% VoltageSolver.m
% Patrick Utz, 4/23/18, 13.1

% Description: Analyzing electric circuits can be accomplished by
% solving sets of equations. For a particular circuit, the voltages V1,
% V2, and V3 are found through the following linear equations:

$V_1 = 5$
$-6V_1 + 10V_2 - 3V_3 = 0$
$-V_2 + 51V_3 = 0$
Write a MATLAB program to solve for the voltages. Show your program and results.

% Variables: A = the constants of V1, V2, V3 respectively; B = the
% answers to each equation; V1, V2, V3 = the answers to the system

```
clear
A = [1 0 0; -6 10 -3; 0 -1 51];
B = [5;0;0];
answer = A\B;
V1 = answer(1)
V2 = answer(2)
V3 = answer(3)
```

```
V1 =

   5.0000


V2 =

   3.0178


V3 =

   0.0592
```

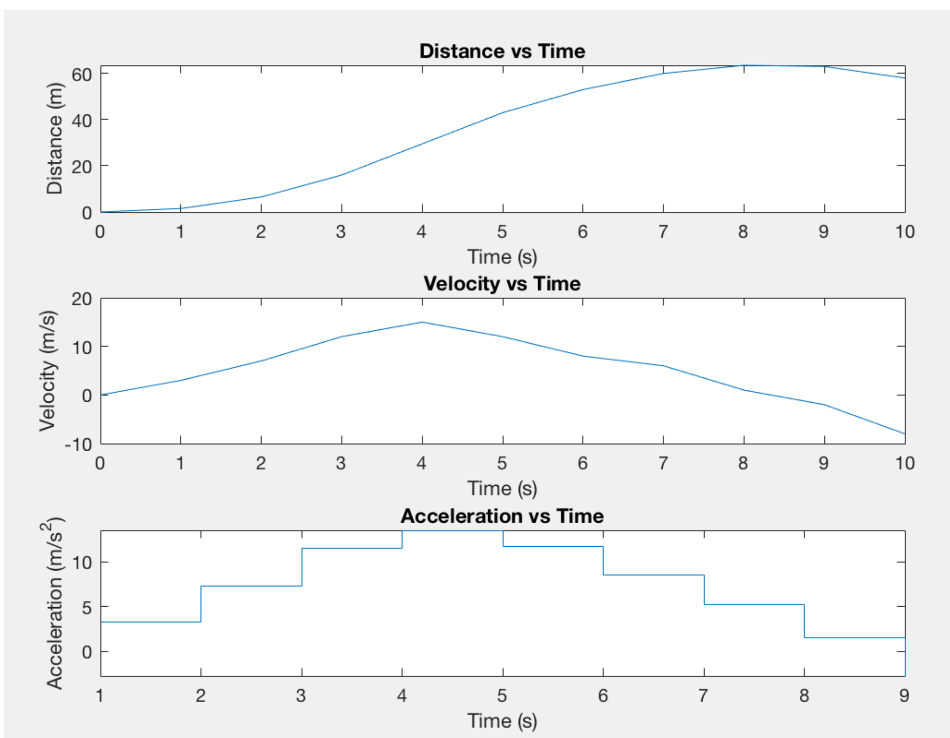
```
% VoltageSolverWithSym.m
% Patrick Utz, 4/23/18, 13.2

% Description: An electrical circuits with time-varying sources is being
% analyzed to follow the following equations. Write a MATLAB program to use
% the solve function to solve for the three voltages, Va, Vb, and Vc, each
% is a function of time (t). Show your program and results.
```

