ME C231A, EECS C220B, Problem Set #1, Modeling and Simulation

Table of Contents

Follow this format to organize the script file which "manages" all of the tasks in producing HW1. Conclude by **publishing** this file, and then printing the resulting html file to pdf. Turn the pdf file (electronically) in via bCourses (more instructions later).

Problem 1

Simulation and Linearization of Engine/Drivetrain model

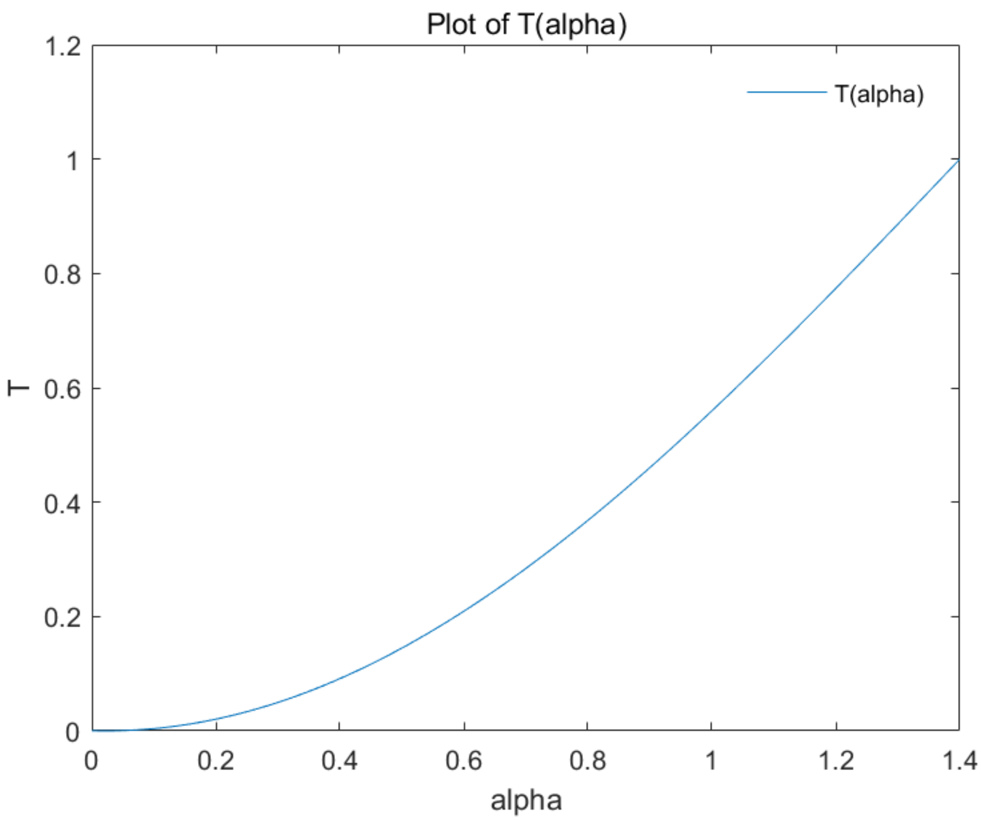
1(a)

Plot Throttle characteristic

```
fprintf('1(a) \n');
x=0:0.01:1.4;
y= HW1_T(x);
f1=figure(1);
set(f1,'name','T(alpha)-1(a)','Numbertitle','off');
hold on;
```

```
plot(x,y);
%plot(tSol,d_vSol,'b-o','LineWidth',2);
%axis([-Inf Inf -1 1]);
xlabel('alpha');
ylabel('T');
title('Plot of T(alpha)');
legend('T(alpha)');
box on;
legend('boxoff');
hold off;
```

1(a)



1(b)

Rewrite equations in state-variable form. Nothing to include in script file. If desired, can enter equations using LaTex.

```
fprintf('1(b) all good \n');
```

1(b) all good

1(c)

`hcModel` is autograded. This section can make small illustration of its behavior.

```
fprintf('1(c) all good \n');
fprintf('Use the equations from 1(b) to establish hcModel.\n');
%Use the equations from 1(b) to establish hcModel.
```

```
1(c) all good
Use the equations from 1(b) to establish hcModel.
```

1(d)

`hcEqPt` is autograded. Nothing to include in script file here. Problem 1(f) below carries out linearization at 2 specific speeds.

1(e)

Based on Throttle characteristic, and maximum value of `T`, what is maximum equilibrium speed car? Use this section to make the simple calculation.

```
fprintf('1(e) \n');
for vBar=0:0.0001:100
[maBar, wBar, alphaBar] = hcEqPt(vBar, 0.6, 0.095, 47500, 0.0026);
if((alphaBar-1.4)*(alphaBar-1.4)<0.000000001)
    break;
end
end
fprintf('Max(alphaBar)=1.4 && T(alpha) is a monotone increasing
 function. \n');
fprintf('So Max(vBar)=%d\n',maBar);
%  Max(alphaBar)=1.4 && T(alpha) is a monotone increasing function.
%  So Max(vBar)=59.883
```

```
1(e)
Max(alphaBar)=1.4 && T(alpha) is a monotone increasing function.
So Max(vBar)=1.360388e-02
```

1(f)

Illustrate the results of `hcEqPt` at `vBar=22` and `vBar=32`

```
fprintf('1(f) \n');
[maBar_v22, wBar_v22, alphaBar_v22] = hcEqPt(22, 0.6, 0.095, 47500,
 0.0026);
[maBar_v32, wBar_v32, alphaBar_v32] = hcEqPt(32, 0.6, 0.095, 47500,
 0.0026);
fprintf('v=22, ma=%d, w=%d, alpha=%d \n',maBar_v22, wBar_v22,
 alphaBar_v22);
fprintf('v=32, ma=%d, w=%d, alpha=%d \n',maBar_v32, wBar_v32,
 alphaBar_v32);
% maBar_v22=0.0027,wBar_v22=170.5426,alphaBar_22=0.3618;
% maBar_v32=0.0047,wBar_v22=248.0620,alphaBar_32=0.5632;
```

