

# MATLAB for BEGINNERS

*In the High School Classroom*

CHINYEN CHUO

## Table of Contents

|          |   |           |
|----------|---|-----------|
| <b>1</b> | <b>INTRODUCTION</b>   | <b>3</b>  |
| 1.1      | Helpful Commands  | 3         |
| 1.2      | Arithmetic Operators  | 4         |
| 1.3      | Constants   | 5         |
| 1.4      | Build-in-Functions  | 5         |
| 1.5      | Practice  | 5         |
| <b>2</b> | <b>MATRICES AND VECTORS</b>                                   | <b>7</b>  |
| 2.1      | Creating Row and Column Vectors                               | 7         |
| 2.2      | Creating $m \times n$ Matrices                                | 7         |
| 2.3      | Using the Colon Operator and 'linspace' Function              | 8         |
| 2.4      | Referring to Elements   | 8         |
| 2.5      | Modifying Matrix Elements & Creating New Matrices             | 8         |
| 2.6      | Dimensions  | 9         |
| 2.7      | Matrix Calculations   | 9         |
| 2.8      | Summary of Array and Matrix Operations                        | 10        |
| 2.9      | Practice  | 10        |
|          | <b>Appendices</b>   | <b>12</b> |
|          | <b>Appendix A - Exploration #1: Computer Graphics</b>         | <b>12</b> |
|          | <b>Appendix B - Exploration #2: Matrices and Cryptography</b> | <b>17</b> |
|          | <b>References</b>   | <b>20</b> |

# MATLAB

## Tutorial & Practice

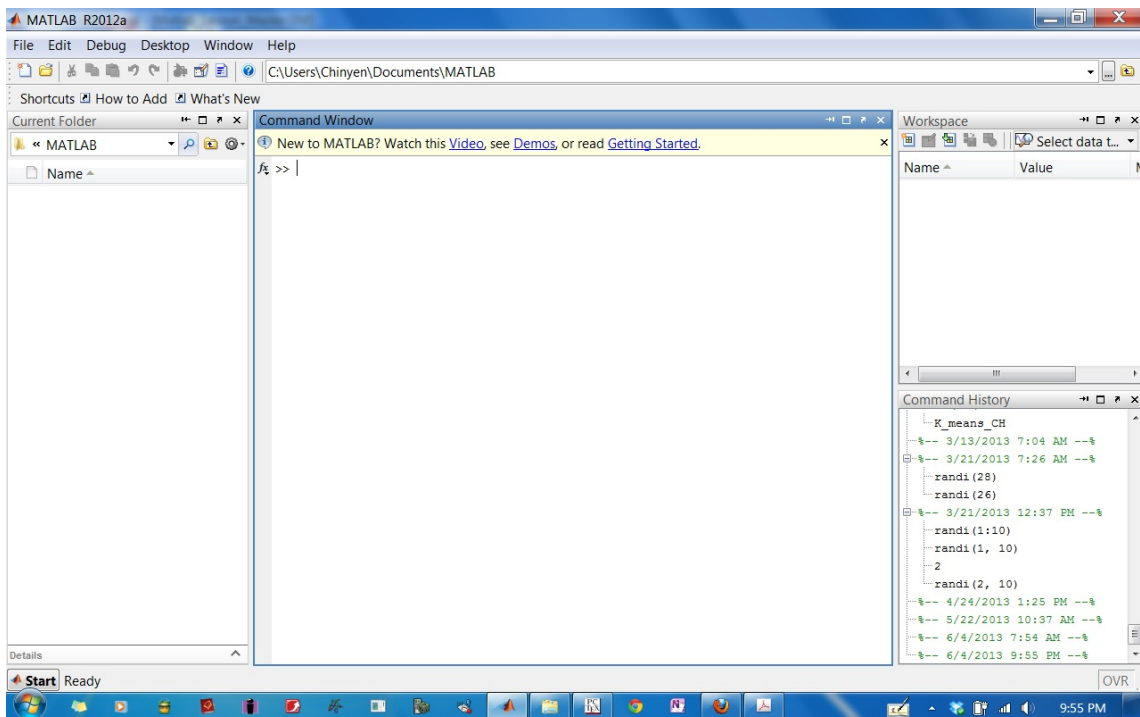
---

## 1 INTRODUCTION

The purpose of this tutorial is to provide a concise material and practice for high school students to gain a basic understanding of MATLAB.

