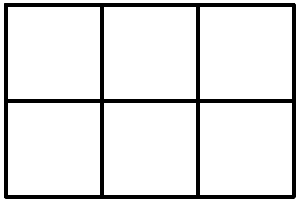
Vectors and Matrices II

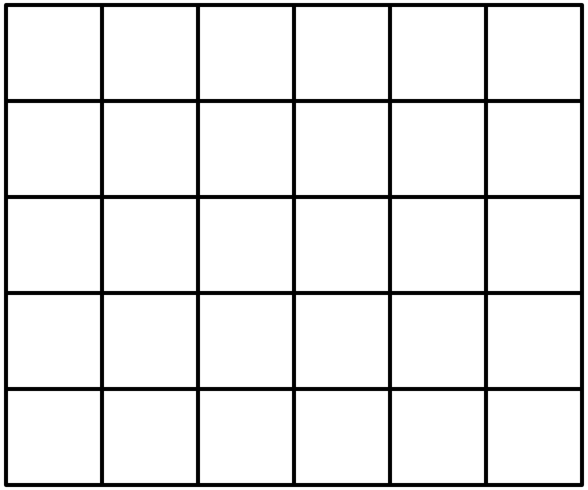
Matrices

- a maxtrix is a 2-dimensional array where the size of the dimensions is usually larger than 1

2 x 3



5 x 6



Creating matrices

- a matrix of size *m* x *n* can be created by entering *m* row vectors of length *n* separated by semi-colons inside square brackets

```
>> I = [[1 0 0];
        [0 1 0];
        [0 0 1]]

I =
     1     0     0
     0     1     0
     0     0     1
```

I is the 3 x 3 identity matrix

Creating matrices

- the square brackets around the individual row vectors are actually unnecessary

```
>> I = [1 0 0;
        0 1 0;
        0 0 1]

I =
     1     0     0
     0     1     0
     0     0     1
```

no square brackets around the row vectors

Creating matrices

- a matrix of size *m* x *n* can be created by entering *n* column vectors of length *m* separated by spaces or commas inside square brackets

4 column vectors of length 2

```
>> P = [[-1; 1], [1; 1], [1; -1], [-1; -1]];

P =
    -1     1     1    -1
     1     1    -1    -1
```

The columns of **P** are the four corners of a square

Indexing elements of a matrix

- the elements of the matrix are usually accessed by using a pair of integer indices
  - textbook calls this *subscripted indexing*
- for a matrix named **`A`**, subscripted indexing has the form:

```
A(row, col)
```

where row is the **`row`** index and **`col`** is the column index of the desired element

```
>>  A = magic(3)
A =
     8     1     6
     3     5     7
     4     9     2

>> A(1, 1)
ans =
     8
>> A(3, 1)
ans =
     4
>> A(2, 3)
ans =
     7
```

$$\begin{bmatrix} A(1,1) & A(1,2) & A(1,3) \\ A(2,1) & A(2,2) & A(2,3) \\ A(3,1) & A(3,2) & A(3,3) \end{bmatrix}$$

Indexing elements of a matrix

- when the colon **:** is used an index it means all rows or columns
  - this is often very useful

```
>>  A = magic(3)
A =
     8     1     6
     3     5     7
     4     9     2
```

```
>> A(:, 2)
ans =
     1
     5
     9
```

$$\begin{bmatrix} A(1,1) & A(1,2) & A(1,3) \\ A(2,1) & A(2,2) & A(2,3) \\ A(3,1) & A(3,2) & A(3,3) \end{bmatrix}$$

```
>> A(3, :)
ans =
     4     9     2
```

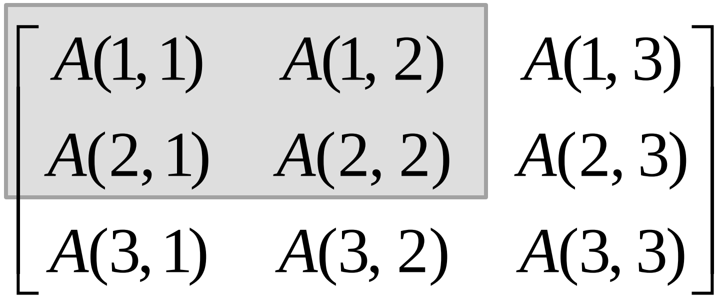
$$\begin{bmatrix} A(1,1) & A(1,2) & A(1,3) \\ A(2,1) & A(2,2) & A(2,3) \\ A(3,1) & A(3,2) & A(3,3) \end{bmatrix}$$

Indexing elements of a matrix

- a submatrix of a matrix can be obtained by using vectors of indices

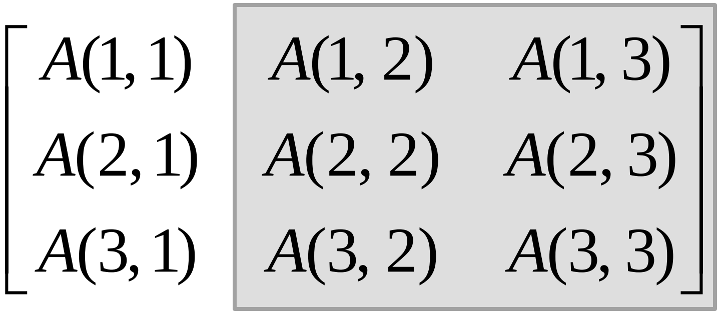
```
>>  A = magic(3)
A =
     8     1     6
     3     5     7
     4     9     2
```

```
>> A(1:2, 1:2)
ans =
     8     1
     3     5
```



$$\begin{bmatrix} A(1,1) & A(1,2) & A(1,3) \\ A(2,1) & A(2,2) & A(2,3) \\ A(3,1) & A(3,2) & A(3,3) \end{bmatrix}$$

```
>> A(:, 2:end)
ans =
     1     6
     5     7
     9     2
```



$$\begin{bmatrix} A(1,1) & A(1,2) & A(1,3) \\ A(2,1) & A(2,2) & A(2,3) \\ A(3,1) & A(3,2) & A(3,3) \end{bmatrix}$$