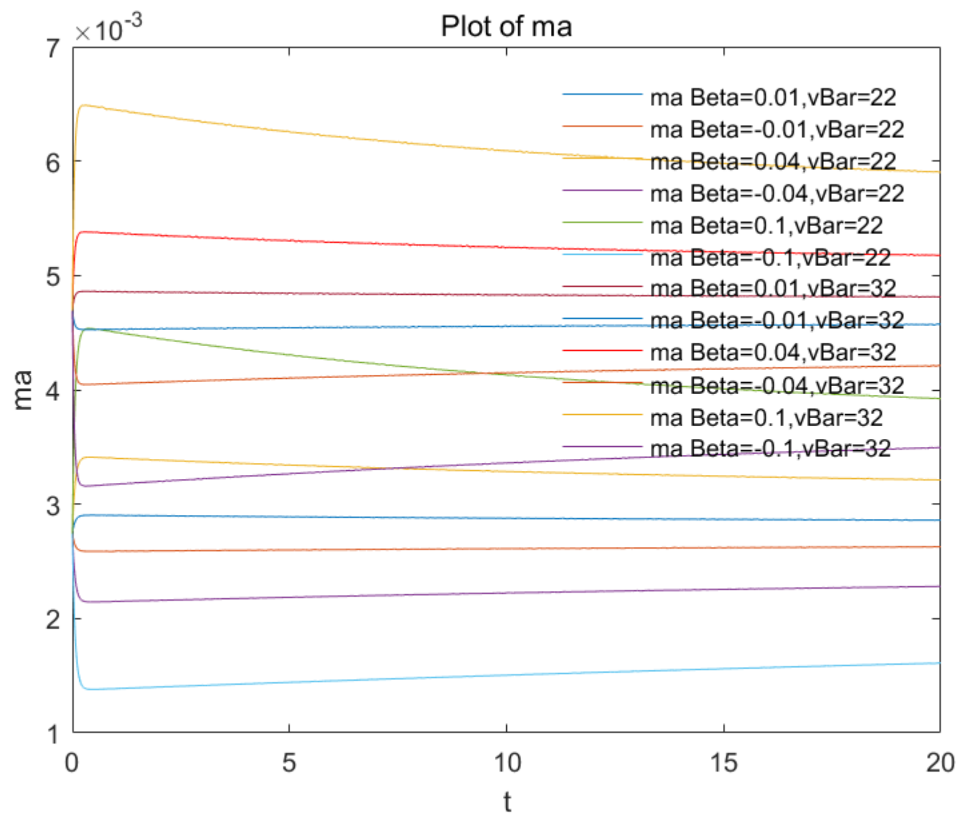
```
1(f)
v=22, ma=2.743115e-03, w=1.705426e+02, alpha=3.618500e-01
v=32, ma=4.692315e-03, w=2.480620e+02, alpha=5.632053e-01
```

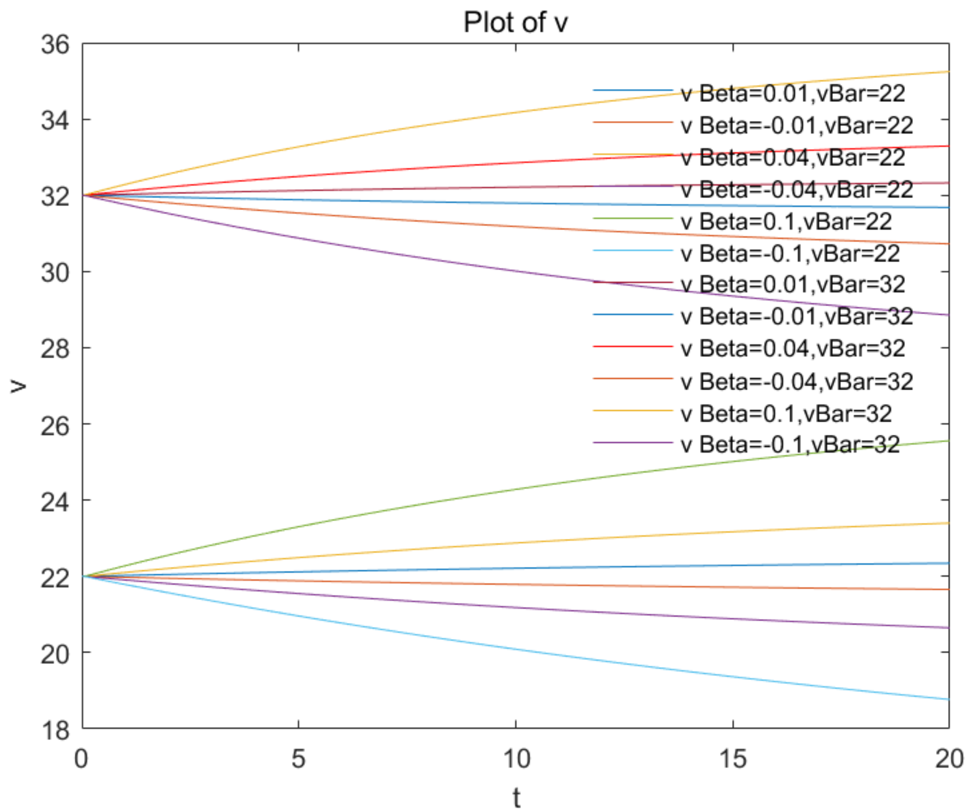
1(g)