When you open MATLAB, you should see the MATLAB Desktop, and within it, you may see the following windows

- *Command Window* - the main part and you should see the symbol ">>", which is called a *prompt*. Commands are entered here.
- *Current Folder* - where a new file is saved. This also shows the files that are in the current folder.
- *Workspace* - lists all variables in the currently active workspace in alphabetical order.
- *Command History* - shows all commands entered.
- *Details* - shows the details of a file.



### 1.1 Helpful Commands

- These commands are used in the Command Window.

| Command | Desription | Example | Result |
|---------|------------|---------|--------|
|---------|------------|---------|--------|

|                           |   |              |   |
|---------------------------|---|--------------|---|
| demo                      | opens the Help browser and shows some features of MATLAB      |              |   |
| help                      | explains any command  | help exp     | exp(X) is the exponential of the elements of X, e to the X. |
| help help                 | explains how help works                                       |              |   |
| lookfor                   | searches through the help for a specific word or phrase       |              |   |
| ;                         | suppresses output   | >>x = 6*5;   | >>  |
| clear                     | removes all variable from workspace                           |              |   |
| clear <i>variablename</i> | clears a specific variable                                    | clear x      | If 2 was assigned to x, now it no longer exists             |
| clc                       | clears command window   |              |   |
| ctrl+c                    | stops a running MATLAB process                                |              |   |
| x=2                       | assigns 2 to the variable x                                   |              |   |
| who                       | shows variables that have been defined                        |              |   |
| whos                      | shows variables that have been defined in this Command Window |              |   |
| format                    | sets output format  | format short | Scaled fixed point format with 5 digits                     |
| rand                      | generates random numbers from 0 to 1                          | rand         | ans = 0.8147  |

## 1.2 Arithmetic Operators

Use MATLAB to find answers and fill in the blanks.

| MATLAB Operator | Math Operation          | Example | Result |
|-----------------|-------------------------|---------|--------|
| +               | Addition                | 3+5     |        |
| -               | Subtraction             | 3-5     |        |
| *               | Multiplication          | 3*5     |        |
| /               | Division (Divided by)   | 3/5     |        |
| \               | Division (Divided into) | 3\5     |        |
| ^               | Power                   | 3^5     |        |

- MATLAB follows the order of operations:

- ( ) parentheses
- ^ exponentiation
- negation
- \*, /, \ multiplication and division
- +, - addition and subtraction

### 1.3 Constants

Use MATLAB to find answers and fill in the blanks.

| MATLAB Command    | Description  | Example           | Output |
|-------------------|--------------|-------------------|--------|
| <code>pi</code>   | $\pi$        | <code>pi+1</code> |        |
| <code>i</code>    | $\sqrt{-1}$  | <code>i+1</code>  |        |
| <code>j</code>    | $\sqrt{-1}$  | <code>1+j</code>  |        |
| <code>inf</code>  | $\infty$     |                   |        |
| <code>-inf</code> | $-\infty$    |                   |        |
| <code>NaN</code>  | Not a number | <code>0/0</code>  |        |

### 1.4 Build-in-Functions

Use MATLAB to find answers and fill in the blanks.

| MATLAB Command            | Description                                   | Example   | Output |
|---------------------------|---|---|--------|
| <code>help elfun</code>   | shows a list of the elementary math functions | <code>pi+1</code>                                     |        |
| <code>sin</code>          | <b>sine</b> of argument in radians            | <code>sin(pi)</code>                                  |        |
| <code>sind</code>         | <b>sine</b> of an argument in degrees         | <code>sind(180)</code>                                |        |
| <code>asin</code>         | inverse <b>sine</b>                           | <code>asin(1)</code>                                  |        |
| <code>asind</code>        | inverse <b>sine</b> results in degrees        | <code>sind(1)</code>                                  |        |
| <code>exp</code>          | exponential                                   | <code>exp(1)</code>                                   |        |
| <code>log</code>          | natural logarithm                             | <code>log(exp(pi))</code>                             |        |
| <code>log10</code>        | common logarithm (base 10)                    | <code>log10(10)</code>                                |        |
| <code>log2</code>         | logarithm of base 2                           | <code>log2(1/2)</code>                                |        |
| <code>sqrt</code>         | square root                                   | <code>sqrt(4)</code>                                  |        |
| <code>nthroot(x,n)</code> | real n-th root of real numbers                | <code>nthroot(8, 3) = <math>\sqrt[3]{8}</math></code> |        |
| <code>abs</code>          | absolute value                                | <code>abs(-4)</code>                                  |        |
| <code>floor</code>        | round toward negative infinity                | <code>floor(-0.001)</code>                            |        |
| <code>ceil</code>         | round toward positive infinity                | <code>ceil(0.01)</code>                               |        |
| <code>round</code>        | round toward the nearest integer              | <code>round(3.55)</code>                              |        |
| <code>rem</code>          | remainder after division                      | <code>rem(12,4)</code>                                |        |

### 1.5 Practice

- For each of the following, think about what the results would be, and then type them in to MATLAB to verify your

answers:

A)  $2^4-1$

- B)  $2^{(4-1)}$   
 C)  $4/5$   
 D)  $4\sqrt{5}$   
 E)  $2*9+6/3$   
 F)  $7--8$
2. What is the result of `rand(10)?`  
`ran*10`? And `rand*20`?