$2(Va-Vb) + 5(Va-Vc) - e^{-t} = 0$
$2(Vb - Va) + 2Vb + 3(Vb - Vc) = 0$
$Vc = 2 \sin(t)$

```
% Variables: Va, Vb, Vc = the voltages; eqX = the equations

clear
syms t Va Vb Vc;
eq1 = 2*(Va - Vb) + 5*(Va - Vc) - exp(-t) == 0;
eq2 = 2*(Vb - Va) + 2*Vb + 3*(Vb - Vc) == 0;
eq3 = Vc == 2*sin(t);
s = solve(eq1,eq2,eq3,Va,Vb,Vc);
disp('Using symbolic solve function');
disp(s.Va);
disp(s.Vb);
disp(s.Vc);
```

```
>> VoltageSolverWithSym
Using symbolic solve function
(7*exp(-t))/45 + (82*sin(t))/45

(2*exp(-t))/45 + (62*sin(t))/45

2*sin(t)
```

```
% MovingObject.m
% Patrick Utz, 4/23/18, 13.3

% Description: An object starts to move at time 0, and its velocity is
% measured at a sequence of time as below. Write and run a MATLAB program
% that determine and plot its distance from its initial position at t=0
% over this time period, the velocity versus time, and the acceleration
% versus time using subplot(3, 1, n). Use the time interval from 0 to 10s
% for all the plots. Use the central difference technique to compute the
% acceleration and use the stairs for the acceleration versus time
% curve. Show your program and curves.
% Time (s) 0 1 2 3 4 5 6 7 8 9 10
% Vel (m/s) 0 3 7 12 15 12 8 6 1 -2 -8

% Variables: all names are self-explanatory

clear
time = 0:10;
velocity = [0 3 7 12 15 12 8 6 1 -2 -8];
distance = cumtrapz(time,velocity);
dx = diff(time);
dy = diff(distance);
acceleration = (dy(1:end-1)+dy(2:end))./(dx(1:end-1)+dx(2:end));

subplot(3,1,1)
plot(time, distance);
title('Distance vs Time');
xlabel('Time (s)');
ylabel('Distance (m)')

subplot(3,1,2)
plot(time, velocity);
title('Velocity vs Time');
xlabel('Time (s)');
ylabel('Velocity (m/s)');

subplot(3,1,3)
stairs(acceleration);
title('Acceleration vs Time');
xlabel('Time (s)');
ylabel('Acceleration (m/s^2)');
```



```
% Mathing_A_Function.m
% Patrick Utz, 4/23/18, 13.4

% Description: Write one program to find the following for the function
%           4x2 + 3.
% (1) Find the indefinite integral of the function using the symbolic
%    int function.
% (2) Find the definite integral of this function from x=-1 to x=3 using
%    the symbolic int function.
% (3) Find the definite integral of this function from x = -1 to x = 3 using
%    the trapz function with 100 evenly spaced data points.
% (4) Find the definite integral of this function from x = -1 to x = 3 using
%    the integral function.
% Show your program and results. Compare the results in (3) and (4), which
% one is more accurate? Why?

% Variables: all names are self-explanatory

clear
syms x;
eq1 = 4*x^2 + 3;
eq1Integral = int(eq1);
fprintf('Indefinite Integral With Symbolic Int Function: ');
disp(eq1Integral);
eq1DefiniteIntegral = int(eq1,-1,3);
fprintf('Definite Integral With Symbolic Int Function: ');
disp(eq1DefiniteIntegral);

x2 = linspace(-1,3,100);
eq2 = 4.*(x2).^2 + 3;
eq2DefiniteIntegralTrapz = trapz(x2,eq2);
fprintf('Definite Integral With Trapz Function: ');
disp(eq2DefiniteIntegralTrapz);

fun = @(x3) 4.*(x3).^2 + 3;
eq3DefiniteIntegral = integral(fun,-1,3);
fprintf('Definite Integral With Integral Function: ');
disp(eq3DefiniteIntegral);
```

——————————————>

>> Mathing_A_Function
Indefinite Integral With Symbolic Int Function: (4*x^3)/3 + 3*x