`constantThrottleSim` is autograded. In this section, carry out the simulations, and format the plots nicely.

```
c1=0.6;
c2=0.095;
c3=47500;
c4=0.0026;
TFinal=20;
w=0.5;
[tSol1, maSol1, vSol1] = constantThrottleSim(22, 0.01,c1, c2, c3, c4,
 TFinal);
[tSol2, maSol2, vSol2] = constantThrottleSim(22, -0.01,c1, c2, c3, c4,
 TFinal);
[tSol3, maSol3, vSol3] = constantThrottleSim(22, 0.04,c1, c2, c3, c4,
 TFinal);
[tSol4, maSol4, vSol4] = constantThrottleSim(22, -0.04,c1, c2, c3, c4,
 TFinal);
[tSol5, maSol5, vSol5] = constantThrottleSim(22, 0.1,c1, c2, c3, c4,
 TFinal);
[tSol6, maSol6, vSol6] = constantThrottleSim(22, -0.1,c1, c2, c3, c4,
 TFinal);
[tSol7, maSol7, vSol7] = constantThrottleSim(32, 0.01,c1, c2, c3, c4,
 TFinal);
[tSol8, maSol8, vSol8] = constantThrottleSim(32, -0.01,c1, c2, c3, c4,
 TFinal);
[tSol9, maSol9, vSol9] = constantThrottleSim(32, 0.04,c1, c2, c3, c4,
 TFinal);
[tSol10, maSol10, vSol10] = constantThrottleSim(32, -0.04,c1, c2, c3,
 c4, TFinal);
[tSol11, maSol11, vSol11] = constantThrottleSim(32, 0.1,c1, c2, c3,
 c4, TFinal);
[tSol12, maSol12, vSol12] = constantThrottleSim(32, -0.1,c1, c2, c3,
 c4, TFinal);
f2=figure(2);
set(f2,'name','constantThrottleSim_ma 1(g)','Numbertitle','off');
hold on;
plot(tSol1,maSol1,tSol2,maSol2,tSol3,maSol3,tSol4,maSol4,tSol5,maSol5,tSol6,maSol6
%plot(tSol,d_vSol,'b-o','LineWidth',2);
%axis([-Inf Inf -1 1]);
xlabel('t');
ylabel('ma');
title('Plot of ma');
legend('ma Beta=0.01,vBar=22','ma
 Beta=0.04,vBar=22','ma Beta=-0.04,vBar=22','ma Beta=0.1,vBar=22','ma
 Beta=-0.1,vBar=22','ma Beta=0.01,vBar=32','ma Beta=-0.01,vBar=32','ma
 Beta=0.04,vBar=32','ma Beta=-0.04,vBar=32','ma Beta=0.1,vBar=32','ma
 Beta=-0.1,vBar=32');
box on;
legend('boxoff');
hold off;
```

```
figure;
f3=figure(3);
set(f3,'name','constantThrottleSim_v 1(g)','Numbertitle','off');
hold on;
plot(tSol1,vSol1,tSol2,vSol2,tSol3,vSol3,tSol4,vSol4,tSol5,vSol5,tSol6,vSol6,tSol7
%plot(tSol,d_vSol,'b-o','LineWidth',2);
%axis([-Inf Inf -1 1]);
xlabel('t');
ylabel('v');
title('Plot of v');
legend('v Beta=0.01,vBar=22','v Beta=-0.01,vBar=22','v
 Beta=0.04,vBar=22','v Beta=-0.04,vBar=22','v Beta=0.1,vBar=22','v
 Beta=-0.1,vBar=22','v Beta=0.01,vBar=32','v Beta=-0.01,vBar=32','v
 Beta=0.04,vBar=32','v Beta=-0.04,vBar=32','v Beta=0.1,vBar=32','v
 Beta=-0.1,vBar=32');
box on;
legend('boxoff');
hold off;
```