3. What is the result of the following code?

```
>> low=2;
>> high=6;
>> round(rand*(high-low)+low)
```

4. Write a code that generates an integer in the range from 0 to 60.
5. Write a code to assign 4 to  $x$ , suppress the output; divide 3 by  $\sqrt{2+x}$ ; state the output.
6. For the following code, describe what each line does and state the output:

```
1. >> x=5;
2. >> y=4;
3. >> z=1;
4. >> y-x;
5. >> w=2/x+5*y*z
```

7. Given the code:

```
>> x=5;
>> X=6;
>> x
```

What is the output? Is MATLAB case sensitive?

8. Solve using MATLAB: Assign the values of 2 and 4 to the variables  $x$  and  $y$ , respectively, then compute the value of  $z = 5x + 2y$ .

9. Perform the following and determine the result for the code:

```
A) Clear all variables
B) Clear the Command Window
C) >> %X=5;
    >> x=2;
    >> X
```

10. Use the `help` feature to find the command for computing the value of  $6!$ .

11. Create a script file and write a generic code for computing distance between any two points  $(x_1, y_1)$  and  $(x_2, y_2)$ . Then use the code to find the distances between the points:

A)  $(3, 4)$  and  $(-5, 2)$   
 B)  $(5, 0)$  and  $(-3, -6)$

12. What is `atan(1)`? What does the output represent?

13. What is `tan(pi/2)`? What is `tand(pi/2)`?

14. What is `log2(0)`? Is the output correct? Why or why not?

15. What is the output of `exp(1)`? What is the equivalent mathematical expression?

## 2 MATRICES AND VECTORS

In mathematics, a vector is defined as a quantity that has both magnitude and direction, such as velocity. Matrix is defined as a rectangular array of numbers. The size (order) of a matrix is generally denoted by  $r \times c$ , that is *row* by *column*. Furthermore, a matrix of size  $1 \times n$  can be called a *row vector* and a matrix of  $m \times 1$  a *column vector*. A matrix of one element is called a *scalar* of size  $1 \times 1$ .

Following are examples of matrices:

1.  $[2]$  - a scalar
2.  $[1 \ 4]$  - a row vector
3.  $\begin{bmatrix} 2 \\ 3 \\ 4 \end{bmatrix}$  - a column vector
4.  $\begin{bmatrix} 1 & 3 & 4 \\ 5 & 6 & 7 \end{bmatrix}$  - a  $2 \times 3$  matrix

MATLAB is specially designed as a tool for doing numerical computations with matrices and vectors that are used to store sets of values of the same type. MATLAB stands for *Matrix Laboratory*.

### 2.1 Creating Row and Column Vectors

Note that this is the same as creating a one-dimensional array, or a matrix of size  $1 \times n$  or  $m \times 1$ .

- Three direct ways of creating row vectors:

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

```
v =  
    1    2    3    4
```

2. `>> v=[1,2,3,4]`

```
v =  
    1    2    3    4
```

3. `w=1:4`

```
w =  
    1    2    3    4
```

- A column vector:

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

```
u =  
    1  
    2  
    3  
    4
```

This is a  $4 \times 1$  vector assigned to the variable `u`.

### 2.2 Creating $m \times n$ Matrices

- A  $3 \times 4$  matrix:

```
a=[1 2 3 4; 5 6 7 8; 0 3 2 1]
```

```
a =  
    1    2    3    4  
    5    6    7    8  
    0    3    2    1
```

- You may also enter each row on a separate line to keep track of number of entries in a matrix:

```
>> b=[1 2 3 4;  
    2 4 3 1;  
    8 9 0 1]
```

```
b =  
    1    2    3    4  
    2    4    3    1  
    8    9    0    1
```

Note that `b` is defined as the matrix above. If you type `b` in the Command Window, you should get a matrix output as the one above. You may also get the variable output from the Workspace.

