Manipulating MATLAB Matrices
Chapter 4
Objectives

After studying this chapter you should be able to:

• Manipulate matrices
• Extract data from matrices
• Solve problems with two variables
• Explore some of the special matrices built into MATLAB
Section 4.1
Manipulating Matrices

• We’ll start with a brief review
• To define a matrix, type in a list of numbers enclosed in square brackets
Remember that we can define a matrix using the following syntax

- \( A = [3.5] \)
- \( B = [1.5, 3.1] \) or
- \( B = [1.5 \ 3.1] \)
- \( C = [-1, 0, 0; 1, 1, 0; 0, 0, 2] \)
2-D Matrices can also be entered by listing each row on a separate line

\[ C = \begin{bmatrix} -1, 0, 0 \\ 1, 1, 0 \\ 1, -1, 0 \\ 0, 0, 2 \end{bmatrix} \]
Use an ellipsis to continue a definition onto a new line

\[ F = [1, 52, 64, 197, 42, -42, \ldots, 55, 82, 22, 109]; \]
A = 3.5;
Vector – the commas are optional
These semicolons are optional.

2-D matrix

$$C = \begin{bmatrix} -1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 2 \end{bmatrix}$$
You can define a matrix using other matrices as components.

```matlab
>> B = [1.5, 3.1];
>> S = [3.0, B]
S =
    3.0000    1.5000    3.1000
>>
```
Or...

```
>> B = [1.5, 3.1];
>> S = [3.0, B]
S =
    3.0000    1.5000    3.1000
>> T = [1, 2, 3; S]
T =
    1.0000    2.0000    3.0000
   3.0000    1.5000    3.1000
```
Indexing Into an Array allows you to change a value.
Adding Elements

```
>> S
S =
   3.0000   1.5000   3.1000
>> S(2)=1.0
S =
   3.0000   1.0000   3.1000
>> S(4)=5.5
S =
   3.0000   1.0000   3.1000   5.5000
>>
```
If you add an element outside the range of the original array, intermediate elements are added with a value of zero.
4.1.2 Colon Operator

- Used to define new matrices
- Modify existing matrices
- Extract data from existing matrices
Evenly spaced vector

```
>> H=1:8
H =
    Columns 1 through 7
         1     2     3     4     5     6     7
    Column 8
         8
>>
```

The default spacing is 1
User specified spacing

The spacing is specified as 0.5

```
>> time=0.0:0.5:2.0

time =
    Columns 1 through 4
          0   0.5000   1.0000   1.5000
    Column 5
          2.0000
>>
```
The colon can be used to represent an entire row or column. All the rows in column 1:

```
>> M=[1 2 3 4 5
       2 3 4 5 6
       3 4 5 6 7];
```

```
>> x=M(:,1)
x =
    1
    2
    3

All the rows in column 1
```

```
>> y=M(:,4)
y =
    4
    5
    6
```

```
All the rows in column 4
```

```
>> z=M(1,:)
z =
    1   2   3   4   5
```

```
All the columns in row 1
```
You don’t need to extract an entire row or column.

Rows 2 to 3, all the columns
Or…

```
>> M
M =
1   2   3   4   5
2   3   4   5   6
3   4   5   6   7
```

```
>> w=M(2:3, 4:5)
```

```
w =
5   6
6   7
```

Rows 2 to 3, in columns 4 to 5
A single colon transforms the matrix into a column

MATLAB is column dominant
Indexing techniques

- To identify an element in a 2-D matrix use the row and column number.
- For example, element $M(2,3)$.
Matrix indexation

Obtain a single value from a matrix:

Ex:

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \\ 1 & 2 & 4 \end{pmatrix}$$

want to know $a_{21}$

Notation:

$$A(2,1)$$

Row index  Column index

```matlab
>> A=[1 2 3; 3 2 1; 1 2 4];
>> A(2,1)
ans =
    3
>> A(3,2)
ans =
    2
```
Matrix indexation

Obtain more than one value from a matrix:

Ex: X=1:10

Colon defines a “range”: 1 to 10

Notation:

Column 2 to 3

Row 1 to 3

Colon can also be used as a “wildcard”