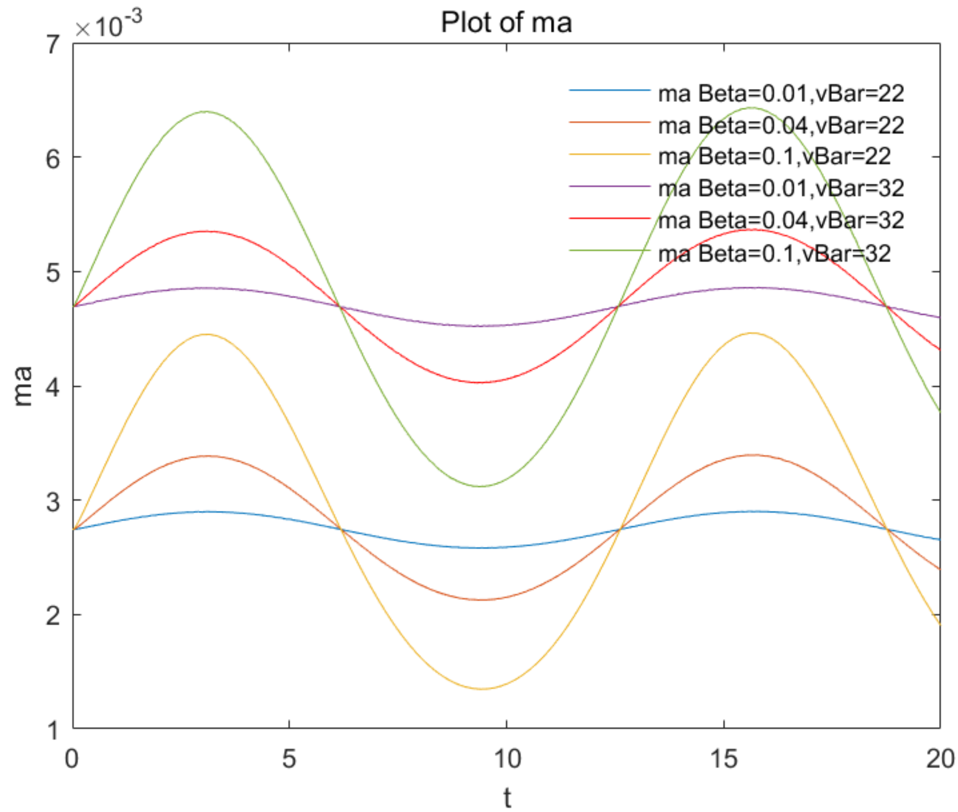


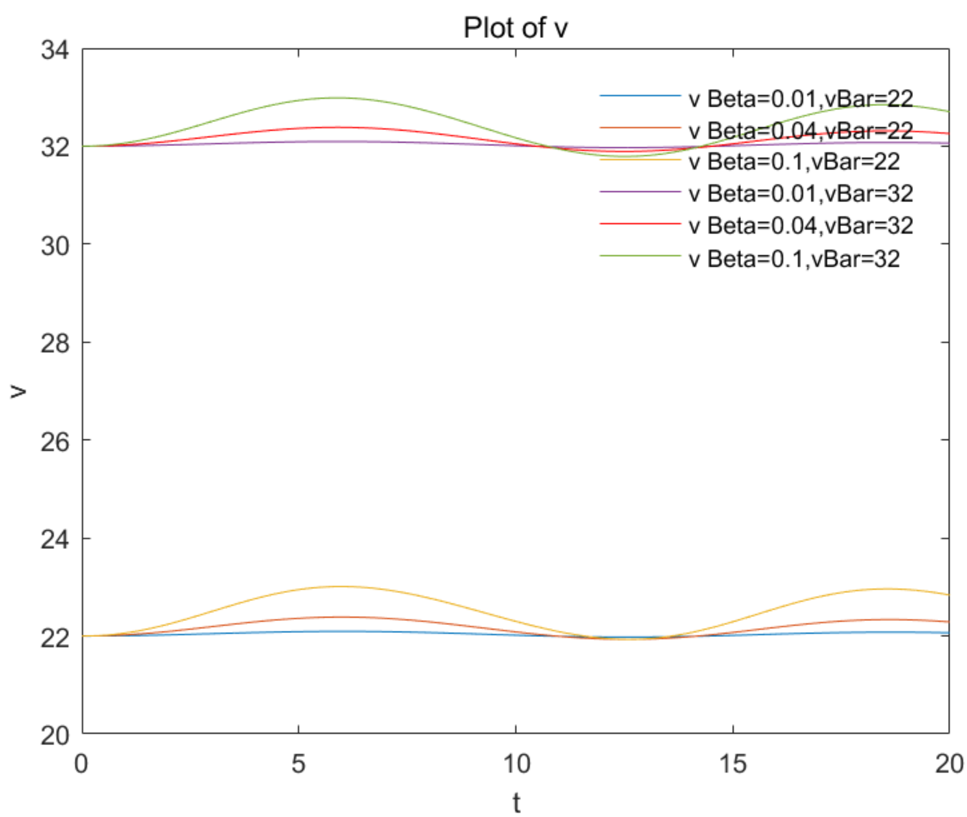
1(h)

`sinusoidThrottleSim` is autograded. In this section, carry out the simulations, and format the plots nicely.

```
c1=0.6;
c2=0.095;
c3=47500;
c4=0.0026;
TFinal=20;
w=0.5;
[tSol1, maSol1, vSol1] = sinusoidThrottleSim(22, 0.01,w,c1, c2, c3,
 c4, TFinal);

[tSol3, maSol3, vSol3] = sinusoidThrottleSim(22, 0.04,w,c1, c2, c3,
 c4, TFinal);

[tSol5, maSol5, vSol5] = sinusoidThrottleSim(22, 0.1,w,c1, c2, c3, c4,
 TFinal);

[tSol7, maSol7, vSol7] = sinusoidThrottleSim(32, 0.01,w,c1, c2, c3,
 c4, TFinal);

[tSol9, maSol9, vSol9] = sinusoidThrottleSim(32, 0.04,w,c1, c2, c3,
 c4, TFinal);
```

```
[tSol11, maSol11, vSol11] = sinusoidThrottleSim(32, 0.1,w,c1, c2, c3,
 c4, TFinal);

figure;
f4=figure(4);
set(f4,'name','sinusoidThrottleSim_ma 1(h)','Numbertitle','off');
hold on;
plot(tSol1,maSol1,tSol3,maSol3,tSol5,maSol5,tSol7,maSol7,tSol9,maSol9,'r',tSol11,m
%plot(tSol,d_vSol,'b-o','LineWidth',2);
%axis([-Inf Inf -1 1]);
xlabel('t');
ylabel('ma');
title('Plot of ma');
legend('ma Beta=0.01,vBar=22','ma Beta=0.04,vBar=22','ma
 Beta=0.1,vBar=22','ma Beta=0.01,vBar=32','ma Beta=0.04,vBar=32','ma
 Beta=0.1,vBar=32');
box on;
legend('boxoff');
hold off;

figure;
f5=figure(5);
set(f5,'name','sinusoidThrottleSim_v 1(h)','Numbertitle','off');
hold on;
plot(tSol1,vSol1,tSol3,vSol3,tSol5,vSol5,tSol7,vSol7,tSol9,vSol9,'r',tSol11,vSol11
%plot(tSol,d_vSol,'b-o','LineWidth',2);
%axis([-Inf Inf -1 1]);
xlabel('t');
ylabel('v');
title('Plot of v');
legend('v Beta=0.01,vBar=22','v Beta=0.04,vBar=22','v
 Beta=0.1,vBar=22','v Beta=0.01,vBar=32','v Beta=0.04,vBar=32','v
 Beta=0.1,vBar=32');
box on;
legend('boxoff');
hold off;
```