## 2.3 Using the Colon Operator and 'linspace' Function

- Creating a vector with all integers from 1 to 8 in steps of 2:

```
>> v2=1:2:8
```

```
v2 =  
     1     3     5     7
```

- Creating a vector with three values evenly spaced between 1 and 10:

```
>> v3=linspace(1,10,3)
```

```
v3 =  
     1.0000     5.5000    10.0000
```

## 2.4 Referring to Elements

- Referring to an entry in a row or column vector:

```
>> v2=1:2:8
```

```
v2 =  
     1     3     5     7
```

```
>> v2(3)
```

```
ans =  
     5
```

- Referring to an entry in a matrix:

```
>> b=[1 2 3 4;  
     2 4 3 1;  
     8 9 0 1]
```

```
b =  
     1     2     3     4  
     2     4     3     1  
     8     9     0     1
```

```
>> b(3,2)
```

```
ans =  
     9
```

- Referring to a row or column of elements:

```
>> b=[1:3; 3:5]
```

```
b = %matrix b  
     1     2     3  
     3     4     5
```

```
>> b(1,:) %all elements in the first row
```

```
ans =  
     1     2     3
```

```
>> b(:,2) %all elements in the second column  
ans =  
     2  
     4
```

## 2.5 Modifying Matrix Elements & Creating New Matrices

- Concatenating two vectors:

```
>> a=zeros(2)
```

```
a =  
     0     0  
     0     0
```

```
>> b=ones(2,3)
```

```
b =  
     1     1     1  
     1     1     1
```

```
>> newvec=[a,b]  
%combining two vectors as one
```

```
newvec =  
     0     0     1     1     1  
     0     0     1     1     1
```



## 2.6 Dimensions

- Using build-in function 'size':

```
>> A=[1:3; 4:6]

A =
     1     2     3
     4     5     6

>> A=[1:3; 4:6]' %transpose matrix A

A =
     1     4
     2     5
     3     6

>> size(A)

ans =
     3     2 %size: 3X2

>> [r c]=size(A)

r =
     3 %row = 3

c =
     2 %column = 2
```

- Empty vectors:

```
>> B=[]

B =

[]

>> size(B)
ans =
     0     0 %size = 0
```

## 2.7 Matrix Calculations

- Adding / subtracting Matrices: A+B or A-B

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

```
a =
     1     3     5     7
     6     2    -1     4
```

```
>> b=[3: 6; -9 4 0 2]
```

```
b =
     3     4     5     6
    -9     4     0     2
```

```
>> c=a+b
```

```
c =
     4     7    10    13
    -3     6    -1     6
```

- Multiplying two matrices:

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

```
a =
     1     2
     3     4
```

```
>> b=[3: 6; -9 4 0 2]
```

```
b =
     3     4     5     6
    -9     4     0     2
```

```
>> c=a*b
```

```
c =
    -15    12     5    10
    -27    28    15    26
```

Use your calculator, do the math, and verify the answer.

- Multiplying two matrices element by element:

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

```
a =
     1     3     5     7
     6     2    -1     4
```

```
>> b=[3: 6; -9 4 0 2]
```

```
b =
     3     4     5     6
    -9     4     0     2
```

```
>> c=a.*b %(notice the dot after a)
```

```
c =
     3    12    25    42
    -54     8     0     8
```

Note: In order for this operation to work, the size of the matrices must be the same.

- Two different divisions of matrices:

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

```
a =
     1     2
     3     4
```

```
>> b=[3 -2; 5 8]
```

```
b =
     3    -2
     5     8
```

```
>> a/b
```

```
ans =
   -0.0588    0.2353
    0.1176    0.5294
```

```
>> a./b
```

```
ans =
    0.3333   -1.0000
    0.6000    0.5000
```

Note:

- In MATLAB ,  $a/b$  is equivalent to  $a \times b^{-1}$ , where  $b^{-1}$  is the inverse of  $b$
- $a./b$  - This is an *array division* and it divides every element of  $a$  by elements of corresponding position in  $b$

