Matlab Review

NDSU ECE 463/663

Lecture #1

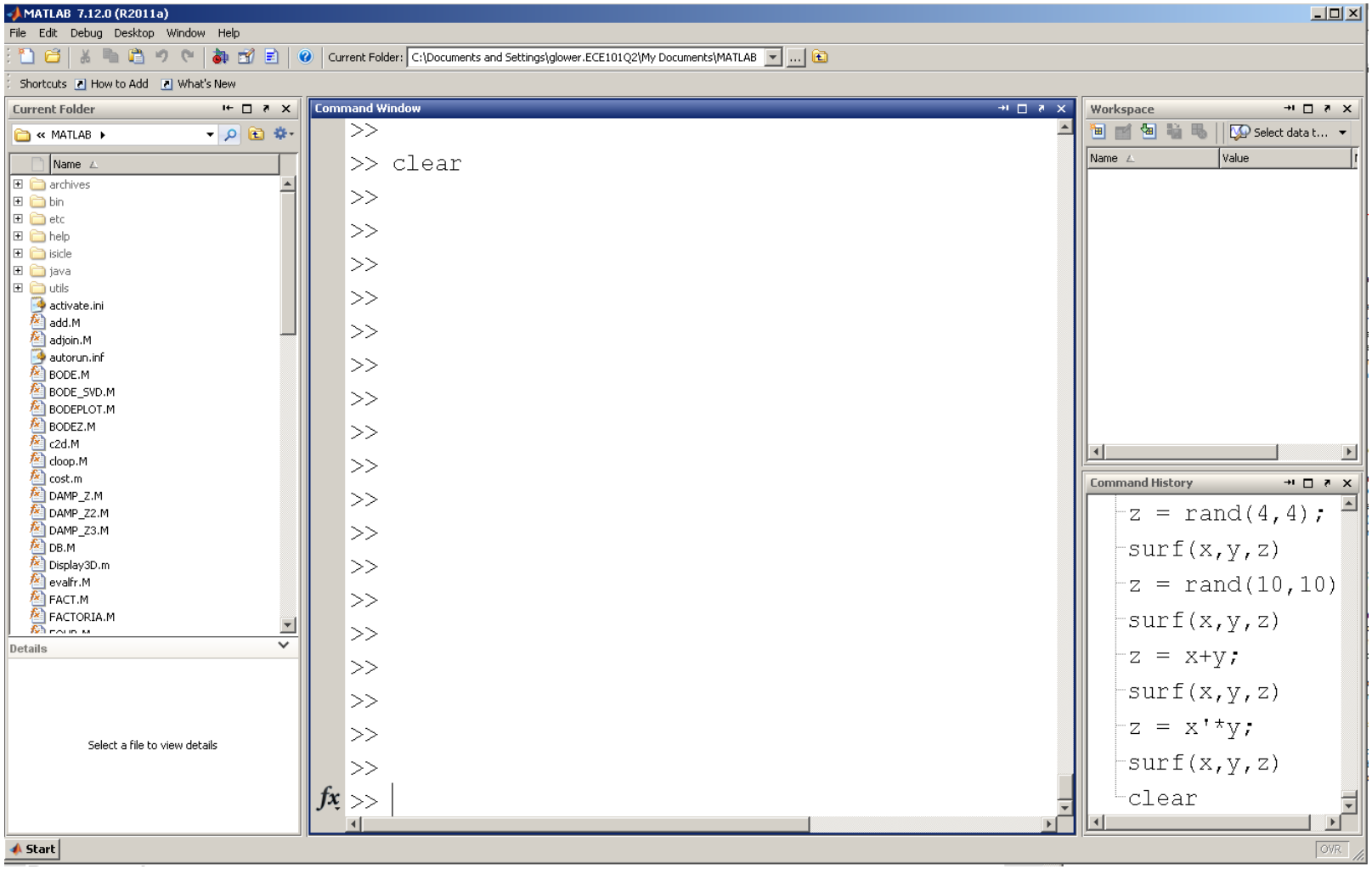
Inst: Jake Glower

Please visit Bison Academy for corresponding
lecture notes, homework sets, and solutions

Becoming familiar with MATLAB

- The console
- The editor
- The graphics windows
- The help menu
- Saving your data (diary)

General environment and the console



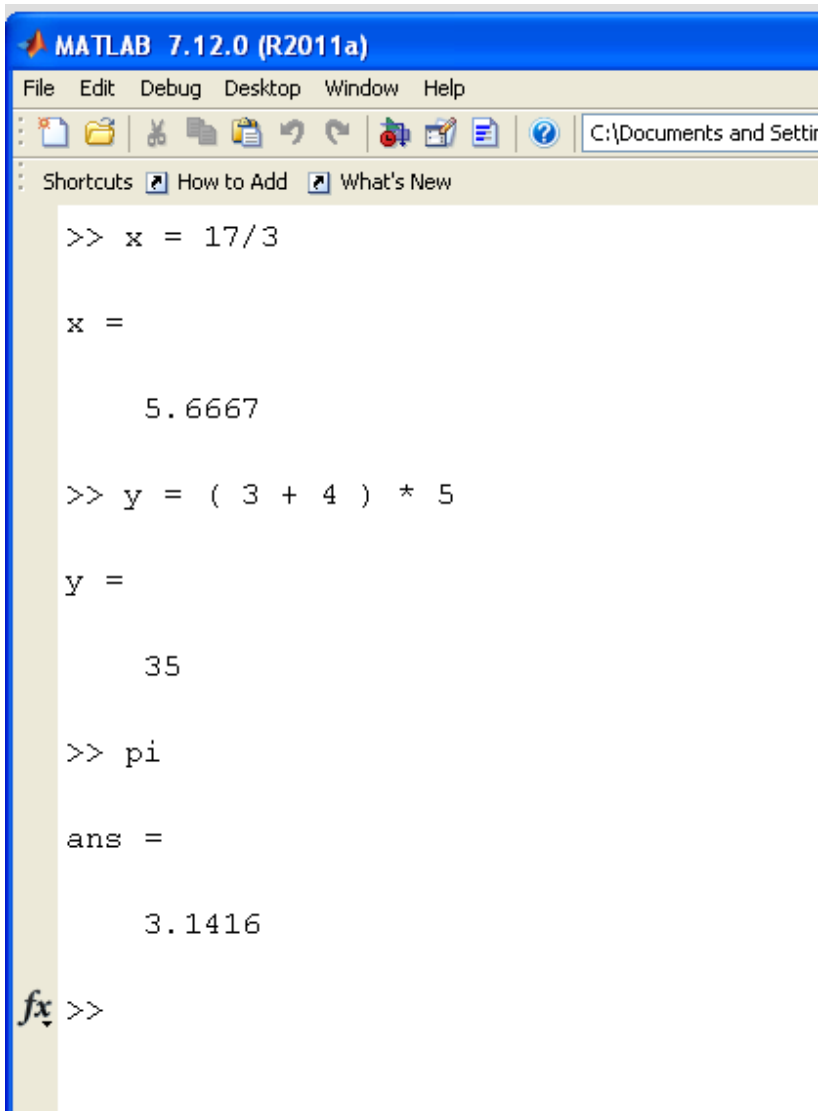
Command Window:

- Similar to a calculator

```
>> x = 17/3
    5.6667

>> y = (3+4)*5
    35
```

Particular numbers

```
>> pi
    3.1416

>> i
          0 + 1.0000i

>> j
          0 + 1.0000i
```

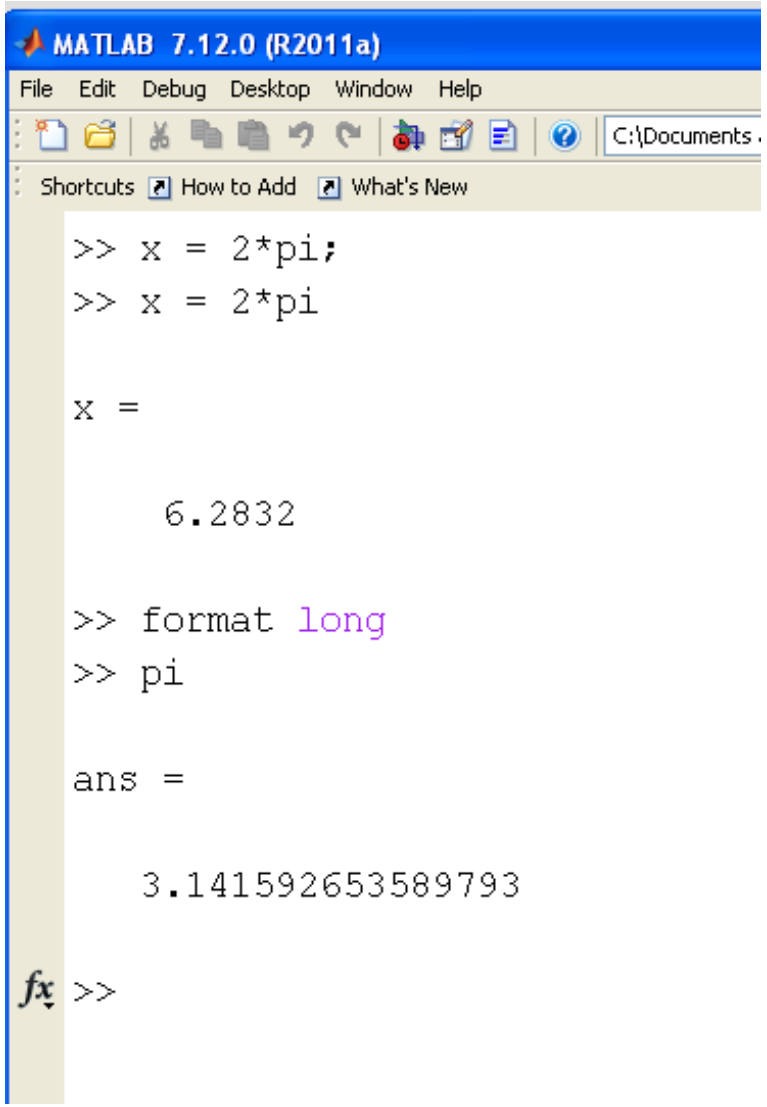


Do and don't display results

- 2*pi; execute but dont' display
- 2*pi execute and do display

Displaying number of decimal places

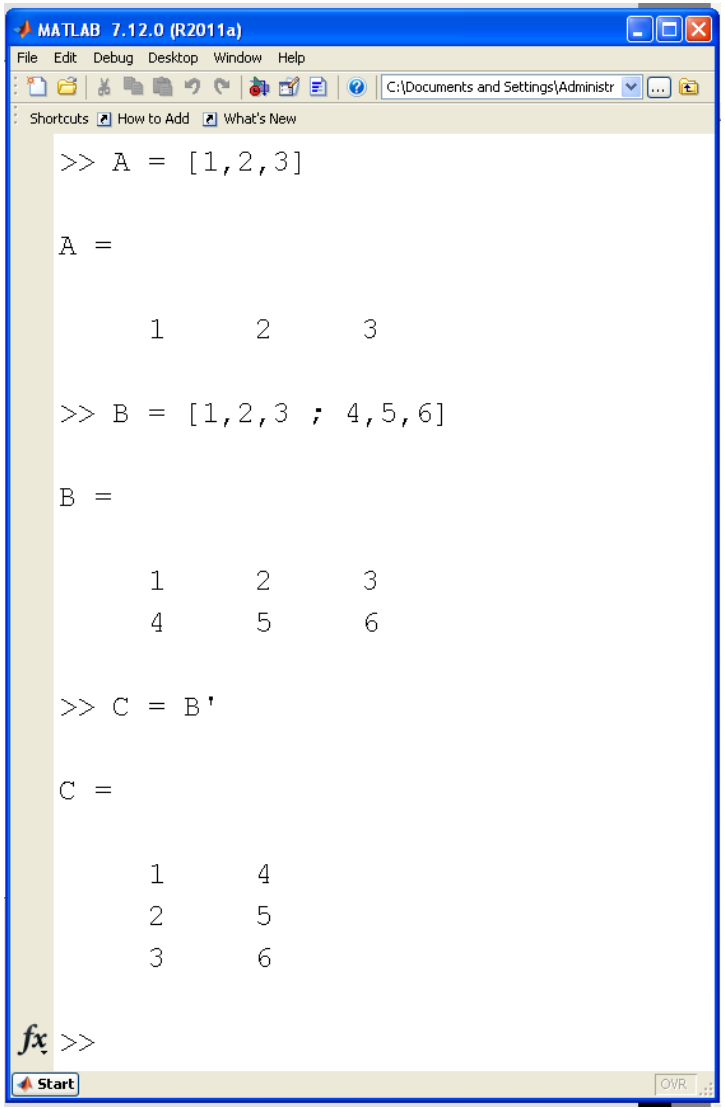
```
format short
format long
format longe
format shorteng
```