Plot of ma
×10^-3
7
6
5
4
3
2
1
ma
0
5
10
15
20
t
ma Beta=0.01,vBar=22
ma Beta=0.04,vBar=22
ma Beta=0.1,vBar=22
ma Beta=0.01,vBar=32
ma Beta=0.04,vBar=32
ma Beta=0.1,vBar=32


Plot of v
34
32
30
28
26
24
22
20
v
0
5
10
15
20
t
v Beta=0.01,vBar=22
v Beta=0.04,vBar=22
v Beta=0.1,vBar=22
v Beta=0.01,vBar=32
v Beta=0.04,vBar=32
v Beta=0.1,vBar=32

1(i)

`hcLinearModel` is autograded. Nothing to include in script file here. This section can make small illustration of its behavior.

```
fprintf('1(i) all good \n');
%Good
```

1(i) all good

1(j)

Illustrate the results of `hcLinearModel` at `vBar=22` and `vBar=32`

```
fprintf('1(j) \n');
c1=0.6;
c2=0.095;
c3=47500;
c4=0.0026;
[A_22, B_22, C_22, D_22, maBar_22, wBar_22, alphaBar_22] =
 hcLinearModel(22,c1, c2, c3, c4);
[A_32, B_32, C_32, D_32, maBar_32, wBar_32, alphaBar_32] =
 hcLinearModel(32,c1, c2, c3, c4);
fprintf('vBar=32, A=%d, B=%d, C=%d,D=%d, wBar=%d, alphabar=%d
 \n',A_22, B_22, C_22, D_22, maBar_22, wBar_22);
fprintf('vBar=32, A=%d, B=%d, C=%d,D=%d, wBar=%d, alphabar=%d
 \n',A_32, B_32, C_32, D_32, maBar_32, wBar_32);
```

```
1(j)
vBar=32, A=-1.620155e+01, B=1.304945e+03,
 C=-2.605959e-04,D=-2.727532e-02, wBar=2.583455e-01, alphabar=0
vBar=32, A=0, B=1.290000e-01, C=0,D=2.743115e-03, wBar=1.705426e
+02, alphabar=vBar=32, A=-2.356589e+01, B=1.304945e+03,
 C=-4.457699e-04,D=-3.834952e-02, wBar=3.956747e-01, alphabar=0
vBar=32, A=0, B=1.290000e-01, C=0,D=4.692315e-03, wBar=2.480620e+02,
 alphabar=
```

1(k)

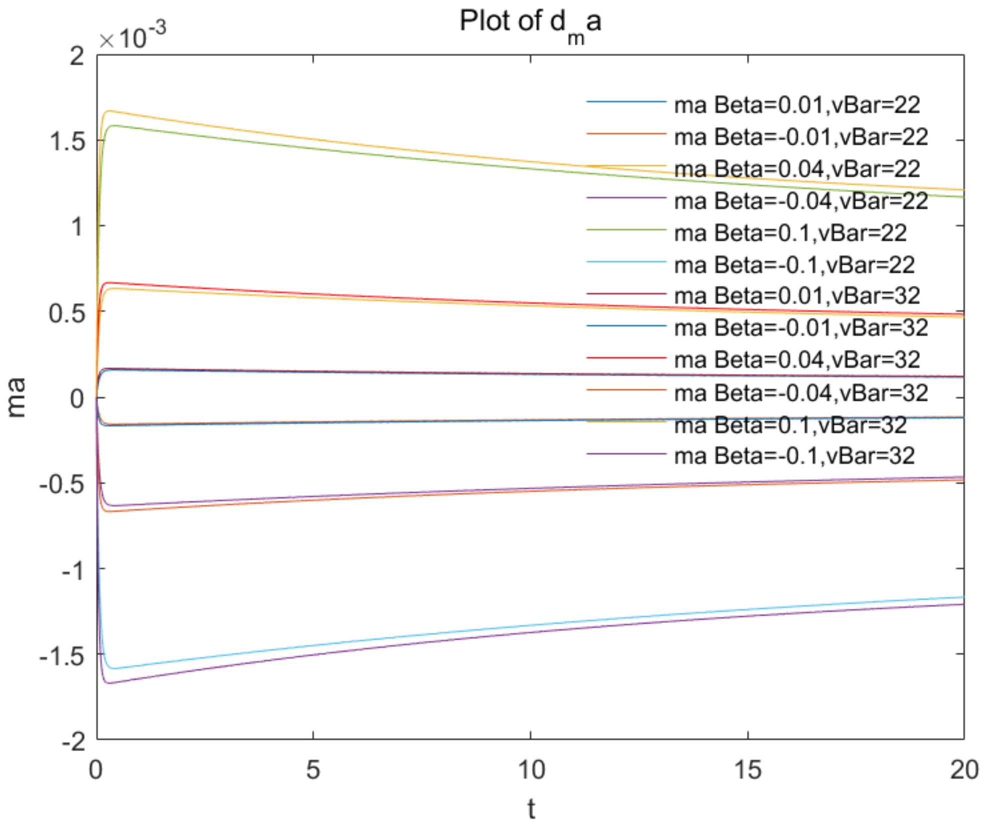
`constantThrottleLinearSim` is autograded. In this section, carry out the simulations, and format the plots nicely.