A(1:3,2:3)

```
>> A=[1 2 3; 3 2 1; 1 2 4];
>> B=A(1:3,2:3)
B =     2     3
      2     1
      2     4

>> C=A(2,:)  
C =     3     2     1
```
Element M(2,3) is in row 2, column 3
Or use single value indexing

M(8) is the same element as M(2,3)
The word “end” signifies the last element in the row or column.
Section 4.2
Problems with Two Variables

- All of our calculations thus far have only included one variable
- Most physical phenomena can vary with many different factors
- We need a strategy for determining the array of answers that results with a range of values for multiple variables
Two scalars give a scalar result

```plaintext
>> x=3;
>> y=5;
>> A=x*y
A =
    15
```
A scalar and a vector give a vector result

```matlab
>> x=1:5
x =
    1    2    3    4    5
>> y=5;
>> A=x*y
A =
    5   10   15   20   25
```
When you multiply two vectors together, they must have the same number of elements.

```
>> x=1:5
x =
    1  2  3  4  5
>> y=1:3
y =
    1  2  3
>> A=x*y
??? Error using ==> mtimes
Inner matrix dimensions must agree.

>> A=x.*y
??? Error using ==> times
Matrix dimensions must agree.
```
Array multiplication gives a result the same size as the input arrays.

x = 1:5
x =
    1   2   3   4   5
y = linspace(1,3,5)
y =
    Columns 1 through 3
    1.0000   1.5000   2.0000
    Columns 4 through 5
    2.5000   3.0000
A = x .* y
A =
    1   3   6  10  15
x and y must be the same size
Results of an element by element (array) multiplication

<table>
<thead>
<tr>
<th>y</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>1.5</td>
<td>1.5</td>
<td>1.5</td>
<td>1.5</td>
<td>1.5</td>
<td>1.5</td>
</tr>
<tr>
<td>2.0</td>
<td>2.0</td>
<td>2.0</td>
<td>2.0</td>
<td>2.0</td>
<td>2.0</td>
</tr>
<tr>
<td>2.5</td>
<td>2.5</td>
<td>2.5</td>
<td>2.5</td>
<td>2.5</td>
<td>2.5</td>
</tr>
<tr>
<td>3.0</td>
<td>3.0</td>
<td>3.0</td>
<td>3.0</td>
<td>3.0</td>
<td>3.0</td>
</tr>
</tbody>
</table>
The meshgrid function maps two vectors onto a 2-D grid.
Now the arrays are the same size, and can be multiplied

```
new_x =
  1  2  3  4  5
  1  2  3  4  5
  1  2  3  4  5

new_y =
  1  1  1  1  1
  2  2  2  2  2
  3  3  3  3  3

A = new_x .* new_y

A =
  1  2  3  4  5
  2  4  6  8 10
  3  6  9 12 15
```
Example 4.2
Distance to the Horizon

- Distance to the horizon, \( d \)
- Height of the mountain
- Radius of the earth, \( R \)
- Radius plus the height of the mountain, \( R+h \)
State the problem

- Find the distance to the horizon from the top of a mountain on the moon and on the earth
Describe the Input and Output

Input
- Radius of the Moon: 1737 km
- Radius of the Earth: 6378 km
- Mountain elevation: 0 to 8000 km

Output
- Distance to the horizon in km
Hand Example

\[ R^2 + d^2 = (R + h)^2 \]  
Pythagorean theorem

\[ d = \sqrt{h^2 + 2Rh} \]  
Solve for d

Using the radius of the earth, and an 8000 meter mountain. (Remember 8000m = 8 km)

\[ d = \sqrt{(8km)^2 + 2 \times 6378km \times 8km} = 319km \]
% Example 4.2
% Find the distance to the horizon
% Define the height of the mountains in meters
clear, clc
format bank
% Define the height vector
h = 0:1000:8000;
% Convert meters to km
h = h/1000;
% Define the radii of the Moon and Earth
radius = [1737 63778];
% Map the radii and heights onto a 2-D grid
[Radius, H] = meshgrid(radius, h);
% Calculate the distance to the horizon
d = sqrt(H.^2 + 2*H.*Radius)
Executing the M-file gives

d =

0       0
58.95   357.15
83.38   505.09
102.13  618.61
117.95  714.31
131.89  798.63
144.50  874.86
156.10  944.96
166.90  1010.20
Test the Solution

