FreeMat Tutorial

FreeMat is a general purpose matrix calculator. It allows you to enter matrices and then perform operations on them in the same way you would write the operations on paper. This handout will guide you through the basic operations. Here is an example of how you might use it. Suppose you want to solve the system of equations

$$\begin{aligned} 3x + 4y - 2z &= 5 \\ 2x - 5y + z &= 8 \\ x + 3y &= -1 \end{aligned}$$

In matrix notation, it is expressed $A\mathbf{x} = \mathbf{b}$ where $A = \begin{bmatrix} 3 & 4 & -2 \\ 2 & -5 & 1 \\ 1 & 3 & 0 \end{bmatrix}$ and $\mathbf{b} = \begin{bmatrix} 5 \\ 8 \\ -1 \end{bmatrix}$. By hand, you would solve the system by forming the augmented matrix $[A \mid \mathbf{b}]$ and row reducing. Below are steps to do the same operations using FreeMat. Follow along in FreeMat, performing each step as you go.

1. Define *A* in FreeMat by typing "`A = [3 4 -2;2 -5 1; 1 3 0]`" and the enter key. Then you should see on the screen.

```
--> A = [3 4 -2;2 -5 1;1 3 0]

A =

  3  4 -2
  2 -5  1
  1  3  0
```

   In the remaining steps, you always type what you see following `-->`, and the rest is what FreeMat displays.

2. Define **b:**

```
--> b = [5;8;-1]

b =

  5
  8
 -1
```

3. Define the augmented matrix:

```
--> [A b]

ans =

  3  4 -2  5
  2 -5  1  8
  1  3  0 -1
```

Here, **ans** is FreeMat's name for the most recent computation. In the next command we will use **ans** to represent the augmented matrix.

4. Use row operations to reduce the augmented matrix:

```
--> rref(ans)

ans =

    1.0000         0         0    2.1111
         0    1.0000         0   -1.0370
         0         0    1.0000   -1.4074
```

Now you can read off the solutions to the equations: $x = 2.1111$, $y = -1.0370$, and $z = -1.4074$.

5. You can also check that this is correct. First, we need to define **x** as the last column of the reduced matrix. FreeMat has a command for extracting rows and columns of an existing matrix. Here is how we get column 4:

```
--> x = ans(:,4)

x =

    2.1111
   -1.0370
   -1.4074
```

Here, **ans**(:,4) is like **ans**(*i*,*j*). The colon indicates that *i* can be anything and the 4 says to take only $j = 4$.

6. Does $A\mathbf{x}$ equal **b**? Use FreeMat to compute both of them:

```
--> A*x

ans =

    5.0000
    8.0000
   -1.0000

--> b

ans =

  5
  8
 -1
```

7. Alternatively, you could have FreeMat print out **b**, *A***x**, and their difference, side by side:

```
--> [b A*x b-A*x]

ans =

    5.0000    5.0000         0
    8.0000    8.0000    0.0000
   -1.0000   -1.0000         0
```

8. Freemat can find the inverse of *A*:

```
--> inv(A)

ans =

    0.1111    0.2222    0.2222
   -0.0370   -0.0741    0.2593
   -0.4074    0.1852    0.8519
```

9. Let's test that this inverse is correct, by computing …

```
--> inv(A)*A

ans =

    1.0000         0         0
   -0.0000    1.0000   -0.0000
    0.0000         0    1.0000
```

10. An alternative solution of *A***x** = **b** is given by **x** = inv(*A*)***b.** In FreeMat:

```
--> inv(A)*b

ans =

    2.1111
   -1.0370
   -1.4074
```

11. How does that compare with the answer we found earlier? FreeMat still remembers that we defined the answer as **x**, so let's compare …

```
--> [inv(A)*b x]

ans =

    2.1111    2.1111
   -1.0370   -1.0370
   -1.4074   -1.4074
```

12. Both methods seem to give the same answer. But if you subtract them, you get:

```
--> inv(A)*b - x

ans =

   1.0e-016 *

    4.4409
   -2.2204
    8.8818
```

This shows that the answers are slightly different. The 1.0e-016 * means that each of the entries below it should be multiplied by 10^{-16}. So the *x* we found using row operations differs from the *x* we found using the inverse of *A* by .00000000000000044409.

More Operations

The commands and format of FreeMat are almost identical to another matrix software product called MATLAB. To provide an overview of additional FreeMat operations, I have adapted a MATLAB tutorial by Tom Nguyen (online at http://edu.levitas.net/Tutorials/Matlab/). What follows is a slightly modified version of the original MATLAB tutorial. I added some comments and changed each appearance of the word MATLAB into FreeMat. As you read, try out the commands shown in the examples.

As you have already seen, FreeMat prompts each input line with `-->`. MATLAB has a similar arrangement, but uses >> as the prompt. This appears in all of the examples below. When you try commands, disregard the >> and start typing at the `-->` prompt.

Start of the modified MATLAB Tutorial.

One of the easy ways to learn FREEMAT is to understand how FREEMAT handles matrices. Think in terms of arrays and vectors when you work with FREEMAT. For example, an array of data A = 1, 0, 9, 11, 5 is a 1x5 matrix, and a scalar number 9 is a 1x1 matrix. To store the array A in FREEMAT, at the command prompt >> enter:

```
» A=[1 0 9 11 5]
```

FREEMAT will display or echo your input:

```
A =

     1     0     9    11     5
```

To suppress the echo, add a "**;**" at the end of the input line.