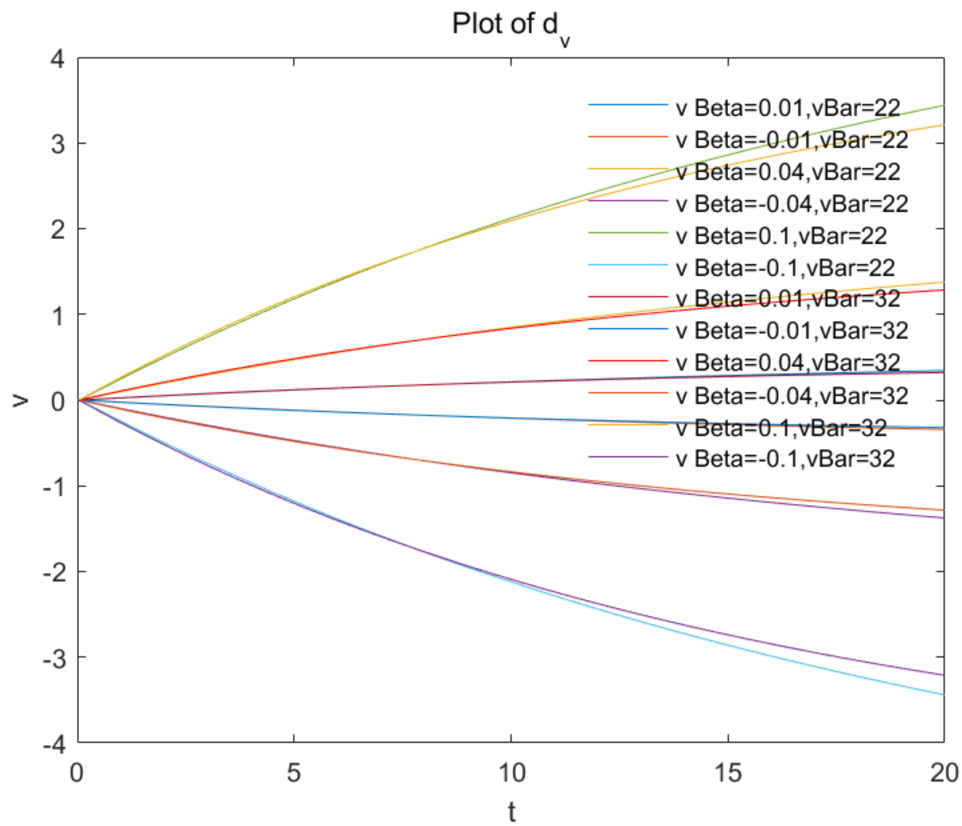
```
fprintf('1(k) \n');
c1=0.6;
c2=0.095;
c3=47500;
c4=0.0026;
TFinal=20;
[tSol1, maSol1, vSol1] = constantThrottleLinearSim(22, 0.01,c1, c2,
 c3, c4, TFinal);
[tSol2, maSol2, vSol2] = constantThrottleLinearSim(22, -0.01,c1, c2,
 c3, c4, TFinal);
[tSol3, maSol3, vSol3] = constantThrottleLinearSim(22, 0.04,c1, c2,
 c3, c4, TFinal);
```

```
[tSol4, maSol4, vSol4] = constantThrottleLinearSim(22, -0.04,c1, c2,
 c3, c4, TFinal);
[tSol5, maSol5, vSol5] = constantThrottleLinearSim(22, 0.1,c1, c2, c3,
 c4, TFinal);
[tSol6, maSol6, vSol6] = constantThrottleLinearSim(22, -0.1,c1, c2,
 c3, c4, TFinal);
[tSol7, maSol7, vSol7] = constantThrottleLinearSim(32, 0.01,c1, c2,
 c3, c4, TFinal);
[tSol8, maSol8, vSol8] = constantThrottleLinearSim(32, -0.01,c1, c2,
 c3, c4, TFinal);
[tSol9, maSol9, vSol9] = constantThrottleLinearSim(32, 0.04,c1, c2,
 c3, c4, TFinal);
[tSol10, maSol10, vSol10] = constantThrottleLinearSim(32, -0.04,c1,
 c2, c3, c4, TFinal);
[tSol11, maSol11, vSol11] = constantThrottleLinearSim(32, 0.1,c1, c2,
 c3, c4, TFinal);
[tSol12, maSol12, vSol12] = constantThrottleLinearSim(32, -0.1,c1, c2,
 c3, c4, TFinal);
figure;
f6=figure(6);
set(f6,'name','constantThrottleLinearSim_ma
 1(k)','Numbertitle','off');
hold on;
plot(tSol1,maSol1,tSol2,maSol2,tSol3,maSol3,tSol4,maSol4,tSol5,maSol5,tSol6,maSol6
%plot(tSol,d_vSol,'b-o','LineWidth',2);
%axis([-Inf Inf -1 1]);
xlabel('t');
ylabel('ma');
title('Plot of d_ma');
legend('ma Beta=0.01,vBar=22','ma Beta=-0.01,vBar=22','ma
 Beta=0.04,vBar=22','ma Beta=-0.04,vBar=22','ma Beta=0.1,vBar=22','ma
 Beta=-0.1,vBar=22','ma Beta=0.01,vBar=32','ma Beta=-0.01,vBar=32','ma
 Beta=0.04,vBar=32','ma Beta=-0.04,vBar=32','ma Beta=0.1,vBar=32','ma
 Beta=-0.1,vBar=32');
box on;
legend('boxoff');
hold off;

figure;
f7=figure(7);
set(f7,'name','constantThrottleLinearSim_v 1(k)','Numbertitle','off');
hold on;
plot(tSol1,vSol1,tSol2,vSol2,tSol3,vSol3,tSol4,vSol4,tSol5,vSol5,tSol6,vSol6,tSol7
%plot(tSol,d_vSol,'b-o','LineWidth',2);
%axis([-Inf Inf -1 1]);
xlabel('t');
ylabel('v');
title('Plot of d_v');
legend('v Beta=0.01,vBar=22','v Beta=-0.01,vBar=22','v
 Beta=0.04,vBar=22','v Beta=-0.04,vBar=22','v Beta=0.1,vBar=22','v
 Beta=-0.1,vBar=22','v Beta=0.01,vBar=32','v Beta=-0.01,vBar=32','v
 Beta=0.04,vBar=32','v Beta=-0.04,vBar=32','v Beta=0.1,vBar=32','v
 Beta=-0.1,vBar=32');
box on;
```

```
legend('boxoff');
hold off;
```

