```
% Scott Wilcox
% ME 564 Matlab Introduction

clear all
close all
clc
```

Basic Operations

-- similar to many basic programming languages (C, etc.)

```
2+2
```

```
ans = 4
```

```
2*2
```

```
ans = 4
```

```
2/2
```

```
ans = 1
```

```
2-2*7   % Order of operations matters
```

```
ans = -12
```

```
(2-2)*7
```

```
ans = 0
```

```

% Assign variables using '='
a=2+2;
```

Loops

```
count=0; % Initialize the variable "count"
% Add all the numbers 1-10
for ii=1:10
    count=ii+count;
end

counter=count; % Assign count to counter

% Subtract numbers until count is not greater than 10
while counter >10
    counter=counter-1;
end
```

Create a vector

```
A_Vector=[ 1 2 3 4] % Brackets [] create arrays for data
```

```
A_Vector = 1×4
     1     2     3     4
```

```
B_Vector=ones(1,4)  % Use matlab commands to make vectors
```

```
B_Vector = 1×4
     1     1     1     1
```

```
% Also available are zeros, randn, eye
```

Create a Matrix

```
A_Matrix=[1 2 3 4
    5 6 7 8]
```

```
A_Matrix = 2×4
     1     2     3     4
     5     6     7     8
```

```
% Matrices are like stacks of vectors so [] zeros, ones, eye.. all work to
% create matrices as well.
```

Operations on Vectors/Matrices

```
A_Matrix*A_Vector' % matrices and vectors have to have one matching
```

```
ans = 2×1
    30
    70
```

```
% dimension to be multiplied together
% The ' operator takes the transpose (NxM to MxN)

A_Matrix.^2        % The "." takes the following operator ( in this case
```

```
ans = 2×4
     1     4     9    16
    25    36    49    64
```

```
% the ^) and performs the operation on all the terms
% This is the "element-wise" operation -- All elements
% of A_Matrix are now squared.
```

Create a function in Matlab

Time, and fuctions of time f(t) are represented by vectors

```
the_time=-100:1:100; % The ":" operator takes elements from -100 to 100 in
% steps of 1 (-100, -99,-98..., 0, ..., 99, 100)
```

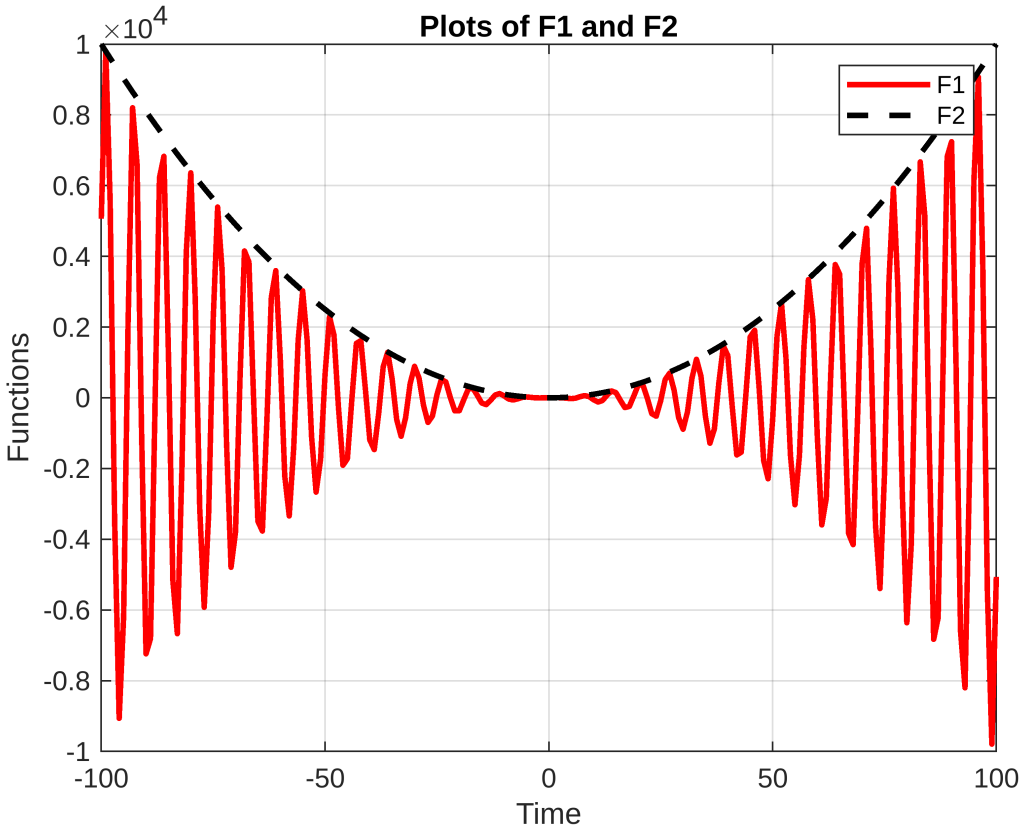
```
F1=(the_time.^2).*sin(the_time);  % t^2*sin(t)
F2=(the_time.^2);                 % t^2

% Plotting

figure(40)
plot(the_time,F1,'r-','Linewidth',2)  % Plots F1(t) in a red solid line

hold on % Holds everything on the current plot to add more later

plot(the_time,F2,'k--','Linewidth',2) % Plots F2(t) in a black dashed line
xlabel('Time')                  % Adds the label "time" to the x-axis
ylabel('Functions')             % Adds the label "functions" to the y-axis
legend('F1','F2')               % Creates a legend with entries "F1", "F2"
grid on                         % Turns on the grid lines
title('Plots of F1 and F2')     % Gives the plot a title
```



