Table of Contents


```
% prepare to run...
clear all  % clear out all variables
close all  % close any open figures
```

Problem 1

```
type('FlipEvensInString')
```

```
function stringOut = FlipEvensInString(stringIn)
%{
Alyssa Rose   2-3-18

Takes in string; characters in even numbered indices are
flipped back to front and odd index characters are
kept in normal order
%}

%extracts the characters in even numbered indices
flipArray = stringIn(2:2:end);
%stores even characters in new array, flipped here
flipArray(1:1:end) = flipArray(end:-1:1);
%flipped evens put back into original array
stringIn(2:2:end) = flipArray(1:1:end);
stringOut = stringIn;
end
```

Problem 2

```
type('ClassifyStorm')
```

```
function stormType = ClassifyStorm(windSpeedMph)
%{
Alyssa Rose   2-3-18

Function returns type of storm based on wind speed
%}

%rounds to eliminate decimals in wind speed input
```

```
newWind = round(windSpeedMph);
if (newWind <= 38)
    stormType ='tropical depression';
elseif (newWind>=39) && (newWind<74)
    stormType = 'tropical storm';
else
    stormType='hurricane';
end
```

Problem 3

```
type('StringToDecision')
```

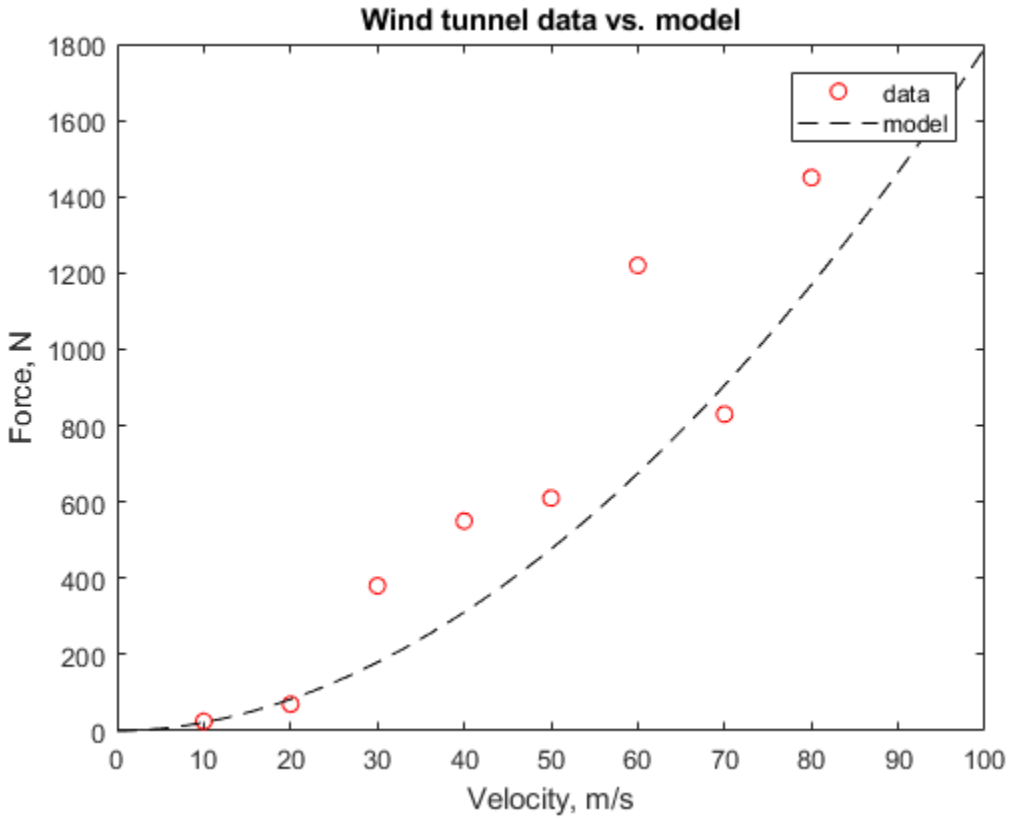
```
function decision = StringToDecision(userString)
%{
Alyssa Rose       2-3-18
 Takes in string to determine decision of user:
 all forms of yes = 1
 all forms of no = 0
 other strings = -1
 non strings = -2
%}

%converts input to lower case, and retrieves first
%letter to check for ischar
stringLower= lower(userString);
stringAnswer = stringLower(1);

decision = 0;
if strcmp(stringLower, 'yes') || strcmp(stringLower, 'y')
    decision = 1;
elseif strcmp(stringLower, 'no') || strcmp(stringLower,'n')
    decision = 0;
elseif ischar(stringAnswer) ~= 1
    decision = -2;
else
    decision = -1;
end
```

Problem 4

```
type('WindTunnelData')
WindTunnelData
```

```
%{
 Alyssa Rose       2-3-18
```

```
 Computes the predicted force for values of
 velocity ranging from 0 to 100 m/s using equation
 to compare with known data
%}

%data fromm given data points
F = [25, 70, 380, 550, 610, 1220, 830, 1450];
v = [10, 20, 30, 40, 50, 60, 70, 80];

%data for predicted force values
velocity = linspace (0, 100, 100);
forcePredicted = 0.271*velocity.^(1.91);

%plots "data" with red circles and "model" with black dashed line
plot(v,F,'ro', velocity, forcePredicted,'black--')
xlabel('Velocity, m/s')
ylabel('Force, N')
title('Wind tunnel data vs. model')
legend('data', 'model')
```