1(k)





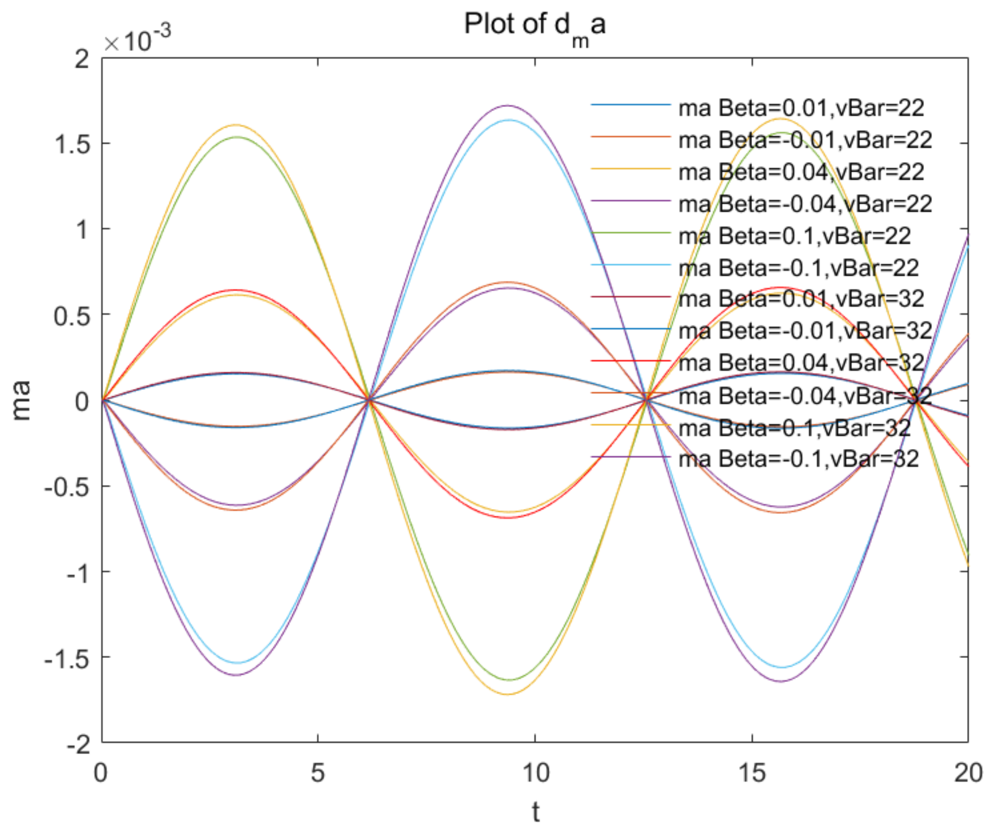
1(l)