Linear System Solving (2 equations + 2 unknowns)

```
% z  + 2y = 1
% 4z + 5y = 1

A=[ 1 2; 4 5]
```

```
A = 2×2
     1     2
     4     5
```

```
b=[ 1; 1]
```

```
b = 2×1
     1
     1
```

```
x=A\b  % The "\" command will solve Ax=b for x given A and b
```

```
x = 2×1
    -1
     1
```

```
% Even for very large system

AA=randn(100,100);
bb=randn(100,1);

xx=AA\bb
```

```
xx = 100×1
    0.4980
    0.3609
    0.0032
    0.7538
   -1.1818
   -0.1171
    0.1231
    0.6710
    0.8301
   -0.1907
     ⋮
```

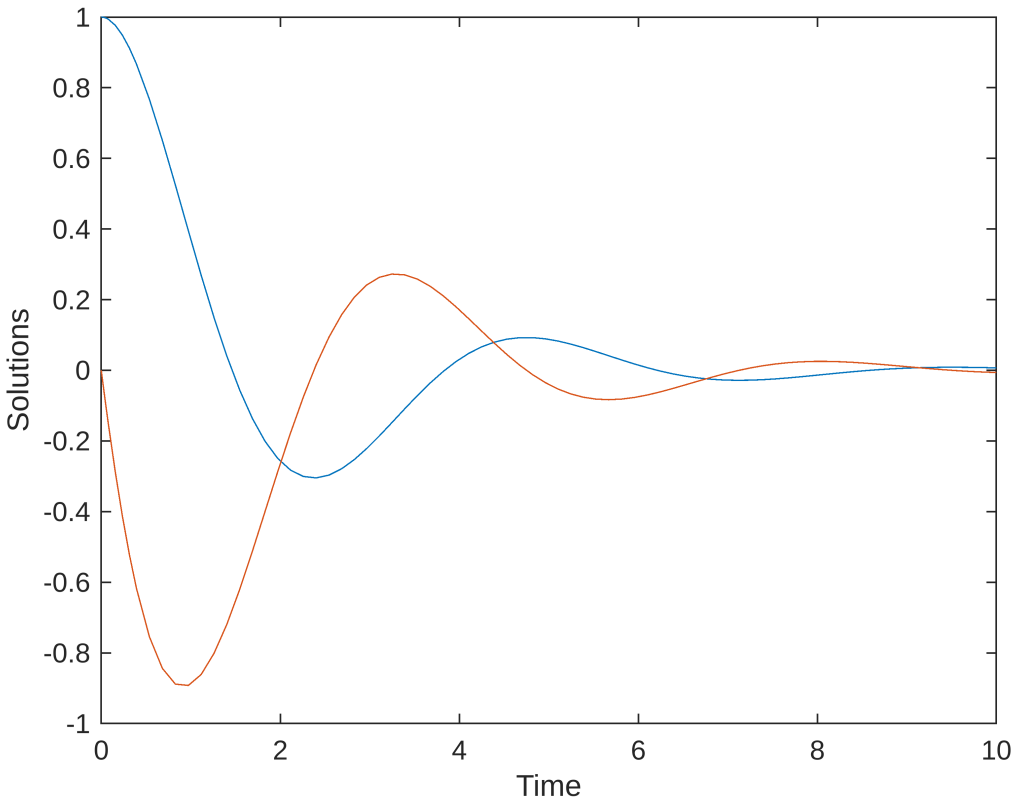
Ordinary Differential Equations (ODE)

```
% help ode45

TSpan=[0 10];   % Initial Time to Final Time
X_initial=[1;0];% Initial Conditions

[T_out,Y_out]=ode45('ode_test', TSpan,X_initial);
% Calles the function ode_test at each time step and solves for new state
% given initial state then repeats at every time step.

figure()
plot(T_out,Y_out)
xlabel('Time')
ylabel('Solutions')
```




```
legend('Position','Velocity')
```

```
Warning: 'Position' interpreted as a legend property name.  To include a label with the same name
as a legend property, specify the labels using a cell array or string array.
Error using legend
Error setting property 'Position' of class 'Legend':
Value must be numeric and finite
```

```
title('Position and Velocity')

figure()
plot(T_out,Y_out(:,1))  % The ":" here takes all elements in the rows
% associated with the first column
xlabel('Time')
ylabel('Solutions')
legend('Position')
title('Just Position')
```

Animate your plots (if time)

```
figure()
% figure('Renderer','zbuffer');
plot(T_out,Y_out(:,1));
axis([0 10 -1.5 1.5]);
set(gca,'NextPlot','replaceChildren');
for j = 1:length(T_out)
    plot(T_out(1:j),Y_out(1:j,1))
```

```
    F(j) = getframe;
end
movie(F,1)
```