.org>

**Examples**

```r
sum(1:9)
sum(matrix(1:9, 3, 3))
```

---

**tic**

**MATLAB timer functions**

**Description**

Provides stopwatch timer. Function `tic` starts the timer and `toc` updates the elapsed time since the timer was started.

**Usage**

```r
tic(gcFirst=FALSE)
toc(echo=TRUE)
```

**Arguments**

- `gcFirst` logical scalar. If `TRUE`, perform garbage collection prior to starting stopwatch
- `echo` logical scalar. If `TRUE`, print elapsed time to screen

**Details**

Provides analog to `system.time`. Function `toc` can be invoked multiple times in a row.

**Author(s)**

P. Roebuck <proebuck@mdanderson.org>

**Examples**

```r
tic()
for(i in 1:100) mad(runif(1000)) # kill time
toc()
```
vander function

**Description**

Generate Vandermonde matrix from a vector of numbers.

**Usage**

vander(v)

**Arguments**

v numeric or complex vector of values

**Details**

Generates the Vandermonde matrix whose columns are powers of the vector \( v \) (of length \( n \)) using the formula

\[
A[i, j] = v[i]^{(n-j)}
\]

Used when fitting a polynomial to given points.

**Value**

Returns an \( n \)-by-\( n \) matrix constructed as described above.

**Author(s)**

H. Borchers <hwborchers@googlemail.com>, P. Roebuck <proebuck@mdanderson.org>

**Examples**

vander(1:5)
Index

*Topic **arith**
  - ceil, 3
  - factors, 7
  - fix, 10
  - isprime, 15
  - mod, 20
  - nextpow2, 23
  - pow2, 28
  - primes, 29
  - sum, 36

*Topic **array**
  - cell, 4
  - eye, 6
  - fliplr, 11
  - hilb, 13
  - magic, 18
  - meshgrid, 19
  - ndims, 22
  - numel, 24
  - ones, 24
  - padarray, 25
  - pascal, 26
  - repmat, 30
  - reshape, 31
  - rosser, 32
  - rot90, 32
  - size, 33
  - vander, 38

*Topic **attribute**
  - find, 9
  - isempty, 14

*Topic **character**
  - strcmp, 35

*Topic **classes**
  - size_t-class, 34

*Topic **color**
  - jet.colors, 16
    - multiline.plot.colors, 21

*Topic **file**
  - fileparts, 8
  - filesep, 9
  - fullfile, 12
  - pathsep, 27

*Topic **hplot**
  - colorbar, 4
  - imagesc, 13

*Topic **list**
  - cell, 4

*Topic **logic**
  - find, 9
  - isempty, 14

*Topic **manip**
  - linspace, 16
  - logspace, 17

*Topic **package**
  - matlab-package, 2

*Topic **univar**
  - std, 35

*Topic **utilities**
  - tic, 37

  - ceil, 3, 10
  - cell, 4
  - colorbar, 4
  - dim, 33
  - eye, 6, 25

  - factors, 7, 15, 29
  - fileparts, 8, 9, 12
  - filesep, 9, 12, 28
  - find, 9
  - fix, 3, 10
  - fliplr, 11, 33
  - fliplr, ANY-method(fliplr), 11
  - fliplr, array-method(fliplr), 11
  - fliplr, matrix-method(fliplr), 11
  - fliplr, missing-method(fliplr), 11
INDEX

fliplr, vector-method (fliplr), 11  
flipud, 33  
flipud (fliplr), 11  
flipud, ANY-method (fliplr), 11  
flipud, array-method (fliplr), 11  
flipud, matrix-method (fliplr), 11  
flipud, missing-method (fliplr), 11  
flipud, vector-method (fliplr), 11  
fullfile, 8, 9, 12  

hilb, 13  
image, 5, 14  
imagesc, 5, 14  
isempty, 14  
issprime, 7, 15, 29  
jet.colors, 5, 14, 16  
layout, 5  
length, 22, 34  
linspace, 16, 18  
logspace, 17, 17  
magic, 18  
matlab-package, 2  
meshgrid, 19  
mod, 20  
multiline.plot.colors, 21  
ndims, 22  
extpow2, 23, 28  
umel, 24  
ones, 4, 7, 19, 24, 30  
padarray, 25  
padarray, array, numeric, ANY, character-method (padarray), 25  
padarray, array, numeric, character, character-method (padarray), 25  
padarray, array, numeric, missing, missing-method (padarray), 25  
padarray, vector, numeric, ANY, ANY-method (padarray), 25  
palette, 5, 14, 16, 21  
par, 5, 14, 16, 21  
pascal, 26  
pathsep, 9, 27  
pow2, 23, 28  
primes, 7, 15, 29  
prod, 24  
rem (mod), 20  
repmat, 30  
reshape, 31  
rgb, 16, 21  
rosser, 32  
rot90, 11, 32  
Round, 3, 10  
sd, 35  
size, 22, 24, 33, 34  
size, array, integer-method (size), 33  
size, array, missing-method (size), 33  
size, array, numeric-method (size), 33  
size, matrix, integer-method (size), 33  
size, matrix, missing-method (size), 33  
size, matrix, numeric-method (size), 33  
size, missing, ANY-method (size), 33  
sizelvector, missing-method (size), 33  
size, vector, numeric-method (size), 33  
sizetlass, 34  
std, 35  
strcm, 35  
sym, 36, 36  
sym, ANY, ANY-method (sym), 36  
sym, array, logical-method (sym), 36  
sym, array, missing-method (sym), 36  
sym, array, numeric-method (sym), 36  
sym, matrix, logical-method (sym), 36  
sym, matrix, missing-method (sym), 36  
sym, matrix, numeric-method (sym), 36  
sym, missing, ANY-method (sym), 36  
sym, vector, logical-method (sym), 36  
sym, vector, missing-method (sym), 36  
sym, vector, numeric-method (sym), 36  
system.time, 37  
tic (tictoc), 37  
tic, 37  
toc (tictoc), 37  
trunc, 10  
vander, 38  
zeros, 4, 7, 19, 30  
zeros (ones), 24