Definite Integral With Symbolic Int Function: 148/3

Definite Integral With Trapz Function: 49.3377

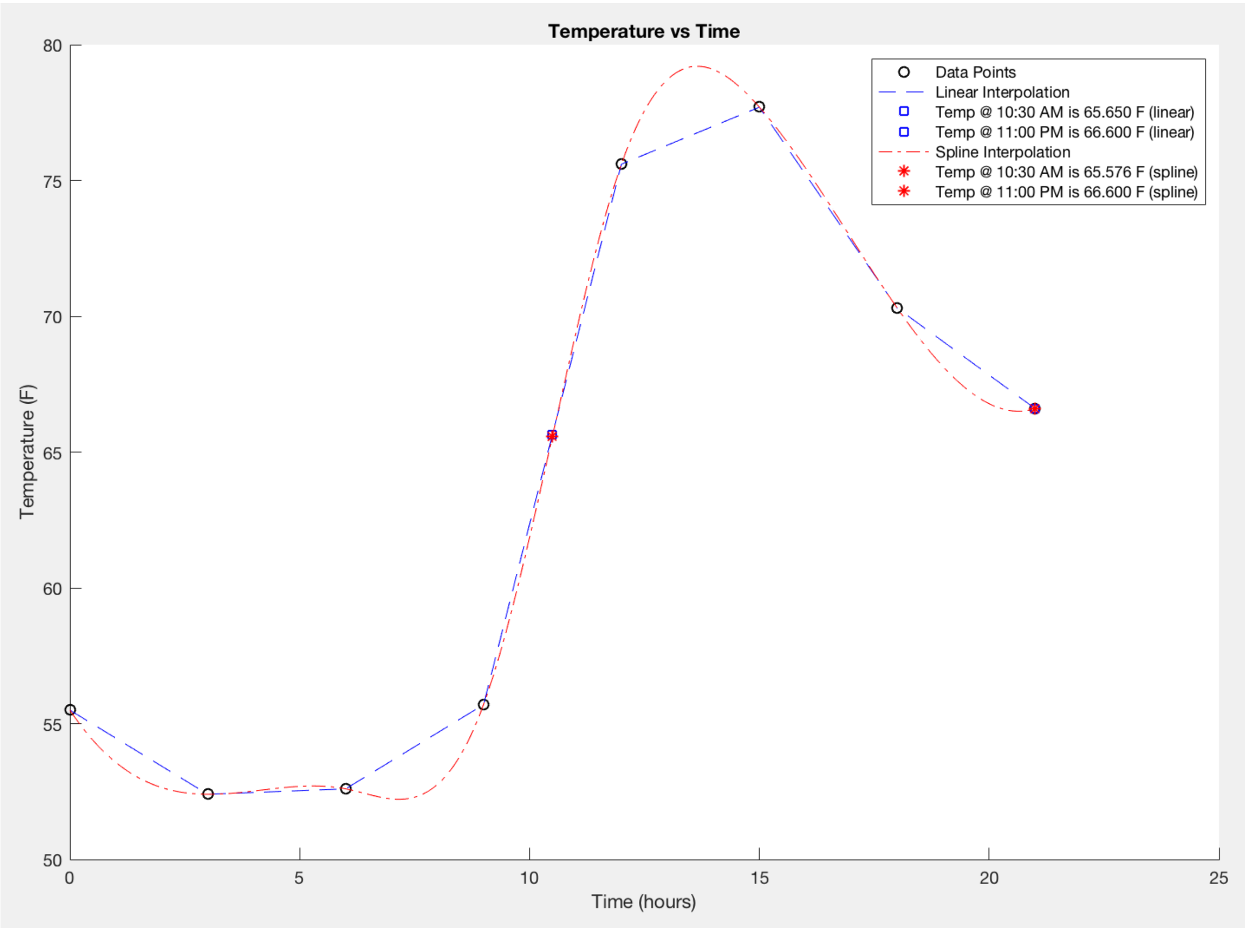
Definite Integral With Integral Function: 49.3333