## 2.8 Summary of Array and Matrix Operations

| MATLAB           | Operation / Comments  |
|------------------|---|
| $a/b$            | Matrix right division.<br>This is equivalent to $a \cdot \text{inv}(b)$   |
| $a \backslash b$ | Matrix left division.<br>Matrix division is defined by $\text{inv}(a) \cdot b$ , where $\text{inv}(a)$ is the inverse of matrix $a$ .   |
| $a.^b$           | Array exponentiation.<br>Element-by-element exponentiation of $a$ and $b$ : $a_{i,j}^{b_{i,j}}$ , or $a(i,j)^{b(i,j)}$ . Both arrays must of the same shape, or one of them must be a scalar. |
| $a./b$           | Array right division.<br>Element-by-element division of $a$ and $b$ : $a(i,j)/b(i,j)$ . Both arrays must of the same shape, or one of them must be a scalar.                                  |
| $a.\backslash b$ | Array left division.<br>Element-by-element division of $a$ and $b$ , but $b$ in the numerator $b(i,j)/a(i,j)$ . Both arrays must of the same shape, or one of them must be a scalar.          |

## 2.9 Practice

1. Consider the following sequence of statements and expressions. First conjecture the output of each statement or expression, and then type them in to MATLAB to verify your answers.

- A)  $a=[1:4; 2 \ 10 \ 3 \ 5]$
- B)  $a(1,4)$
- C)  $a(:, 4)$
- D)  $a(4)$
- E)  $\text{size}(a)$
- F)  $\text{numel}(a)$
- G)  $\text{reshape}(a, 1, \text{numel}(a))$
- H)  $\text{vec}=a(1,:)$
- I)  $\text{vec}(2)$
- J)  $\text{vec}(3) = []$

- K) `vec(5) = 8`
- L) `vec = [vec 9]`
- 2. Use the **linspace** function to create a matrix of five evenly spaced values from 5 to 20. (Hint: Use **help** to find the syntax code).
- 3.
  - A) Create a matrix **A** of size  $1 \times 4$  that stores angle measures 10, 15, 70, and 90 in degrees.
  - B) Create an empty matrix called **radians**.
  - C) Convert all the entries in **A** to radians measures and store them in the matrix **radian**.
  - D) Create a 4 matrix call **table** that stores all angle values in degrees from matrix **A** and in radians from matrix **radian** in 2 columns. (Hint: Use transpose)

# Appendices

## Appendix A - Exploration #1: Computer Graphics [7]



## DISCOVERY PROJECT

### Computer Graphics I

Matrix algebra is the basic tool used in computer graphics to manipulate images on a computer screen. We will see how matrix multiplication can be used to “move” a point in the plane to a prescribed location. Combining such moves enables us to stretch, compress, rotate, and otherwise transform a figure, as we see in the images below.



Image



Compressed



Rotated



Sheared

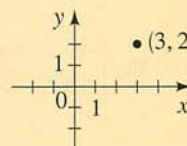
### Moving Points in the Plane

Let's represent the point  $(x, y)$  in the plane by a  $2 \times 1$  matrix:

$$(x, y) \leftrightarrow \begin{bmatrix} x \\ y \end{bmatrix}$$

For example, the point  $(3, 2)$  in the figure is represented by the matrix

$$P = \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$

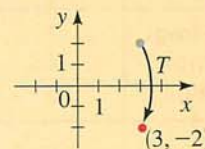


Multiplying by a  $2 \times 2$  matrix *moves* the point in the plane. For example, if

$$T = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

then multiplying  $P$  by  $T$  we get

$$TP = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 3 \\ 2 \end{bmatrix} = \begin{bmatrix} 3 \\ -2 \end{bmatrix}$$



We see that the point  $(3, 2)$  has been moved to the point  $(3, -2)$ . In general, multiplication by this matrix  $T$  reflects points in the  $x$ -axis. If every point in an image is multiplied by this matrix, then the entire image will be flipped upside down about the  $x$ -axis. Matrix multiplication “transforms” a point to a new point in the plane. For this reason, a matrix used in this way is called a **transformation**.

Table 1 gives some standard transformations and their effects on the gray square in the first quadrant.

Table 1

| Transformation matrix  | Effect |  |
|--|--------|--|
| $T = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$<br>Reflection in $x$ -axis                         |        |  |
| $T = \begin{bmatrix} c & 0 \\ 0 & 1 \end{bmatrix}$<br>Expansion (or contraction) in the $x$ -direction |        |  |
| $T = \begin{bmatrix} 1 & c \\ 0 & 1 \end{bmatrix}$<br>Shear in $x$ -direction                          |        |  |

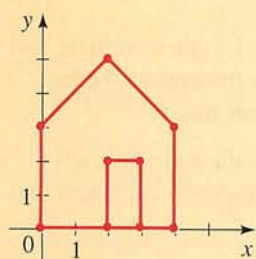


Figure 1

### Moving Images in the Plane

Simple line drawings such as the house in Figure 1 consist of a collection of vertex points and connecting line segments. The entire image in Figure 1 can be represented in a computer by the  $2 \times 11$  **data matrix**

$$D = \begin{bmatrix} 2 & 0 & 0 & 2 & 4 & 4 & 3 & 3 & 2 & 2 & 3 \\ 0 & 0 & 3 & 5 & 3 & 0 & 0 & 2 & 2 & 0 & 0 \end{bmatrix}$$