Matrices

- [start of matrix
-] end of matrix
- , next element
- ; next row

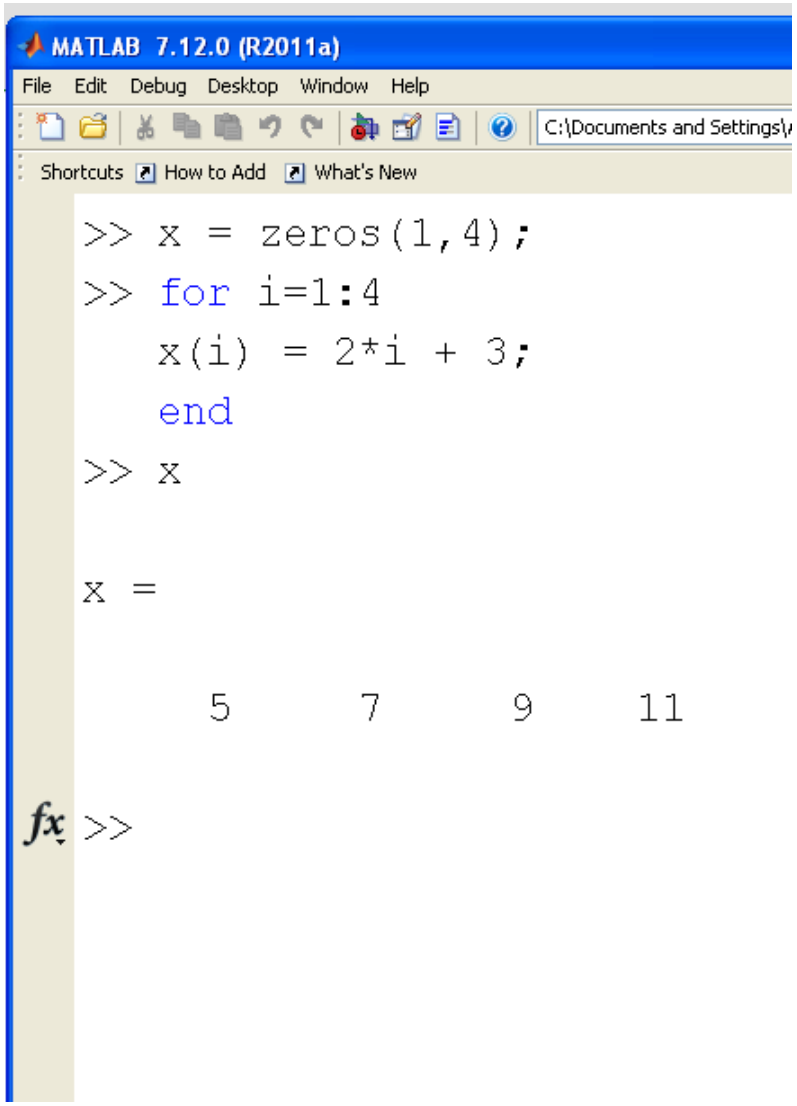
```
>> D = zeros(1,3)

     0     0     0

>> E = rand(3,2)

    0.5860    0.0835
    0.2467    0.6260
    0.6664    0.6609
```



for loop

Repeat a section of code a fixed number of times

```
for i=1:3
    x(i) = i^2;
    end
x

     1     4     9
```

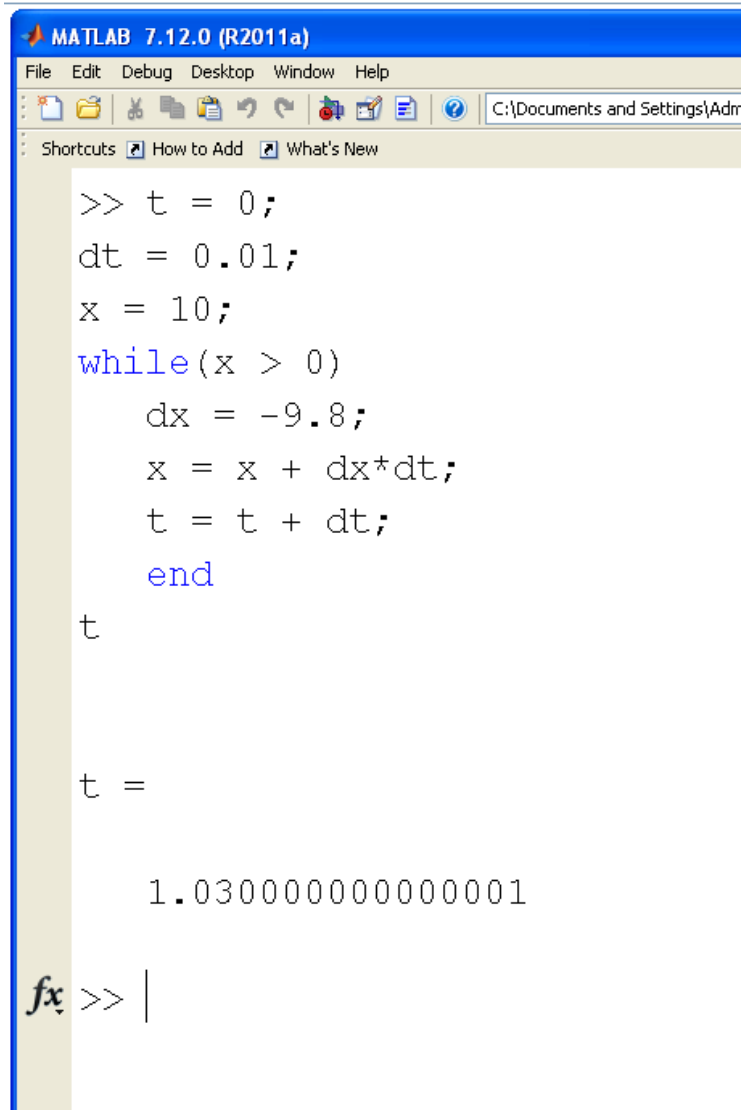


While Loop

Execute a section of code as long as the condition is true

Example: Determine how long it takes a ball to hit the ground when dropped from a height of 10m.