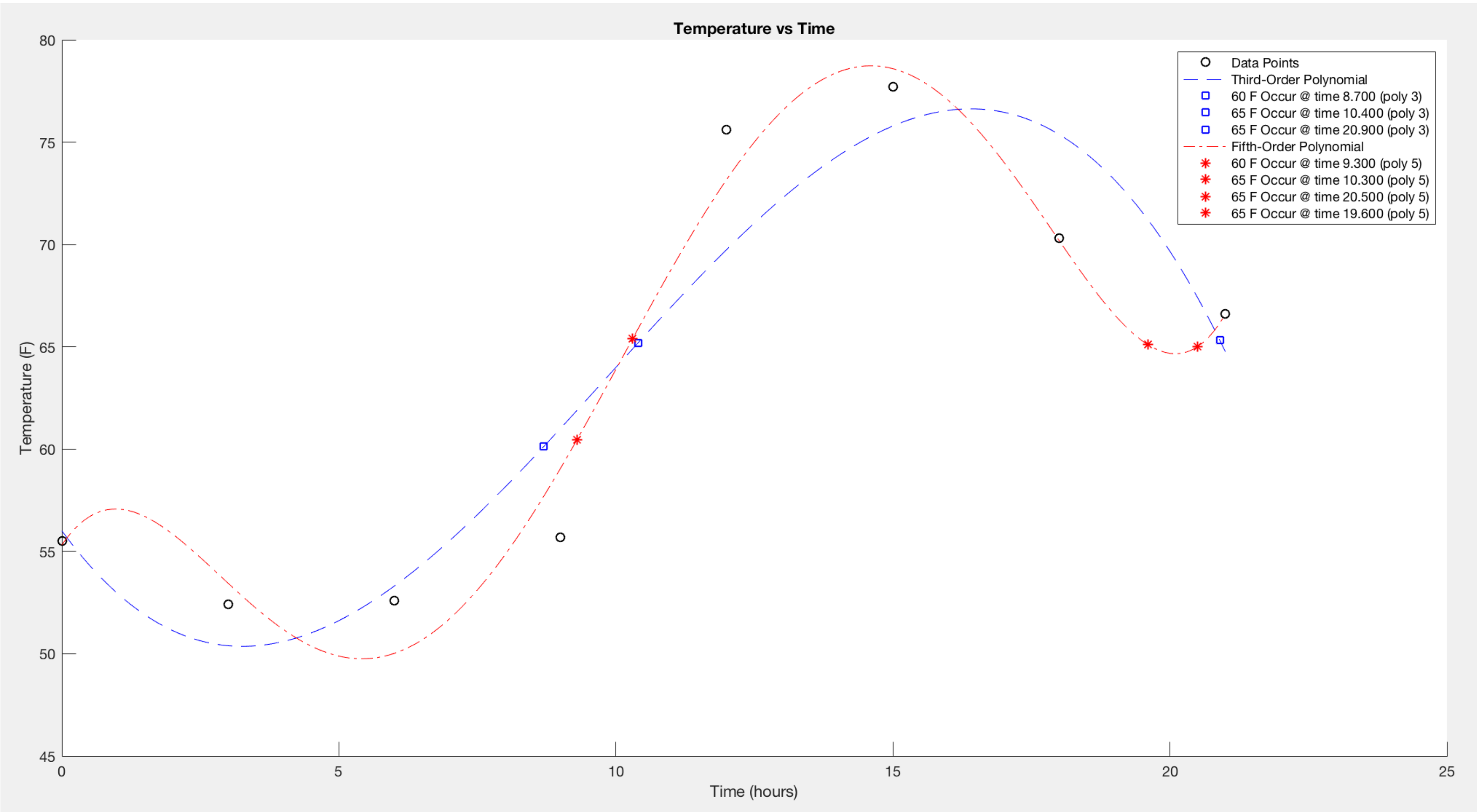
*** The last result is more accurate than the trapz function result since a greater number of sample data points is used to estimate the integral as opposed to the 100 linearly spaced data points used in the trapz function***

```
% TemperatureProblem.m
% Patrick Utz, 4/23/18, 13.5

% Description: Write a MATLAB program to solve the following problem
% The temperature (in degrees Fahrenheit) was recorded every 3 hours for
% a day at a particular location. Using a 24 hour clock where midnight is
% 0, the measured data is shown below:

% Time:0 3 6 9 12 15 18 21
% Temp: 55.5 52.4 52.6 55.7 75.6 77.7 70.3 66.6

% 1. plot the data points as black circles
% 2. plot linear inter/extrapolation of the temperature using blue dashed
%     line and spline inter/extrapolation using red dot-dashed line, in the
%     same figure window as that in 1.
% 3. find the temperature at 10:30am and 11:00pm during the day from both
%     curves in 2, and plot the points in the figure as blue squares(for
%     linear) and red stars(for spline), respectively.
% 4. in a new figure window, repeat 1, curve-fitting the data with a third-
%     order polynomial, plot the smooth fitted curve as blue dashed line,
%     and a fifth-order polynomial plotted in red dot dashed line.
% 5. find the time during the day when temperature is 60 and 65 degree,
%     from both polynomials in 4, and plot those points in the figure
%     as blue squares(for 3rd order) and red stars(5th order).
% Properly label and title your figures, use legends if necessary.

% Variables: all names are self-explanatory (for the most part)

clear
time = [0:3:21];
temp = [55.5 52.4 52.6 55.7 75.6 77.7 70.3 66.6];
time1 = 0:.1:21

hold on
plot(time,temp,'ko')
title('Temperature vs Time');
xlabel('Time (hours)');
ylabel('Temperature (F)');

temp2 = interp1(time,temp,time1,'linear')
temp2_1030AM = temp2(time1 == 10.500);
temp2_1100PM = temp2(time1 == 21.000);
plot(time1,temp2,'b--');
plot(10.500,temp2_1030AM,'bs');
plot(21.000,temp2_1100PM,'bs');
```

```
temp3 = interp1(time,temp,time1,'spline');
temp3_1030AM = temp3(time1 == 10.500);
temp3_1100PM = temp3(time1 == 21.000);
plot(time1,temp3,'r-.');
plot(10.500,temp3_1030AM,'r*');
plot(21.000,temp3_1100PM,'r*');

Lin1030 = sprintf('Temp @ 10:30 AM is %.3f F (linear)', temp2_1030AM);