The columns of  $D$  represent the vertex points of the image. To draw the house, we connect successive points (columns) in  $D$  by line segments. Now we can transform the whole house by multiplying  $D$  by an appropriate transformation matrix. For

example, if we apply the shear transformation  $T = \begin{bmatrix} 1 & 0.5 \\ 0 & 1 \end{bmatrix}$ , we get the following matrix.

$$\begin{aligned} TD &= \begin{bmatrix} 1 & 0.5 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 2 & 0 & 0 & 2 & 4 & 4 & 3 & 3 & 2 & 2 & 3 \\ 0 & 0 & 3 & 5 & 3 & 0 & 0 & 2 & 2 & 0 & 0 \end{bmatrix} \\ &= \begin{bmatrix} 2 & 0 & 1.5 & 4.5 & 5.5 & 4 & 3 & 4 & 3 & 2 & 3 \\ 0 & 0 & 3 & 5 & 3 & 0 & 0 & 2 & 2 & 0 & 0 \end{bmatrix} \end{aligned}$$



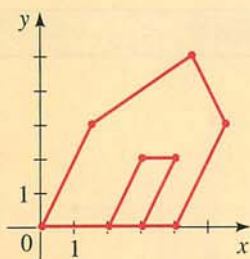


Figure 2

```

PROGRAM:IMAGE
:For(N,1,10)
:Line([A])(1,N),
[A](2,N),[A](1,N+1),
[A](2,N+1))
:End

```

To draw the image represented by  $TD$ , we start with the point  $\begin{bmatrix} 2 \\ 0 \end{bmatrix}$ , connect it by a line segment to the point  $\begin{bmatrix} 0 \\ 0 \end{bmatrix}$ , then follow that by a line segment to  $\begin{bmatrix} 1.5 \\ 3 \end{bmatrix}$ , and so on. The resulting tilted house is shown in Figure 2.

A convenient way to draw an image corresponding to a given data matrix is to use a graphing calculator. The TI-83 program in the margin converts a data matrix stored in  $[A]$  into the corresponding image, as shown in Figure 3. (To use this program for a data matrix with  $m$  columns, store the matrix in  $[A]$  and change the “10” in the For command to  $m - 1$ .)

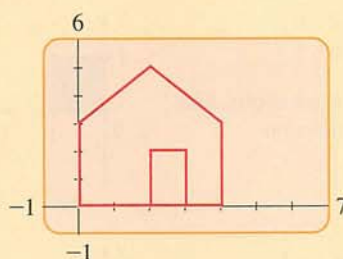
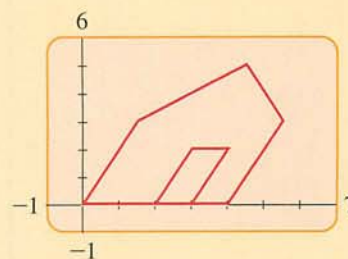
(a) House with data matrix  $D$ (b) Tilted house with data matrix  $TD$ 

Figure 3

We will revisit computer graphics in the *Discovery Project* on page 792, where we will find matrices that rotate an image by any given angle.

1. The gray square in Table 1 has the following vertices.

$$\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Apply each of the three transformations given in Table 1 to these vertices and sketch the result, to verify that each transformation has the indicated effect. Use  $c = 2$  in the expansion matrix and  $c = 1$  in the shear matrix.

2. Verify that multiplication by the given matrix has the indicated effect when applied to the gray square in the table. Use  $c = 3$  in the expansion matrix and  $c = 1$  in the shear matrix.

$$T_1 = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$$

Reflection in y-axis

$$T_2 = \begin{bmatrix} 1 & 0 \\ 0 & c \end{bmatrix}$$

Expansion (or contraction) in y-direction

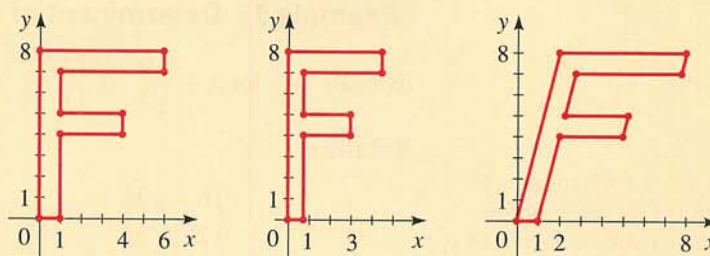
$$T_3 = \begin{bmatrix} 1 & 0 \\ c & 1 \end{bmatrix}$$

Shear in y-direction

3. Let  $T = \begin{bmatrix} 1 & 1.5 \\ 0 & 1 \end{bmatrix}$ .

- (a) What effect does  $T$  have on the gray square in the Table 1?

