

Chapter Three

Processes on Matrices

3.1 One dimensional matrices

Any set of numbers in horizontal appearance (row) or vertical appearance (column) is a one dimensional matrix or (array). Each element in the array has a value and a position (index or argument):

```
x = [ 4 , 5 , 0 , 12 , -7 , 145 , 52.66 , 10 ] ↵  
x =  
    4  5  0  12  -7  145  52.66  10
```

If we want to recall any element, its position must be identified:

```
y = x ( 1 ) ↵           y = x ( 5 ) ↵           y = x ( end ) ↵  
y =                     y =                     y =  
    4                     -7                     10
```

The vertical display can be either obtained using semicolon (;) or single quote ('):

```
x = [ 4 ; 3 ; 0 ; 12 ; -7 ] ↵           x = [ 4 , 3 , 0 , 12 , -7 ] ' ↵  
x =                                     x =  
    4                                     4  
    3                                     3  
    0                                     0  
   12                                     12  
   -7                                     -7
```

The command (length) can be used to recall the number of elements:

```
x = [ 1  3  -3  7  4 ];  
z = length ( x ) ↵  
z =  
    5
```

3.2 Two dimensional matrices

In two dimensional matrices the numbers are displayed in the form of rows and columns. The semicolon is used to define a new row:

```
x = [ 2, 5, 7; 1, 1, 1; 0, 0, 0 ] ↵      x = [ 2 5 7; 1 1 1; 0 0 0 ] ↵
x =                                     x =
    2  5  7                             2  5  7
    1  1  1                             1  1  1
    0  0  0                             0  0  0
```

To recall an element in a two dimensional matrix the row and column numbers must be identified with a comma between them:

```
y = x ( 2 , 3 ) ↵      y = x ( 1 , 3 ) ↵      y = x ( end , end ) ↵
y =                    y =                    y =
    1                    7                    0
```

The command (size) can be used to recall the number of rows and columns:

```
x = [ 1:4; 2, 2, 2, 2; 5, 8, -9, 6 ];      x = [ 1:4; 2, 2, 2, 2; 5, 8, -9, 6 ];
y = size ( x ) ↵                          [ a , b ] = size ( x ) ↵
y =                                         a =          b =
    3  4                                    3          4
```

3.3 Manipulating matrices using colon operator

The colon operator (:) can be used to manipulate one and two dimensional matrices:

a) One dimensional matrices

```
x = [ 1, 5, 8, -3, 4, 9, 2, 6 ];      x = [ 1, 5, 8, -3, 4, 9, 2, 6 ];
y = x ( 3 : 6 ) ↵                      y = x ( 5 : end ) ↵
y =                                     y =
    8 -3  4  9                          4  9  2  6
```

b) Two dimensional matrices

```
x = [ 1:4; 5, 3, 2, 6; 1, 1, 1, 1 ] ↵
x =                                     y = x ( 1 , : ) ↵      y = x ( : , 4 ) ↵
    1  2  3  4                             y =
    5  3  2  6                             1  2  3  4
    1  1  1  1                             4
                                           6
                                           1
```

```
y = x(1:2, 2:3) ↵
```

```
y =
```

```
2 3
```

```
3 2
```

```
y = x(2:end, 3:end) ↵
```

```
y =
```

```
2 6
```

```
1 1
```

Note: the single quote (') can be used to transpose the two dimensional matrix, that is; the rows becomes columns and columns become rows:

```
x = [1 2 3; 4 5 6] ↵
```

```
x =
```

```
1 2 3
```

```
4 5 6
```

```
y = x' ↵
```

```
y =
```

```
1 4
```

```
2 5
```

```
3 6
```

Note: the commands (flipud) and (fliplr) can be used to mirror the two dimensional matrix around horizontal and vertical axes respectively:

```
x = [1 2 3; 4 5 6; 7 8 9] ↵
```

```
x =
```

```
1 2 3
```

```
4 5 6
```

```
7 8 9
```

```
y = flipud(x) ↵
```

```
y =
```

```
7 8 9
```

```
4 5 6
```

```
1 2 3
```

```
z = fliplr(x) ↵
```

```
z =
```

```
3 2 1
```

```
6 5 4
```

```
9 8 7
```

Note: the commands (ones) and (zeros) can be used to generate matrices of ones and zeros:

```
x = ones(2) ↵
```

```
x =
```

```
1 1
```

```
1 1
```

```
x = ones(1, 3) ↵
```

```
x =
```

```
1 1 1
```

```
x = zeros(2) ↵
```

```
x =
```

```
0 0
```

```
0 0
```

```
x = zeros(3, 2) ↵
```

```
x =
```

```
0 0
```

```
0 0
```

```
0 0
```