Creating Matrixes in Matlab

back to Fan's Intro Math for Econ, Matlab Examples, or Dynamic Asset Repositories

Matlab Define Row and Column Vectors (Matrix)

```
% A column vector 4 by 1, with three numbers you fill in by yourself
col_vec = [5;2;3;10]
```

```
col_vec = 4×1
     5
     2
     3
    10
```

```
% Another column vector with 4 random numbers
col_vec_rand = rand(4,1)
```

```
col_vec_rand = 4×1
    0.5472
    0.1386
    0.1493
    0.2575
```

```
% A row vector 1 by 4
row_vec = [3,2,4,5]
```

```
row_vec = 1×4
     3     2     4     5
```

```
% A row vector 1 by 4 with random number
row_vec_rand = rand(1,4)
```

```
row_vec_rand = 1×4
    0.8407    0.2543    0.8143    0.2435
```

Matlab Define a Matrix

```
% A 2 by 3 matrix by hand
mat_a = [1,2,1;
         3,4,10]
```

```
mat_a = 2×3
     1     2     1
     3     4    10
```

```
% Another 2 by 3 matrix, now with random numbers
mat_rand = rand(2,3)
```

```
mat_rand = 2×3
    0.9293    0.1966    0.6160
    0.3500    0.2511    0.4733
```

```
% Another 2 by 3 matrix, now with random integers between 1 and 10
% rand draws between 0 and 1, ceil converts 0.1 to 1, 1.1 to 2, etc
mat_rand = ceil(rand(2,3)*10)
```

```
mat_rand = 2×3
```

```
     4     6    10
     9     6     3
```

Matlab Define a Square Matrix

```
% A 4 by 4 square matrix
mat_square = rand(4)
```

```
mat_square = 4×4
    0.7572    0.0759    0.9340    0.0119
    0.7537    0.0540    0.1299    0.3371
    0.3804    0.5308    0.5688    0.1622
    0.5678    0.7792    0.4694    0.7943
```

```
% or can define 4 by 4
mat_square = rand(4, 4)
```

```
mat_square = 4×4
    0.3112    0.2630    0.4505    0.1524
    0.5285    0.6541    0.0838    0.8258
    0.1656    0.6892    0.2290    0.5383
    0.6020    0.7482    0.9133    0.9961
```

```
% or can define 4 by 4, between 1 and 5 each number
mat_square = ceil(rand(4, 4)*5)
```

```
mat_square = 4×4
     1     1     1     3
     3     4     2     5
     1     5     2     1
     5     5     5     2
```

Identity Matrix

If a matrix *A* is square matrix with the same number of rows and columns, and all diagonal elements are 1 and non-diagonal elements are 0, then *A* is an identity matrix:

- $A_{i,j}$ are the value in the ith row and jth column of the matrix *A*
- *A* is an identity matrix, when: $A_{i,j} = 0 \text{ if } i \neq j$, $A_{i,j} = 1 \text{ if } i = j$

```
% 4 by 4 identity matrix
identity_4by4 = eye(4)
```

```
identity_4by4 = 4×4
     1     0     0     0
     0     1     0     0
     0     0     1     0
     0     0     0     1
```

When a matrix is muplieid by the identity matrix, you get the same matrix back, for example, multiplying random integer 4 by 4 matrix by the 4 by 4 identity matrix:

```
mat_square
```

```
mat_square = 4×4
     1     1     1     3
     3     4     2     5
     1     5     2     1
     5     5     5     2
```

```
mat_square_times_identity = mat_square*identity_4by4
```

```
mat_square_times_identity = 4×4
     1     1     1     3
     3     4     2     5
     1     5     2     1
     5     5     5     2
```

When a row vector is muplieid by the identity matrix, you get the same vector back, for example, multiplying random integer 1 by 4 row vector by the 4 by 4 identity matrix:

```
row_vec
```

```
row_vec = 1×4
     3     2     4     5
```

```
row_vec_times_identity = row_vec*identity_4by4
```

```
row_vec_times_identity = 1×4
     3     2     4     5
```

When an identity matrix is multiplied by a column vector, you get the same vector back, for example, multiplying 4 by 4 identity matrix by random integer 4 by 1 column vector by the :

```
col_vec
```

```
col_vec = 4×1
     5
     2
     3
    10
```

```
col_vec_times_identity = identity_4by4*col_vec
```

```
col_vec_times_identity = 4×1
     5
     2
     3
    10
```

Lower-Triangular Matrix and Upper-Triangular Matrix

A lower triangular matrix is a square matrix where:

- Square matrix*A* is a **lower triangular** matrix, when: $A_{i,j} = 0$ if $i < j$
- Square matrix*A* is a **upper triangular** matrix, when: $A_{i,j} = 0$ if $i > j$

```
% lower triangular matrix of mat_a
lower_triangular = tril(mat_square)
```

```
lower_triangular = 4×4
     1     0     0     0
     3     4     0     0
     1     5     2     0
     5     5     5     2
```

```
% upper triangular matrix of mat_a
upper_triangular = triu(mat_square)
```

```
upper_triangular = 4×4
     1     1     1     3
     0     4     2     5
     0     0     2     1
     0     0     0     2
```

Three Dimensions Matrix (Tensor)

```
% 3 by 3 by 2, storing multiple matrixes together in ten_a
ten_a = zeros(3,3,2);
ten_a(:,:,1) = rand(3,3);
ten_a(:,:,2) = rand(3,3);
disp(ten_a);
```

```
(:,:,1) =

    0.1455    0.5797    0.8530
    0.1361    0.5499    0.6221
    0.8693    0.1450    0.3510


(:,:,2) =

    0.5132    0.2399    0.2400
    0.4018    0.1233    0.4173
    0.0760    0.1839    0.0497
```

```

% Creating four 2 by 3 matrixes
mat_rand = rand(2,3,4)
```

```
mat_rand = 
mat_rand(:,:,1) =

    0.9027    0.4909    0.3377
    0.9448    0.4893    0.9001


mat_rand(:,:,2) =

    0.3692    0.7803    0.2417
    0.1112    0.3897    0.4039


mat_rand(:,:,3) =

    0.0965    0.9421    0.5752
    0.1320    0.9561    0.0598


mat_rand(:,:,4) =

    0.2348    0.8212    0.0430
    0.3532    0.0154    0.1690
```

```
disp(mat_rand);
```

```
(:,:,1) =

    0.9027    0.4909    0.3377
    0.9448    0.4893    0.9001


(:,:,2) =

    0.3692    0.7803    0.2417
    0.1112    0.3897    0.4039


(:,:,3) =

    0.0965    0.9421    0.5752
    0.1320    0.9561    0.0598


(:,:,4) =

    0.2348    0.8212    0.0430
    0.3532    0.0154    0.1690
```