- Compare the results to the hand solution
Section 4.3
Special Matrices

- zeros
  - Creates a matrix of all zeros
- ones
  - Creates a matrix of all ones
- diag
  - Extracts a diagonal or creates an identity matrix
- magic
  - Creates a “magic” matrix
**Special matrix creation**

- **zeros(M,N)**
  - Matrix of zeros
  - `>> A=zeros(2,3)`
  - `A =
    0 0 0
    0 0 0`

- **ones(M,N)**
  - Matrix of ones
  - `>> B=ones(2,2)`
  - `B =
    1 1
    1 1`

- **eye(M,N)**
  - Matrix of ones on the diagonal
  - `>> C=eye(2,2)`
  - `C =
    1 0
    0 1`

- **rand(M,N)**
  - Matrix of random numbers between 0 and 1
  - `>> D=rand(3,2)`
  - `D =
    0.9501  0.4860
    0.2311  0.8913
    0.6068  0.7621`
With a single input a square matrix is created with the zeros or ones function.

```
>> A=zeros(3)
A =
    0     0     0
    0     0     0
    0     0     0
>> B=ones(3)
B =
    1.00   1.00   1.00
    1.00   1.00   1.00
    1.00   1.00   1.00
```
Two input arguments specify the number of rows and columns.

```
>> A=zeros(2,3)
A =
  0   0   0
  0   0   0
>> B=ones(3,2)
B =
  1.00  1.00
  1.00  1.00
  1.00  1.00
>>
```
The diag function

When the input argument to the diag function is a square matrix, the diagonal is returned
The diag function

When the input is a vector, it is used as the diagonal of an identity matrix

```matlab
>> B = [1 2 3];
>> diag(B)
ans =
    1.0000    0.0000    0.0000
    0.0000    2.0000    0.0000
    0.0000    0.0000    3.0000
>>
```
Magic Matrices

```plaintext
>> A=magic(4)
A =
   16.00    2.00    3.00    13.00
   5.00   11.00   10.00     8.00
   9.00    7.00    6.00   12.00
   4.00   14.00   15.00     1.00

>> sum(A)
ans =
   34.00    34.00    34.00    34.00

>> sum(A')
ans =
   34.00    34.00    34.00    34.00

>> sum(diag(A))
ans =
   34.00
```

This woodcut called Melancholia was created by Albrecht Durer, in 1514. It contains a magic matrix above the angel’s head.
Albrecht Durer included the date in this magic matrix.
The Durer matrix is different from MATLAB’s 4x4 magic matrix

```
>> durer=[16, 3, 2, 13;
       5, 10, 11, 8;
       9, 6, 7, 12;
       4, 15, 14, 1];

>> sum(durer)
ans =
   34.00    34.00    34.00    34.00

>> sum(durer')
ans =
   34.00    34.00    34.00    34.00

>> sum(diag(durer))
ans =
   34.00

>>
```
**Concatenation:** Combine two (or more) matrices into one

Notation:

\[
C = [A, B]
\]

**Example:**

\[
A = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}
\]

\[
C = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix}, \quad D = \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}
\]
Summary

• Matrices can be created by combining other matrices
• Portions of existing matrices can be extracted to form smaller matrices
Summary – The colon operator

• The colon operator
  • can be used to create evenly spaced matrices
  • can be used to extract portions of existing matrices
  • can be used to transform a 2-D matrix into a single column
Summary - Meshgrid

• Meshgrid is an extremely useful function that can be used to map vectors into two dimensional matrices
  • This makes it possible to perform array calculations with vectors of unequal size
Summary – Special Matrices

• zeros – creates a matrix composed of all zeros
• ones – creates a matrix composed of all ones
• diag – extracts the diagonal from a square matrix or can be used to create a square matrix identity matrix
• magic – creates a “magic matrix”