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 column vector.

Creating a vector

MATLAB function \texttt{zeros}: \texttt{vecA= zeros(1,5)}

MATLAB function \texttt{ones}: \texttt{vecB= ones(5,1)}

MATLAB short-cut expression for consecutive numbers: \texttt{1:6} or \texttt{1:1:6}

Note that the syntax is \texttt{left bound};\texttt{increment};\texttt{right bound} , so the expression \texttt{7:-2:0} gives \([7 \hspace{1em} 5 \hspace{1em} 3 \hspace{1em} 1]\).

Assignment: \texttt{vecC(5) = 9} gives \([0 \hspace{1em} 0 \hspace{1em} 0 \hspace{1em} 0 \hspace{1em} 9]\)

Build vectors using square brackets: \texttt{vecD = [2 \hspace{1em} 3.5 \hspace{1em} 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: \([ [4 \hspace{1em} 5] [1 \hspace{1em} 3 \hspace{1em} 2] ]\) gives \([4 \hspace{1em} 5 \hspace{1em} 1 \hspace{1em} 3 \hspace{1em} 2] ; [4 \hspace{1em} 5 \hspace{1em} 9 \hspace{1em} 8 \hspace{1em} 7 \hspace{1em} 6] \)

“Grow” a vector: The statement \texttt{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 initial 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: \texttt{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: \([3 \hspace{1em} 5 \hspace{1em} 1]’\) gives the row vector \([3 \hspace{1em} 5 \hspace{1em} 1]\)

Example 1

Write a program fragment that calculates the cumulative sums of a given vector \( v \). The cumulative sums should be stored in a vector of the same length as \( v \). E.g., the cumulative sums for the sequence 1,3,5,0 is 1,4,9,9. Do not use any MATLAB predefined functions other than \texttt{length}. (The function call \texttt{length(v)} returns the length of vector \( v \).)

Example 2

Write a function \texttt{evalPoly} to evaluate an \( n \textsuperscript{th} \) order polynomial of \( x \):

\[ a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n \]

The input parameters are \texttt{coef} and \texttt{x} where \texttt{coef} has length \( n + 1 \) and contains the coefficients of the polynomial and \( x \) is the value at which to evaluate the polynomial. Return the evaluated value. Note that \texttt{coef(1)} is the coefficient for the term \( x^0 \). Do not use MATLAB predefined functions other than \texttt{length}. 
function val = evalPoly(coef,x)
% val is the value of a polynomial with coefficients coef evaluated at x.
% coef is a vector and coef(1) is the coefficient for the term x^0.

declare randomWalk function that performs n steps of a "random walk" starting from position $(x_0, y_0)$ and draws the path. In a random walk, possible moves are left, right, up, or down (in a Cartesian plane).

Example 3: A random walk with graphics

Write a function randomWalk that performs n steps of a “random walk” starting from position $(x_0, y_0)$ and draws the path. In a random walk, possible moves are left, right, up, or down (in a Cartesian plane).

function randomWalk(n,x0,y0)
% Perform n steps of random walk starting from position (x0,y0). Display the path.

% possible movements: ( deltaX(i), deltaY(i) )

deltaX=

deltaY=

x= [x0 zeros(1,n)]; % trajectory in x direction
y= [y0 zeros(1,n)]; % trajectory in y direction

% Perform walk, each step is based on a random integer
for k = 2:n+1
  % get a random integer in (1..4)
  r= % take the step
  x(k)=
  y(k)=
end

% Show the walk
plot(x,y) % x=,y=x1,y1,'r*',x(end),y(end),'ro')
axis('equal')
title([num2str(n) ' steps of random walk from * to o'])

Plotting

It is very easy to make plots using MATLAB. An x-y plot can be generated using the built-in function plot. The command

plot(a,b,'-', c,d,'*')

will generate a plot with two graphs, one showing the data contained in vectors a (x-coordinates) and b (y-coordinates) as a line and the other showing the data in vectors c and d as asterisks. Use the help facility in MATLAB to learn more about plot and the many formatting options. If you omit the formatting option ('-' and '*' above), the default on most system is to show the data as a line.