- (b) Find  $T^{-1}$ .
- (c) What effect does  $T^{-1}$  have on the gray square?
- (d) What happens to the square if we first apply  $T$ , then  $T^{-1}$ ?
4. (a) Let  $T = \begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix}$ . What effect does  $T$  have on the gray square in Table 1?
- (b) Let  $S = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$ . What effect does  $S$  have on the gray square in Table 1?
- (c) Apply  $S$  to the vertices of the square, and then apply  $T$  to the result. What is the effect of the combined transformation?
- (d) Find the product matrix  $W = TS$ .
- (e) Apply the transformation  $W$  to the square. Compare to your final result in part (c). What do you notice?
5. The figure shows three outline versions of the letter **F**. The second one is obtained from the first by shrinking horizontally by a factor of 0.75, and the third is obtained from the first by shearing horizontally by a factor of 0.25.
- (a) Find a data matrix  $D$  for the first letter **F**.
- (b) Find the transformation matrix  $T$  that transforms the first **F** into the second. Calculate  $TD$  and verify that this is a data matrix for the second **F**.
- (c) Find the transformation matrix  $S$  that transforms the first **F** into the third. Calculate  $SD$  and verify that this is a data matrix for the third **F**.



6. Here is a data matrix for a line drawing.

$$D = \begin{bmatrix} 0 & 1 & 2 & 1 & 0 & 0 \\ 0 & 0 & 2 & 4 & 4 & 0 \end{bmatrix}$$

- (a) Draw the image represented by  $D$ .
- (b) Let  $T = \begin{bmatrix} 1 & 1 \\ 0 & -1 \end{bmatrix}$ . Calculate the matrix product  $TD$  and draw the image represented by this product. What is the effect of the transformation  $T$ ?
- (c) Express  $T$  as a product of a shear matrix and a reflection matrix. (See problem 2.)



## Appendix B - Exploration #2: Matrices and Cryptography [8]

# Graphing Technology Lab

## Matrices and Cryptography

**Objective**

Use a graphing calculator and matrices to encode and decode messages.

*Cryptography* is the study of coded messages. Matrices can be used to code messages so that they can only be read after being deciphered by a *key*.

The first step is to decide on a key that can be used to encode the matrix. The key must be an  $n \times n$  invertible matrix. The next step is to convert the message to numbers and write it as a matrix. Each letter of the alphabet is represented by a number. The number 0 is used to represent a blank space.

| A | B | C | D | E | F | G | H | I | J  | K  | L  | M  |
|---|---|---|---|---|---|---|---|---|----|----|----|----|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |

| N  | O  | P  | Q  | R  | S  | T  | U  | V  | W  | X  | Y  | Z  |
|----|----|----|----|----|----|----|----|----|----|----|----|----|
| 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 |

Finally, the message is encoded by multiplying it by the key.

### ACTIVITY 1 Encode a Message

Use  $\begin{bmatrix} 1 & -2 \\ -1 & 3 \end{bmatrix}$  to encode the message SATURDAY AT NOON.

**Step 1** Convert the message to numbers and write it as a matrix.

S A T U R D A Y \_ A T \_ N O O N  
19 1 20 21 18 4 1 25 0 1 20 0 14 15 15 14

The key is a  $2 \times 2$  matrix. To make the matrix multiplication possible, write the message as an  $8 \times 2$  matrix.

$$\begin{bmatrix} 19 & 1 \\ 20 & 21 \\ 18 & 4 \\ 1 & 25 \\ 0 & 1 \\ 20 & 0 \\ 14 & 15 \\ 15 & 14 \end{bmatrix}$$

**Step 2** Multiply the converted message by the key using a graphing calculator.

$$\begin{bmatrix} 18 & -35 \\ -1 & 23 \\ 14 & -24 \\ -24 & 73 \\ -1 & 3 \\ 20 & -40 \\ -1 & 17 \end{bmatrix} \downarrow$$

**Step 3** Remove the matrix notation to reveal the encoded message.

18 -35 -1 23 14 -24 -24 73 -1 3 20 -40 -1 17 1 12

**StudyTip**

**Conversion** Add zeros to the end of a message if additional entries are needed to fill a matrix.

**Exercises**

Use  $\begin{bmatrix} 3 & 5 \\ -2 & -3 \end{bmatrix}$  to encode each message.