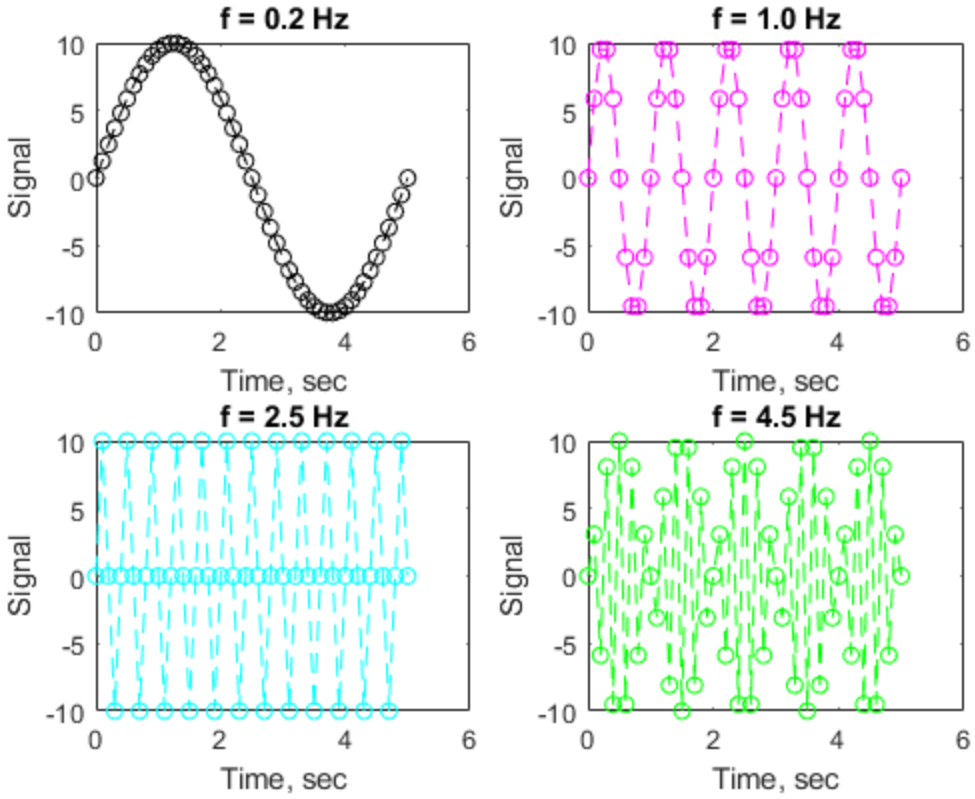


Problem 5

```
type('SampledSine')
```

```
type('SimulateADC')
disp('----test SimulateADC----')
figure  % adding this to ensure we don't overwrite Problem 4 plot
SimulateADC


function y =SampledSine(amp, freq, dT, numSamples)
%{
Alyssa Rose      2-3-18

Takes in amplitude, frequency, "step size" of time,
and the number of samples. Negative dT and numSamples
not allowed. A sine wave of the signal is returned.
%}

%checks for allowed dT/numSamples values
if dT < 0
    y = [];
elseif numSamples < 0
    y = [];
else
%uses numSamples to find last value in the time vector
finalStep = (numSamples - 1)*dT;
%creates time vector starting from 0 with dT intervals
t = 0:dT:finalStep;
y = amp.*sin(2.*pi.*freq.*t);
end




%{
Alyssa Rose     2-3-18

Creates 4 subplots of sine waves dependent on freq.
using SampledSine function
%}

amplitude = 10;
time = 0:0.1:5;
dT=0.1;
sampl = (5 ./dT) +1;
y1 = SampledSine(amplitude, 0.2, dT, sampl);
y2 = SampledSine(amplitude, 1.0, dT, sampl);
y3 = SampledSine(amplitude, 2.5, dT, sampl);
y4 = SampledSine(amplitude, 4.5, dT, sampl);
%plot 1: freq @ 0.2 Hz
subplot(2,2,1)
plot(time,y1,'black--o')
title('f = 0.2 Hz')
xlabel('Time, sec')
ylabel('Signal')
```

```
%plot 2: freq @ 1.0 Hz
subplot(2,2,2)
plot(time,y2,'magenta--o')
title('f = 1.0 Hz')
xlabel('Time, sec')
ylabel('Signal')

%plot 3: freq @ 2.5 Hz
subplot(2,2,3)
plot(time,y3,'cyan--o')
title('f = 2.5 Hz')
xlabel('Time, sec')
ylabel('Signal')

%plot 4: freq @ 4.5 Hz
subplot(2,2,4)
plot(time,y4,'green--o')
title('f = 4.5 Hz')
xlabel('Time, sec')
ylabel('Signal')

%{
in each plot, the period of the wave gets smaller.
In the first graph the period takes 5 sec to complete
which is equivalent to 1/0.2. As the frequency increases
the period of the wave will continue to decrease. Notably
at 2.5 Hz, the waves appear as though they would cancel
each other out (constructive/destructive waves)
%}
----test SimulateADC----
```



Published with MATLAB® R2017b