Creating Arrays

1-Dimensional Array: Vector

An array is a *named* collection of data values organized into rows and/or columns. A 1-d array is a row or a column, also known as a *vector*. An *index* is a positive integer that identifies the position of a value in the vector. MATLAB **array index starts at 1**, not zero. To access a value in an array, use parentheses to enclose the index value. For example, `x(2)` is the value in the 2nd cell of vector `x`. MATLAB distinguishes between *row* and *column* vectors. Numbers (or expressions) separated by commas or blanks and enclosed by *square brackets* give a *row* vector, while numbers separated by semicolons and enclosed by square brackets give a em column vector.

Creating a vector

MATLAB function **zeros**: **vecA= zeros(1,5)**

MATLAB function **ones**: **vecB= ones(5,1)**

MATLAB colon expression for consecutive numbers: **1:6** or **1:1:6**
Note that the syntax is ⟨*starting value*⟩**:**⟨*increment*⟩**:**⟨*ending bound*⟩ , so the expression **7:-2:0** gives [7 5 3 1]. What if the colon expression specifies an "impossible" set? E.g., **5:1:0** will result in the empty set, which is the empty vector [ ] in MATLAB.

Assignment: **vecC(5) = 9** gives [0 0 0 0 9]

Build vectors using square brackets: **vecD = [2 3.5 6]**
Use a blank or a comma as the separator to get a *row*; use a semi-colon as the separator to get a *column*.

Combine or concatenate vectors: [**[4 5]** **[1 3 2]**] gives [4 5 1 3 2]; **[4 5 9:-1:6]** gives [4 5 9 8 7 6].

"Grow" a vector: The statement **v= [v 9]** concatenates 9 to the end of vector **v** and re-assigns the entire new vector back to the name **v**. If you put this statement inside a loop, assuming that **v** has some intial value before the start of the loop, vector **v** will "grow" in length one cell at a time. Note that you can create an *empty vector*: **v= [ ]**. Such an assignment is sometimes used as an initialization for a variable.

Transpose a vector (change from row to column or vice versa): **'**
Example: **[3; 5; 1]'** gives the *row* vector [3 5 1]

2-Dimensional Array: Matrix

Type these statements in the MATLAB Command Window to see what matrices get created:

```
m= [1 2 3 4; 5 6 7 8]  % 2-by-4 matrix
[nr,nc]= size(m)
m= [m; zeros(1,nc)]
m= [m m]
m= [m; m]
v= 1:6
newm= [m v']
newm= newm'
m1= rand(4,3)  % 4-by-3 random matrix (uniform probability distribution)
```