1. CALL ME

2. SEE YOU LATER

3. ORDER PIZZA

4. **CHALLENGE** Use  $\begin{bmatrix} 2 & -4 & 3 \\ -1 & 5 & 1 \\ 6 & -2 & 4 \end{bmatrix}$  to encode the message MEET ME AT FIVE.

To decode a message, the inverse of the key must be found. The coded message is then written in matrix form to make the multiplication possible. For instance, if the key is an  $n \times n$  matrix, the message is written as a  $k \times n$  matrix, where  $k$  is the number of rows necessary to include each number in the matrix. If there are not enough characters to fill a row, insert "0"s as spaces. Finally, the coded matrix is multiplied by the inverse of the key.

## ACTIVITY 2 Decode a Message

Use the inverse of  $\begin{bmatrix} -1 & -1 & -3 \\ 3 & 6 & 4 \\ 2 & 1 & 8 \end{bmatrix}$  to decode the message 38 83 39 77 99 202.

**Step 1** Use a graphing calculator to find the inverse of the key.

```
[A]-1
[[ -44  -5 -14]
 [ 16   2   5]
 [ 9    1   3]]
```

**Step 2** Write the coded message as a matrix. The coded matrix will have 3 columns because the key is a  $3 \times 3$  matrix. It is a  $2 \times 3$  matrix because there are enough numbers to fill two rows. Enter it into your graphing calculator.

```
MATRIX[B] 2 × 3
[[ 38  83  39]
 [ 77  99  202]]
2, 3=202
```

**Step 3** Use a graphing calculator to multiply the coded matrix by the inverse of the key.

```
[B][A]-1
[[ 7  15  0]
 [14  15  23]]
```

**Step 4** Remove the matrix notation and convert the numbers to letters.

```
7  15  0  14  15  23
G  O  _  N  O  W
```

## Exercises

Use the inverse of  $\begin{bmatrix} 12 & -7 \\ -5 & 3 \end{bmatrix}$  to decode each message.

- 128 -73 232 -135 300 -175 99 -56 83 -48 180 -104 300 -175
- 27 17 38 -21 84 -49 21 -11 131 -76 201 -116 161 -93
- 151 -88 150 -86 93 -54 -35 22 -5 3 191 -111 -30 18 182 -105
- 102 -58 45 -26 -48 29 -69 42 39 -21 228 -133 141 -81 -19 12 228 -133

9. **CHALLENGE** Use the inverse of  $\begin{bmatrix} 2 & 4 & 6 & 0 \\ 1 & 8 & -4 & -6 \\ 7 & 6 & -5 & 3 \\ 1 & 7 & 9 & 2 \end{bmatrix}$  to decode

```
126 265 -49 -34 198 347 193 96 174 239 49 72 177 286 -61 -27 48 200 70 -76 122 162
-21 35 81 190 -37 -63 130 331 214 17 67 267 94 -25 93 161 120 25.
```

## References

- [1] *MATLAB Primer, R2013a*. MathWorks Product Documentation.  
[http://www.mathworks.com/help/pdf\\_doc/matlab/getstart.pdf](http://www.mathworks.com/help/pdf_doc/matlab/getstart.pdf)
- [2] Neuman, Edward. *Tutorial 1: Getting Started with MATLAB*.  
<http://www.math.siu.edu/matlab/tutorial1.pdf>
- [3] Neuman, Edward. *Tutorial 1: Getting Started with MATLAB*.  
<http://www.math.siu.edu/matlab/tutorial2.pdf>
- [4] Arnold, David. *Plotting in MATLAB*. College of the Redwoods.  
<http://msemac.redwoods.edu/darnold/math50c/matlab/plotting/index.php>
- [5] Moore, Holly. *MATLAB for Engineers*. 2nd ed. New York: Prentice Hall, 2009
- [6] Attaway, Storm, and Butterworth-Heinemann. *MATLAB: A Practical Introduction to Programming and Problem solving*.
- [7] Stewart, James, and L. Redlin. *Precalculus: Mathematics for Calculus*. 5th ed. Australia: Brooks/Cole, 2010. p.700-703.
- [8] *Precalculus*. Columbus, OH: Glencoe/McGraw-Hill, 2011. p.395-396.