Lin1100 = sprintf('Temp @ 11:00 PM is %.3f F (linear)', temp2_1100PM);
Spli1030 = sprintf('Temp @ 10:30 AM is %.3f F (spline)', temp3_1030AM);
Spli1100 = sprintf('Temp @ 11:00 PM is %.3f F (spline)', temp3_1100PM);

legend('Data Points','Linear Interpolation',Lin1030,Lin1100,...
    'Spline Interpolation',Spli1030,Spli1100);

hold off

hold on
plot(time,temp,'ko')
title('Temperature vs Time');
xlabel('Time (hours)');
ylabel('Temperature (F)');

p3 = polyfit(time,temp,3);
temp4 = polyval(p3,time1);
temp4_60_time = time1(temp4 == 60.133505194805210);
temp4_65_time1 = time1(temp4 == 65.190458425525110);
temp4_65_time2 = time1(temp4 == 65.319716001282600);
plot(time1,temp4,'b--');
plot(temp4_60_time,60.133505194805210,'bs');
plot(temp4_65_time1,65.190458425525110,'bs');
plot(temp4_65_time2,65.319716001282600,'bs');


p5 = polyfit(time,temp,5);
temp5 = polyval(p5,time1);
temp5_60_time = time1(temp5 == 60.472017920512840);
temp5_65_time1 = time1(temp5 == 65.385637097035910);
temp5_65_time2 = time1(temp5 == 65.006782524317100);
temp5_65_time3 = time1(temp5 == 65.105937685596690);
plot(time1,temp5,'r-.');
plot(temp5_60_time,60.472017920512840,'r*');
plot(temp5_65_time1,65.385637097035910,'r*');
plot(temp5_65_time2,65.006782524317100,'r*');
plot(temp5_65_time3,65.105937685596690,'r*');

Poly3_60   = sprintf('60 F Occur @ time %.3f (poly 3)', temp4_60_time);
Poly3_65_1 = sprintf('65 F Occur @ time %.3f (poly 3)', temp4_65_time1);
```

```
Poly3_65_2 = sprintf('65 F Occur @ time %.3f (poly 3)', temp4_65_time2);
Poly5_60   = sprintf('60 F Occur @ time %.3f (poly 5)', temp5_60_time);
Poly5_65_1 = sprintf('65 F Occur @ time %.3f (poly 5)', temp5_65_time1);
Poly5_65_2 = sprintf('65 F Occur @ time %.3f (poly 5)', temp5_65_time2);
Poly5_65_3 = sprintf('65 F Occur @ time %.3f (poly 5)', temp5_65_time3);

legend('Data Points','Third-Order Polynomial',Poly3_60,Poly3_65_1,...
    Poly3_65_2,'Fifth-Order Polynomial',Poly5_60,Poly5_65_1,...
    Poly5_65_2,Poly5_65_3);

hold off
```





Temperature vs Time
Data Points
Third-Order Polynomial
60 F Occur @ time 8.700 (poly 3)
65 F Occur @ time 10.400 (poly 3)
65 F Occur @ time 20.900 (poly 3)
Fifth-Order Polynomial
60 F Occur @ time 9.300 (poly 5)
65 F Occur @ time 10.300 (poly 5)
65 F Occur @ time 20.500 (poly 5)
65 F Occur @ time 19.600 (poly 5)
Temperature (F)
Time (hours)
80
75
70
65
60
55
50
45
0
5
10
15
20
25