To verify the size of the input array or matrix, use the command "size" as shown below:

```
» size(A)

ans =

     1     5
```

which verifies the dimension of matrix A as 1x5 (one row and five columns).

In FREEMAT, rows are separated by ";" and columns are separated by ",". For example, a 3x5 matrix **B** with the following elements:

first row: 1, 0, 9, 4, 3
second row: 0, 8, 4, 2, 7
third row: 14, 90, 0, 43, 25

would be entered in FREEMAT as follow:

```
» B=[1,0,9,4,3;0,8,4,2,7;14,90,0,43,25]

B =

     1     0     9     4     3
     0     8     4     2     7
    14    90     0    43    25
```

Note that you may use a space in place of the comma in separating the column entries.

You may add, subtract, multiply, and divide matrices with simple operations in FREEMAT. Please keep in mind the rules concerning matrix operations. For example, to add matrices, they must have the same dimensions. To multiply *A*B*, the number of columns in *A* must equal the number of rows in *B*.

You may extract a certain group of elements from an existing matrix. Say you wish to create a new array C from the second row of matrix B. Specify the row number and ":" for all columns in that row as shown below:

```
» C=B(2,:)

C =

     0     8     4     2     7
```

To get columns 3, 4, and 5, put : in for the rows and put 3:5 in for the columns.

```
» C=B(:,3:5)

C =

     0     9     4
     8     4     2
    90     0    43
```

Similarly, you may also form a matrix from the elements of an existing matrix:

```
» D=[B(1,2),B(1,4);B(3,2),B(3,5)]

D =

     0     4
    90    25
```

Here, a square matrix D has been created from the specified elements of matrix B.

You may also delete rows and columns from a matrix using a pair of square brackets. For example, to delete the third column of matrix B, you simply enter

```
» B(:,3)=[]
```

and FREEMAT will return:

```
B =

     1     0     4     3
     0     8     2     7
    14    90    43    25
```

Note that the column 3 is excluded.

FREEMAT can perform arithmetic operations on arrays in an element-by-element manner. This operation is accomplished by including a dot or a period before the arithmetic operator such as multiplication, division, etc. The table below gives a list of both regular operators and the element-by-element operators with examples of how the operators are used.

To perform the examples in the table below, enter the following matrices:

```
A =

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

```
B =

     2     4     6
     0     3     7
     9     8     1
```

Operation	Description	Example (FREEMAT actual inputs and outputs)
+	Addition	>> C=A+B C = 3 6 9 4 8 13 16 16 1
-	Substraction	>> C=A-B C = -1 -2 -3 4 2 -1 -2 0 -1
*	Multiplication	>> C=A*B C = 29 34 23 62 79 65 14 52 98
.*	Element-by-element multiplication. Note that this is different from multiplication of two matrices.	» C=A.*B C = 2 8 18 0 15 42 63 64 0 Note that this is not the same as **C = A*B**
/	Right matrix division. Dividing matrix **B** into matrix **A;** same as computing A*inv(B)	>> C=A/B C = 0.5000 0 0 3.6875 -2.2500 -0.3750 -8.3125 6.7500 2.6250

\	Left matrix division. Dividing **A** into **B**. This is equivalent to **inv(A)*B**. Note that **X** = **C** is the solution to **A***X=**B**	>> C=A\B C = -4.5556 -5.3333 -4.5556 5.1111 5.6667 4.1111 -1.2222 -0.6667 0.7778 >> X=inv(A)*B X = -4.5556 -5.3333 -4.5556 5.1111 5.6667 4.1111 -1.2222 -0.6667 0.7778
./	Element-by-element division note that **D**(2,1) is undefined due to the zero at **B**(2,1)	» D=A./B Warning: Divide by zero. D = 0.5000 0.5000 0.5000 Inf 1.6667 0.8571 0.7778 1.0000 0
.\	Element-by-element left division. Note that left division in this particular example means elements of B divided by the corresponding elements of A.	» E=A.\B Warning: Divide by zero. E = 2.0000 2.0000 2.0000 0 0.6000 1.1667 1.2857 1.0000 Inf
.^	Element-by-element power	» F=A.^B F = 1 16 729 1 125 279936 40353607 16777216 0

Transposing a matrix in FREEMAT involves a simple prime notation (') after the defined matrix as shown below:

Example:

```
>> A_transpose=A'

A_transpose =

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

The inverse of matrix **A** can be obtained with the command:

```
>> inv(A)

ans =

  -1.7778   0.8889  -0.1111
   1.5556  -0.7778   0.2222
  -0.1111   0.2222  -0.1111
```

Eigenvalues and Eigenvectors can easily be obtained with the command **[V,E]=eig(***matrix name***)**:

```
>> [V,E]=eig(A)

V =

  -0.2998  -0.7471  -0.2763
  -0.7075   0.6582  -0.3884
  -0.6400  -0.0931   0.8791


E =

  12.1229       0       0
        0  -0.3884       0
        0       0  -5.7345
```

where the columns of V are the eigenvectors and the corresponding diagonal enries of E are the eigenvalues. Eigenvalues alone can be obtained without the notation "[V,E]":

```
>> E=eig(A)

E =

  12.1229
  -0.3884
  -5.7345
```