- Assume gravity = 9.8 m/s^2



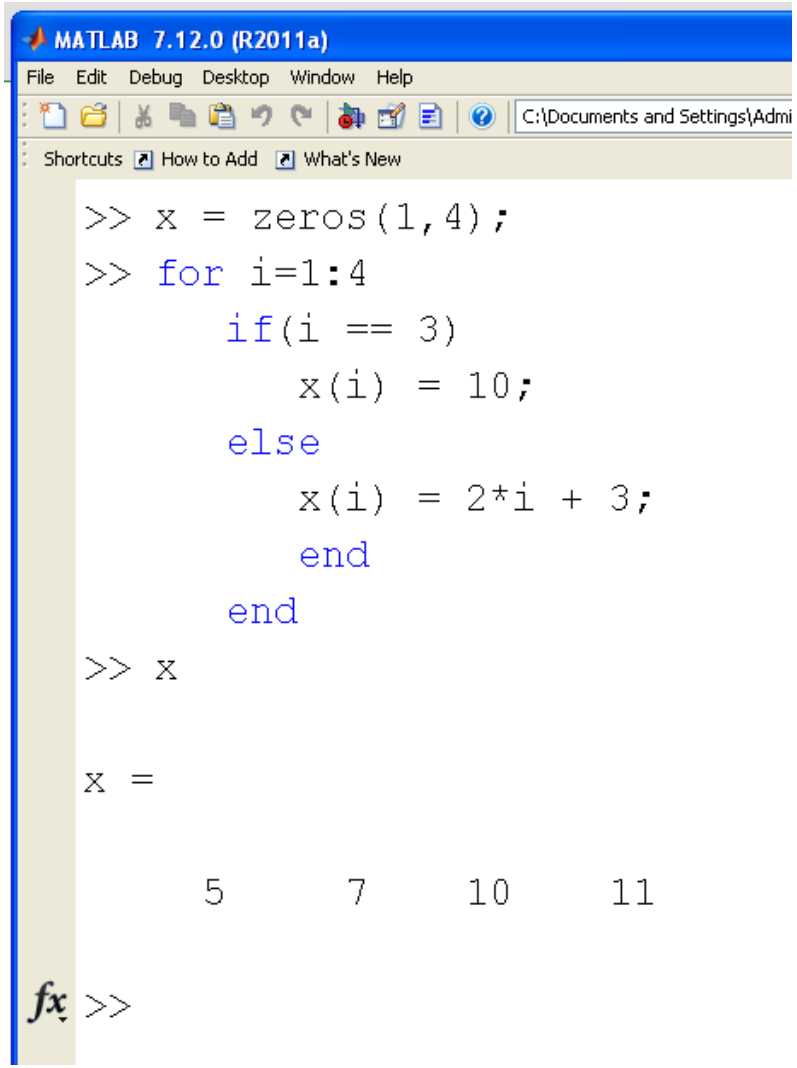
if Statements

- Execute if the condition is true
- Skips if false

if - else - end

- Executes first section if true

Executed *else* section if false

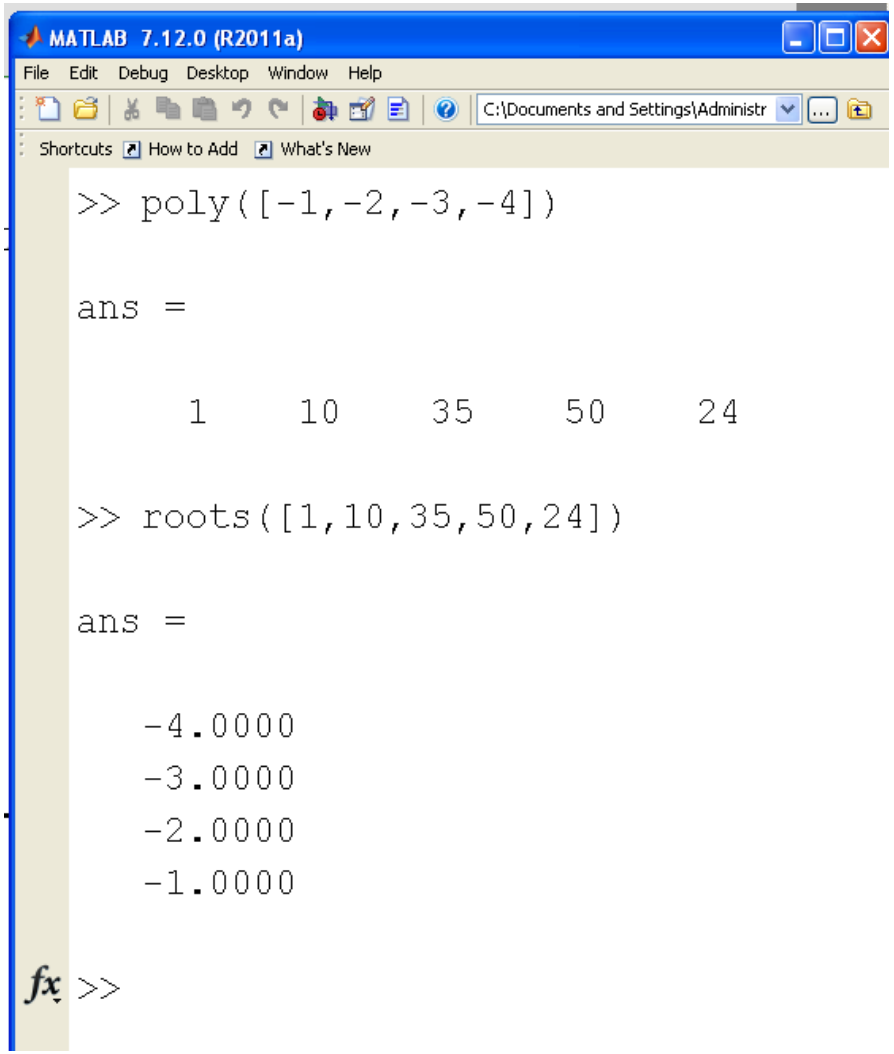


Controls Related Functions

poly: Create a polynomial with a given set of roots

- Result given decreasing powers of s
- $s^4 + 10s^3 + 35s^2 + 50s + 24$

roots: Roots to a polynomial



ss:

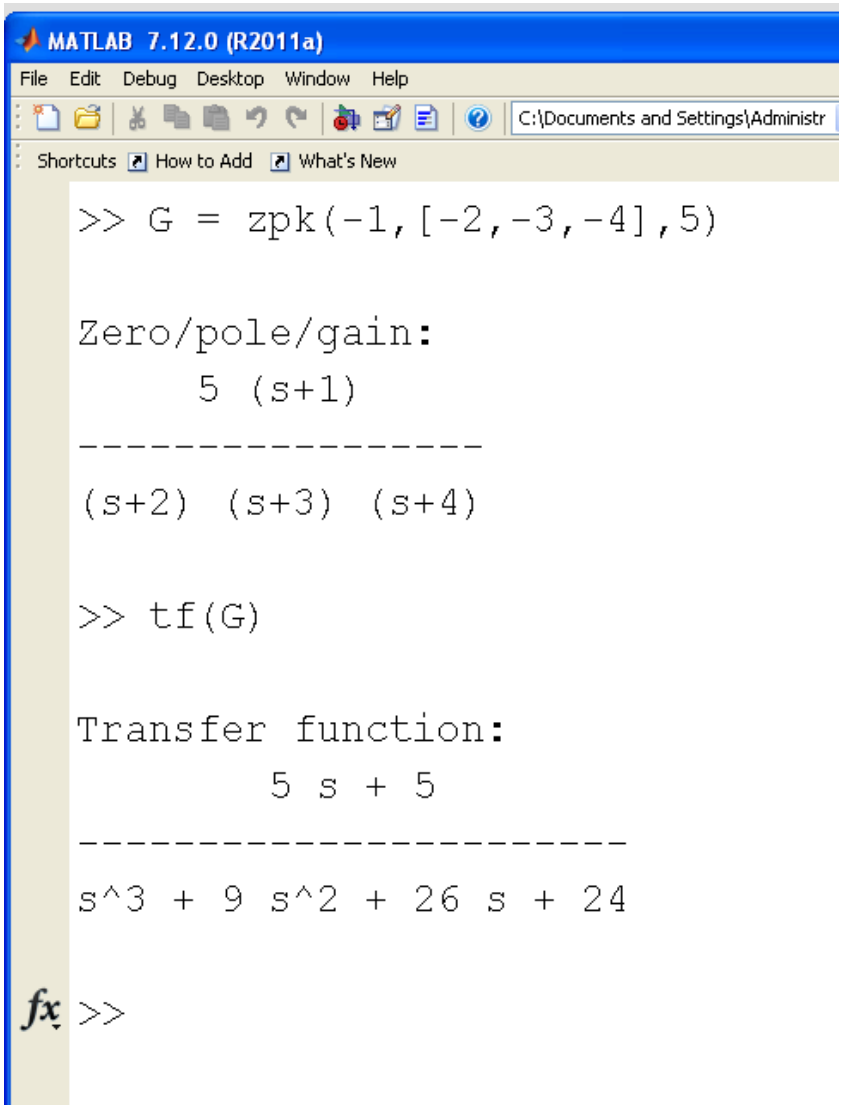
- Input a system in state-space form
- Convert to state-space form

zpk:

- Input a system as zeros - poles - gain
- Convert to zeros - poles - gain

tf:

- Input a system as a transfer function
- Convert to transfer function form.



Example: Starting from state-space

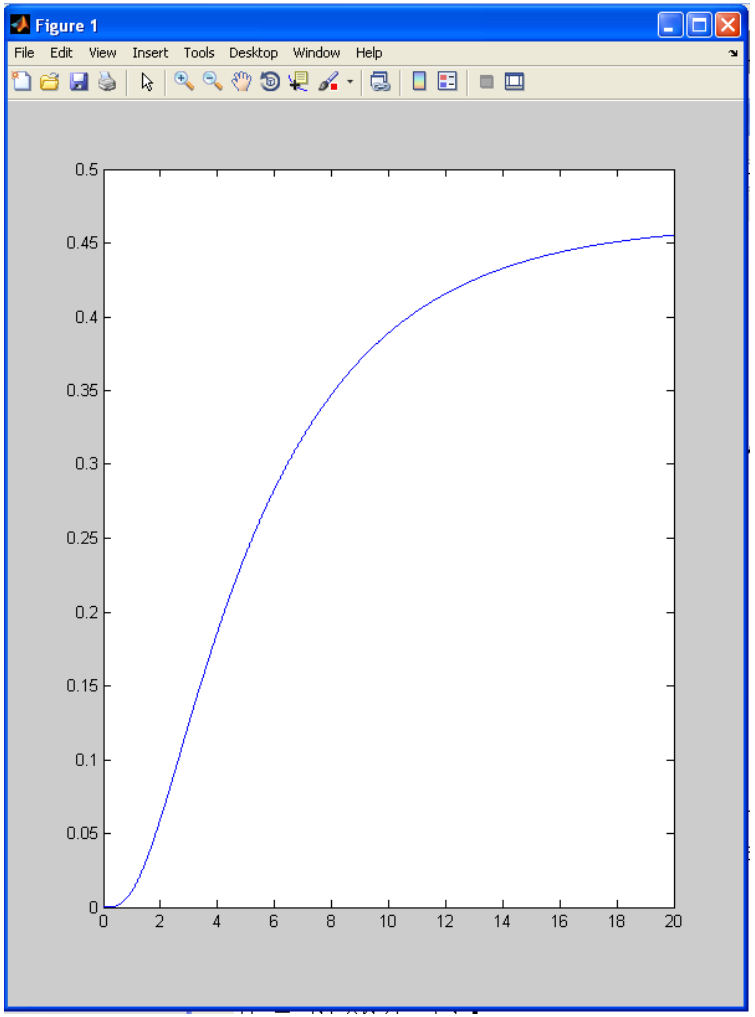
```
>> A = [-2.1,1,0,0;1,-2.1,1,0;0,1,-2.1,1;0,0,1,-1.1];
>> B = [1;0;0;0];
>> C = [0,0,0,1];
>> D = 0;
>> G = ss(A,B,C,D);
>> zpk(G)

                    1
-------------------------------------
(s+3.632) (s+2.447) (s+1.1) (s+0.2206)


>> tf(G)

                        1
-----------------------------------------
s^4 + 7.4 s^3 + 17.16 s^2 + 13.21 s + 2.157
```

evalfr(G, s): Calculate G(s) at point s.

```
>> evalfr(G,j*3)

   -0.0023 + 0.0052i
```