Indexing elements of a matrix

- you can replace elements of a matrix using indexing

```
>>  A = magic(3)
A =
     8     1     6
     3     5     7
     4     9     2

>> A(3, 3) = 1
ans =
     8     1     6
     3     5     7
     4     9     1
```

$$\begin{bmatrix} A(1,1) & A(1,2) & A(1,3) \\ A(2,1) & A(2,2) & A(2,3) \\ A(3,1) & A(3,2) & \boxed{A(3,3)} \end{bmatrix}$$

```
>>  A = magic(3)
A =
     8     1     6
     3     5     7
     4     9     2

>> A(1, :) = [0 0 0]
ans =
     0     0     0
     3     5     7
     4     9     2
```

$$\begin{bmatrix} A(1,1) & A(1,2) & A(1,3) \\ A(2,1) & A(2,2) & A(2,3) \\ A(3,1) & A(3,2) & A(3,3) \end{bmatrix}$$

```
>>  A = magic(3)
A =
     8     1     6
     3     5     7
     4     9     2

>> A(1:2, 1:2) = [1 0; 0 1]
ans =
     1     0     6
     0     1     7
     4     9     2
```

$$\begin{bmatrix} A(1,1) & A(1,2) & A(1,3) \\ A(2,1) & A(2,2) & A(2,3) \\ A(3,1) & A(3,2) & A(3,3) \end{bmatrix}$$

Creating matrices

- any function that returns a vector can be exploited to create rows of the matrix

```
>> p = [cosd(0:10:360);
        sind(0:10:360)];
>> plot(p(1, :), p(2, :));
>> axis equal
```

p is an array where each column is a point on a circle

- **`axis equal`** scales the plot axes so that 1 unit in x has the same length as 1 unit in y when drawn on the plot
  - i.e., so that a circle will look like a circle instead of an ellipse

Creating matrices

- there are many functions that can be used to create matrices

```
>> help eye
>> help zeros
>> help ones
>> help diag
```

Adding elements to an array

- you can add new elements to an array as long as the dimension of new elements are compatible with the existing array

```
>> v = [1];
>> v = [v 2]
v =
     1     2
```

add to the end of the vector

```
>> v = [1];
>> v = [-1; 0; v]
v =
    -1
     0
     1
```

add to the beginning of the vector

```
>> v = [1 2];
>> v = [v; 3 4]
v =
     1     2
     3     4
```

add a new row to the end of the vector

```
>> v = [1 5];
>> v = [v(1) [2 3 4] v(2)]
v =
     1     2     3     4     5
```

insert in the middle of the vector

```
>> v = [1 5];
>> v = [v(1) [2; 3; 4] v(2)]
Error using horzcat
CAT arguments dimensions are not consistent.
```

```
>> A = zeros(2, 2)
A =
     0     0
     0     0
```

```
>> A = [ones(2, 1) A]
A =
     1     0     0
     1     0     0
```

add a new column to the front of the matrix

```
>> A = [A ones(2, 1)]
A =
     1     0     0     1
     1     0     0     1
```

add a new column to the end of the matrix

example continued on next slide

```
>> v = ones(1, 4);
>> A = [v; A]
A =
     1     1     1     1
     1     0     0     1
     1     0     0     1
```

add a new row to the top of the matrix

```
>> A = [A; v]
A =
     1     1     1     1
     1     0     0     1
     1     0     0     1
     1     1     1     1
```

add a new row to the bottom of the matrix

example continued on next slide

```
>> v = ones(1, 4);
>> A = [A(1:2, :);
        v;
        A(3:4, :)]
A =
     1     1     1     1
     1     0     0     1
     1     1     1     1
     1     0     0     1
     1     1     1     1
```

add a new row to the middle of the matrix

Adding elements to an array

- what is the output of the following MATLAB statements ?

```
>> A = [1];
>> A(2:3, 2:3) = ones(2, 2)
```

Deleting elements from an array

- to delete elements from an array replace the elements with the empty array `[]`
  - the size of the array will decrease
- you can select the elements using indexing

```
>> v = 1:6
v =
     1     2     3     4     5     6
```

```
>> v(1) = []
v =
     2     3     4     5     6
```

delete first element

```
>> v(end) = []
v =
     2     3     4     5
```

delete last element

```
>> v([2 4]) = []
v =
     2     4
```

delete second and last elements

```
>> A = [[1; 2; 3] [4; 5; 6] [7; 8; 9]]
A =
     1     4     7
     2     5     8
     3     6     9
```

```
>> A(3, :) = []
```

delete last row

```
A =
     1     4     7
     2     5     8
```

```
>> A(1, 1) = []
Subscripted assignment dimension mismatch.
```

must delete entire rows or columns