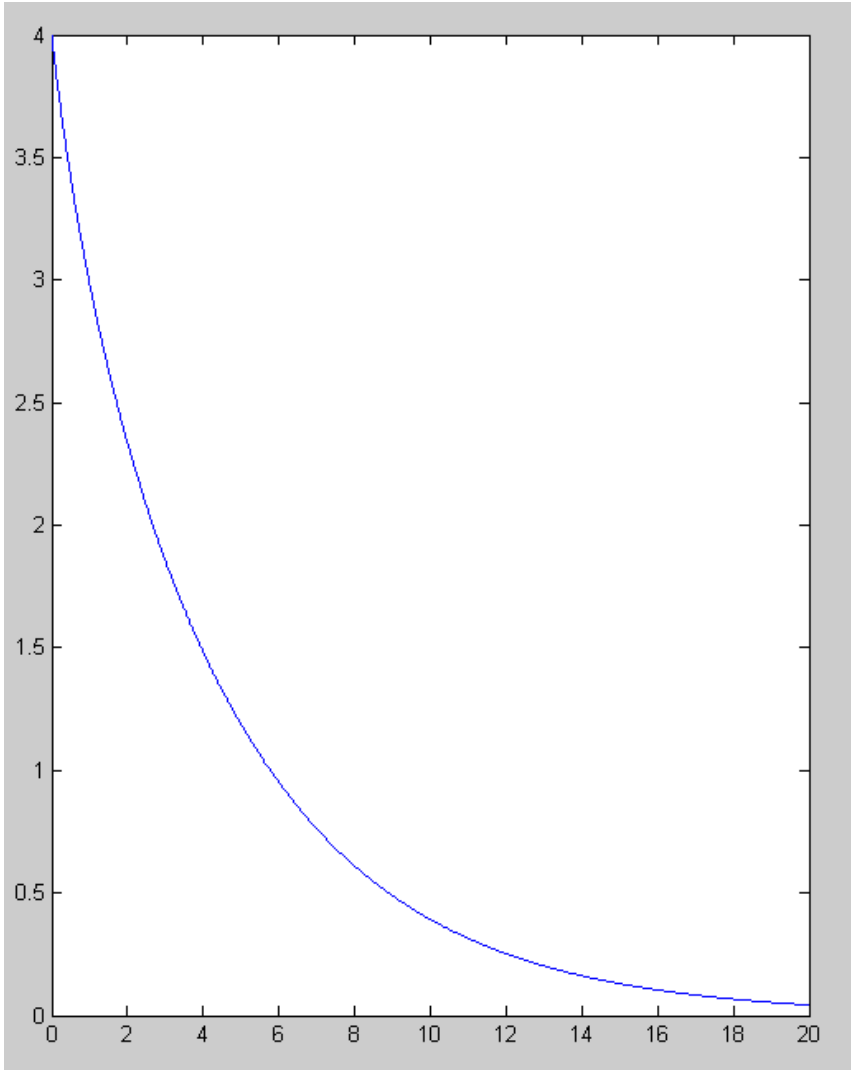
step(G,t): Response to a unit step input:

```
>> t = [0:0.01:20]';
>> y = step(G,t);
>> plot(t,y)
```



impulse(G,t): Impulse response. Also the zero-input response with initial condition X0:

```
>>
>> X0 = [1,2,3,4]';
>> G = ss(A,X0,C,D);
>> y = impulse(G,t);
>> plot(t,y);
```



eig(A): Eigenvalues of matrix A

```
>> A

   -2.1000    1.0000         0         0
    1.0000   -2.1000    1.0000         0
         0    1.0000   -2.1000    1.0000
         0         0    1.0000   -1.1000

>> eig(A)

   -3.6321
   -2.4473
   -1.1000
   -0.2206
```

poly(eig(A)): Characteristic polynomial of matrix A

```
>> poly(eig(A))

    1.0000    7.4000   17.1600   13.2140    2.1571
```

Eigenvalues and Eigenvectors

```
>> [M,N] = eig(A)

M =
   -0.4285   -0.6565    0.5774    0.2280
    0.6565    0.2280    0.5774    0.4285
   -0.5774    0.5774   -0.0000    0.5774
    0.2280   -0.4285   -0.5774    0.6565


N =

   -3.6321         0         0         0
         0   -2.4473         0         0
         0         0   -1.1000         0
         0         0         0   -0.2206
```

det(A): Determinant of matrix A.

- Also equal to the product of the eigenvalues.

```
>> det(A)

    2.1571

>> prod(eig(A))

    2.1571
```

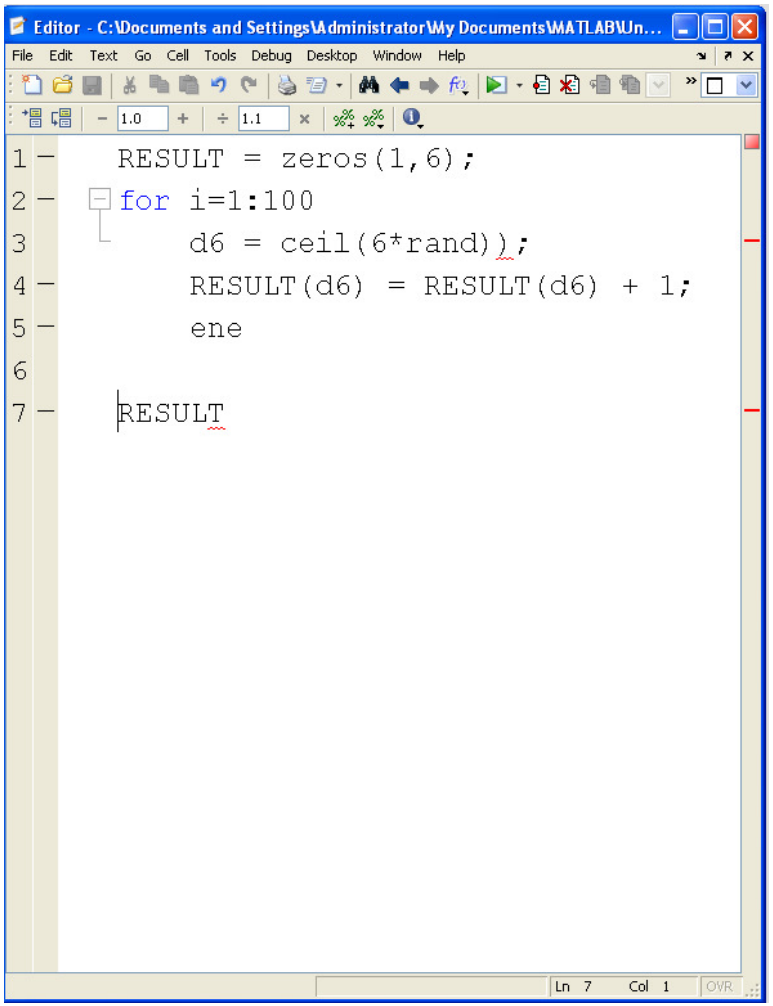
trace(A): Trace of matrix A. Also equal to the sum of the eigenvalues.

```
>> trace(A)

   -7.4000

>> sum(eig(A))

   -7.4000
```

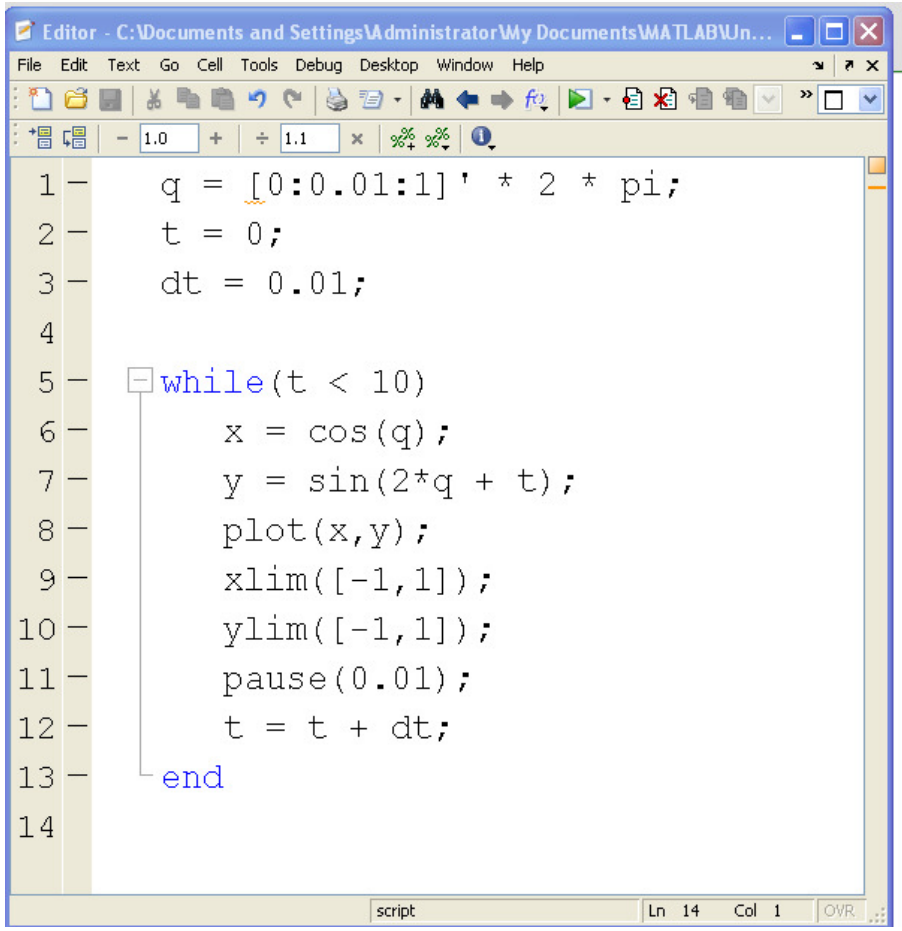
Matlab Scripts

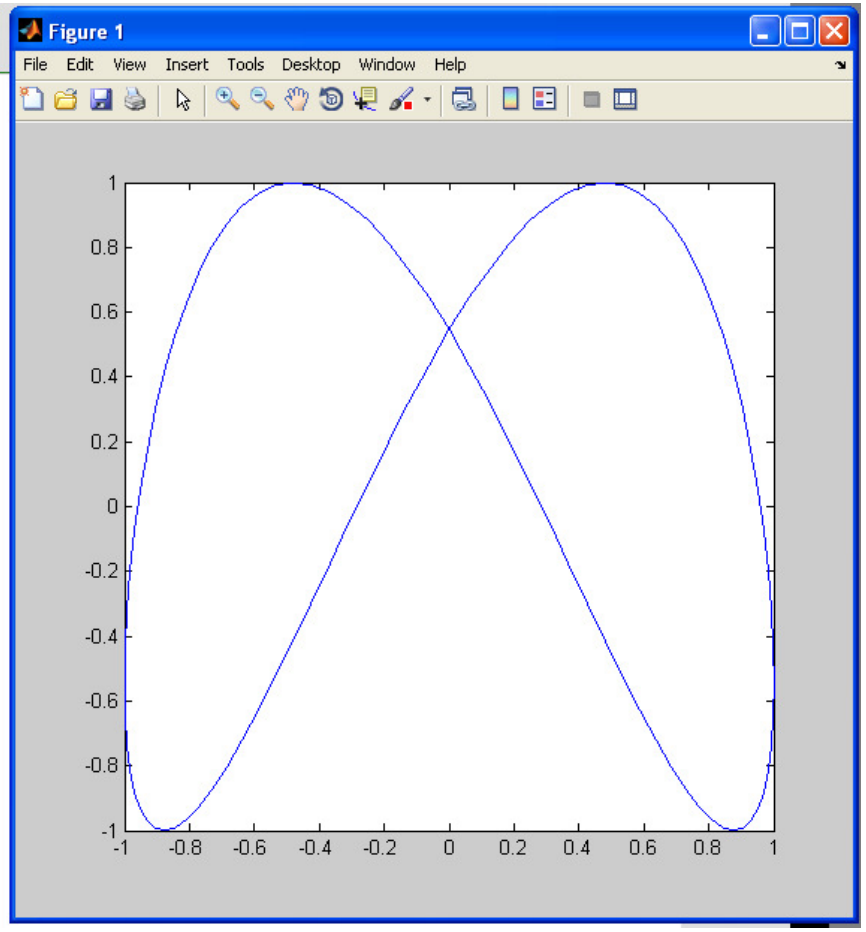
- Similar to the command window
- Allows you to execute a set of code
- Convenient when making small changes
- Good way to build and debug your code

Execute (green arrow) is the same as copying the code into the command window



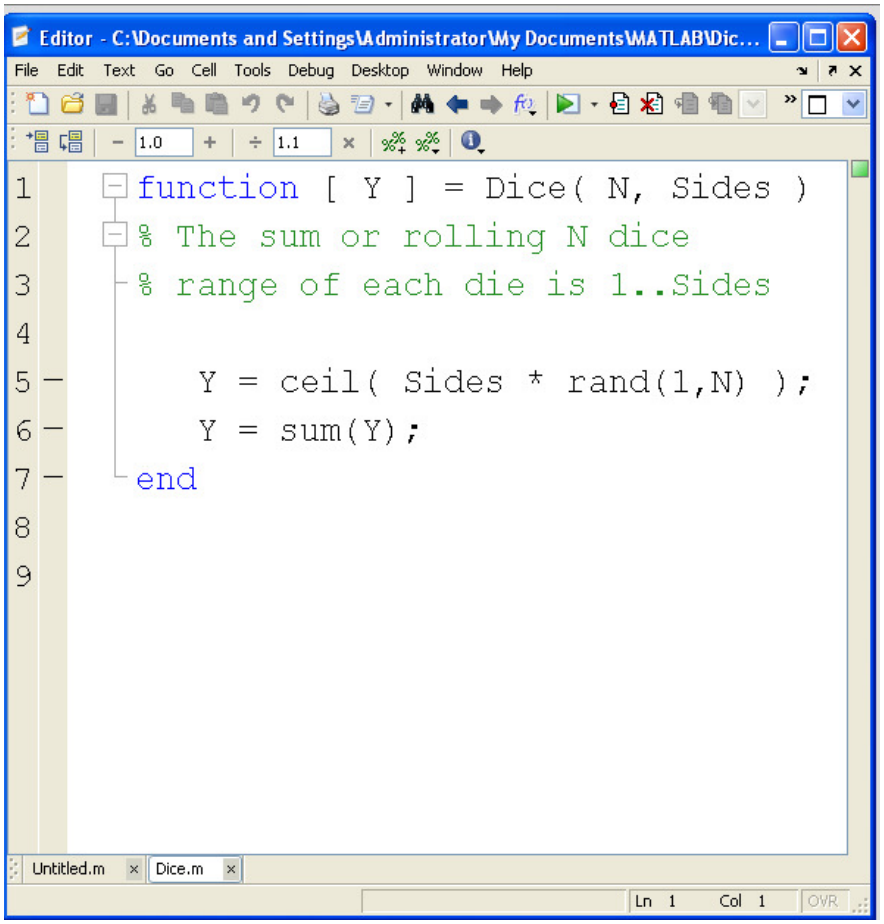
Script Example: Lissajou Figure





Matlab Functions

- Similar to a subroutine
- Function name *must* match file name
- *Must* be somewhere Matlab can find the file



Once created, you can call this function from the command window

Example: Find the sum of 20 six-sided dice (level 20 fireball)

```
>> Dice(20,6)

ans =     83
```

Function Example

Compute the acceleration of a ball rolling in a bin.

- X = [position, velocity]
- dX = [velocity, acceleration]

```
function [dX] = BallDynamics(X)
% [] = BallDynamics(X)
%
% JSG - 8/5/15
 x = X(1);
 dx = X(2);
 g = -9.8;

 dX = zeros(2,1);
 dX(1) = dx;
 dX(2) = (5.6*dx*dx + 2*g)*x / ( 1.4*(1 + 4*x*x));

end
```

Analysis

- sqrt(x) square root of x
- log(x) log base e
- log10(x) log base 10
- exp(x) e^x
- exp10(x) 10^x
- abs(x) |x|
- round(x) round to the nearest integer
- floor(x) round down (integer value of x)
- ceil(x) round up to the next integer
- real(x) real part of a complex number
- imag(x) imaginary part of a complex number
- abs(x) absolute value of x, magnitude of a complex number
- angle(x) angle of a complex number (answer in radians)
- unwrap(x) remove the discontinuity at pi (180 degrees) for a vector of angles

Polynomials

- poly(x)
- roots(x)
- conv(x,y)

Trig Functions

• sin(x)	sin(x) where x is in radians
• cos(x)	cos()
• tan(x)	tan()
• asin(x)	arcsin(x)
• acos(x)	arccos(x)
• atan(x)	arctan(x)
• atan2(y,x)	angle to a point (x,y)

Probability and Statistics

- factorial(x) (x-1)!
- gamma(x) x!
- rand(n,m) create an nxm matrix of random numbers between 0 and 1
- randn(n,m) create an nxm matrix of random numbers with a normal distribution
- sum(x) sum the columns of x
- prod(x) multiply the columns of x
- sort(x) sort the columns of x from smallest to largest
- length(x) return the dimensions of x
- mean(x) mean (average) of the columns of x
- std() standard deviation of the columns of x

Display Functions

- plot(x) plot x vs sample number
- plot(x,y) plot x vs. y
- xlim([0,10]) limit the x axis from 0 and 10
- ylim([-2,2]) limit the y axis from -2 to +2
- semilogx(x,y) log(x) vs y
- semilogy(x,y) x vs log(y)
- loglog(x,y) log(x) vs log(y)
- mesh(x) 3d plot where the height is the value at x(a,b)
- contour(x) contour plot
- bar(x,y) draw a bar graph
- xlabel('time') label the x axis with the word 'time'
- ylabel() label the y axis
- title() put a title on the plot
- grid() draw the grid lines

Useful Commands

- hold on — don't erase the current graph
- hold off — do erase the current graph
- diary — create a text file to save whatever goes to the screen
- linepace(a, b, n) — create a 1xn array starting at a, increment by b
- logspace(a,b,n) — create a 1xn from 10^a to 10^b spaced logarithmically
- subplot() — create several plots on the same screen
- disp('hello') — display the message *hello*

Utilities

• format	set the display format
• zeros(n,m)	create an nxm matrix of zeros
• eye(n,m)	create an nxm matrix with ones on the diagonal
• ones(n,m)	create an nxm matrix of ones
• help	help using different functions
• pause(x)	pause x seconds (can be a fraction). Show the graph as well
• clock	the present time
• etime	the difference between to times
• tic	start a stopwatch
• toc	the number of seconds since tic

System Analysis

- G = tf([2,3],[1,4,5,6]); Input a system G(s) in transfer function form
- G = zpk([-1],[-2,-3],10); Input a system by its zeros, poles, and gain
- G = ss(A,B,C,D); Input a system in state-space form
- eig(G) Eigenvalues of system G
- eig(A) Eigenvalues of matrix A
- poly([-1,-2,-3]) Find a polynomial with roots at {-1, -2, -3}
- roots([1,2,3,4]) Find the roots to s4 + 2s2 + 3s + 4 = 0
- evalfr(G,-2+j*3) Evaluate G(s) at s = -2 + j3
- y = step(G,t); Compute the step response of G(s)
- Kx = lrq(A, B, Q, R) LQR method for finding feedback gains