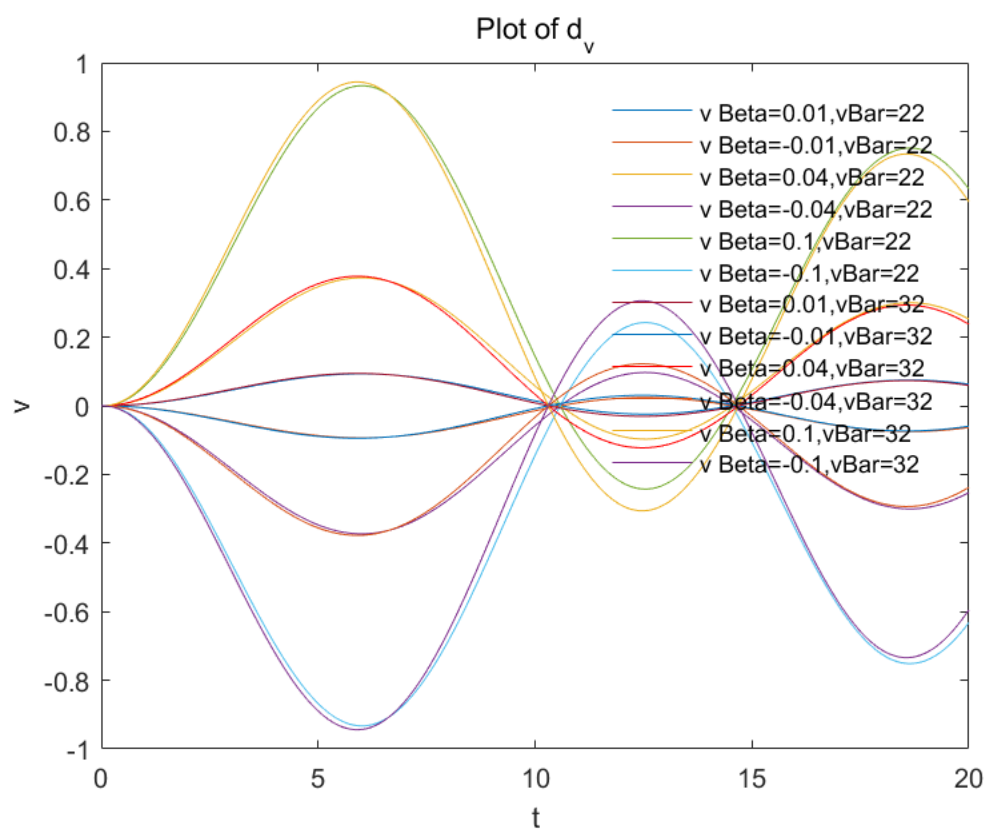
`sinusoidalThrottleLinearSim` is autograded. In this section, carry out the simulations, and format the plots nicely.

```
fprintf('1(I) all good \n');
c1=0.6;
c2=0.095;
c3=47500;
c4=0.0026;
TFinal=20;
w=0.5;
[tSol1, maSol1, vSol1] = sinusoidalThrottleLinearSim(22, 0.01,w,c1,
 c2, c3, c4, TFinal);
[tSol2, maSol2, vSol2] = sinusoidalThrottleLinearSim(22, -0.01,w,c1,
 c2, c3, c4, TFinal);
[tSol3, maSol3, vSol3] = sinusoidalThrottleLinearSim(22, 0.04,w,c1,
 c2, c3, c4, TFinal);
[tSol4, maSol4, vSol4] = sinusoidalThrottleLinearSim(22, -0.04,w,c1,
 c2, c3, c4, TFinal);
[tSol5, maSol5, vSol5] = sinusoidalThrottleLinearSim(22, 0.1,w,c1, c2,
 c3, c4, TFinal);
[tSol6, maSol6, vSol6] = sinusoidalThrottleLinearSim(22, -0.1,w,c1,
 c2, c3, c4, TFinal);
[tSol7, maSol7, vSol7] = sinusoidalThrottleLinearSim(32, 0.01,w,c1,
 c2, c3, c4, TFinal);
```

```
[tSol8, maSol8, vSol8] = sinusoidalThrottleLinearSim(32, -0.01,w,c1,
 c2, c3, c4, TFinal);
[tSol9, maSol9, vSol9] = sinusoidalThrottleLinearSim(32, 0.04,w,c1,
 c2, c3, c4, TFinal);
[tSol10, maSol10, vSol10] = sinusoidalThrottleLinearSim(32,
 -0.04,w,c1, c2, c3, c4, TFinal);
[tSol11, maSol11, vSol11] = sinusoidalThrottleLinearSim(32, 0.1,w,c1,
 c2, c3, c4, TFinal);
[tSol12, maSol12, vSol12] = sinusoidalThrottleLinearSim(32, -0.1,w,c1,
 c2, c3, c4, TFinal);
figure;
f8=figure(8);
set(f8,'name','sinusoidalThrottleLinearSim_ma
 1(i)','Numbertitle','off');
hold on;
plot(tSol1,maSol1,tSol2,maSol2,tSol3,maSol3,tSol4,maSol4,tSol5,maSol5,tSol6,maSol6
%plot(tSol,d_vSol,'b-o','LineWidth',2);
%axis([-Inf Inf -1 1]);
xlabel('t');
ylabel('ma');
title('Plot of d_ma');
legend('ma Beta=0.01,vBar=22','ma Beta=-0.01,vBar=22','ma
 Beta=0.04,vBar=22','ma Beta=-0.04,vBar=22','ma Beta=0.1,vBar=22','ma
 Beta=-0.1,vBar=22','ma Beta=0.01,vBar=32','ma Beta=-0.01,vBar=32','ma
 Beta=0.04,vBar=32','ma Beta=-0.04,vBar=32','ma Beta=0.1,vBar=32','ma
 Beta=-0.1,vBar=32');
box on;
legend('boxoff');
hold off;

figure;
f9=figure(9);
set(f9,'name','sinusoidalThrottleLinearSim_v
 1(i)','Numbertitle','off');
hold on;
hold on;
plot(tSol1,vSol1,tSol2,vSol2,tSol3,vSol3,tSol4,vSol4,tSol5,vSol5,tSol6,vSol6,tSol7
%plot(tSol,d_vSol,'b-o','LineWidth',2);
%axis([-Inf Inf -1 1]);
xlabel('t');
ylabel('v');
title('Plot of d_v');
legend('v Beta=0.01,vBar=22','v Beta=-0.01,vBar=22','v
 Beta=0.04,vBar=22','v Beta=-0.04,vBar=22','v Beta=0.1,vBar=22','v
 Beta=-0.1,vBar=22','v Beta=0.01,vBar=32','v Beta=-0.01,vBar=32','v
 Beta=0.04,vBar=32','v Beta=-0.04,vBar=32','v Beta=0.1,vBar=32','v
 Beta=-0.1,vBar=32');
box on;
legend('boxoff');
hold off;
```

1(I) all good


Plot of $d_m a$
$\times 10^{-3}$
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
0
5
10
15
20
t
ma
ma Beta=0.01,vBar=22
ma Beta=-0.01,vBar=22
ma Beta=0.04,vBar=22
ma Beta=-0.04,vBar=22
ma Beta=0.1,vBar=22
ma Beta=-0.1,vBar=22
ma Beta=0.01,vBar=32
ma Beta=-0.01,vBar=32
ma Beta=0.04,vBar=32
ma Beta=-0.04,vBar=32
ma Beta=0.1,vBar=32
ma Beta=-0.1,vBar=32


Plot of d_v
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
5
10
15
20
t
v
v Beta=0.01,vBar=22
v Beta=-0.01,vBar=22
v Beta=0.04,vBar=22
v Beta=-0.04,vBar=22
v Beta=0.1,vBar=22
v Beta=-0.1,vBar=22
v Beta=0.01,vBar=32
v Beta=-0.01,vBar=32
v Beta=0.04,vBar=32
v Beta=-0.04,vBar=32
v Beta=0.1,vBar=32
v Beta=-0.1,vBar=32

Problem 2

Stability of specific discrete-time system. `isSystemStable` is autograded. Illustrate its behavior here for a few values of `alpha`

```
fprintf('2 all good \n');
for alpha=[0.01 0.2 3 10];
    [TF] = isSystemStable(alpha);
    fprintf('alpha=%d,TF=%i \n',alpha,all(TF));
end
```

```
2 all good
alpha=1.000000e-02,TF=0
alpha=2.000000e-01,TF=0
alpha=3,TF=0
alpha=10,TF=0
```

Problem 3

Modeling an Euler-discretization of building heat transfer model

3(a)

`bldgHTM` is autograded. Illustrate its behavior here.

```
fprintf('3(a) all good \n');
```

```
3(a) all good
```

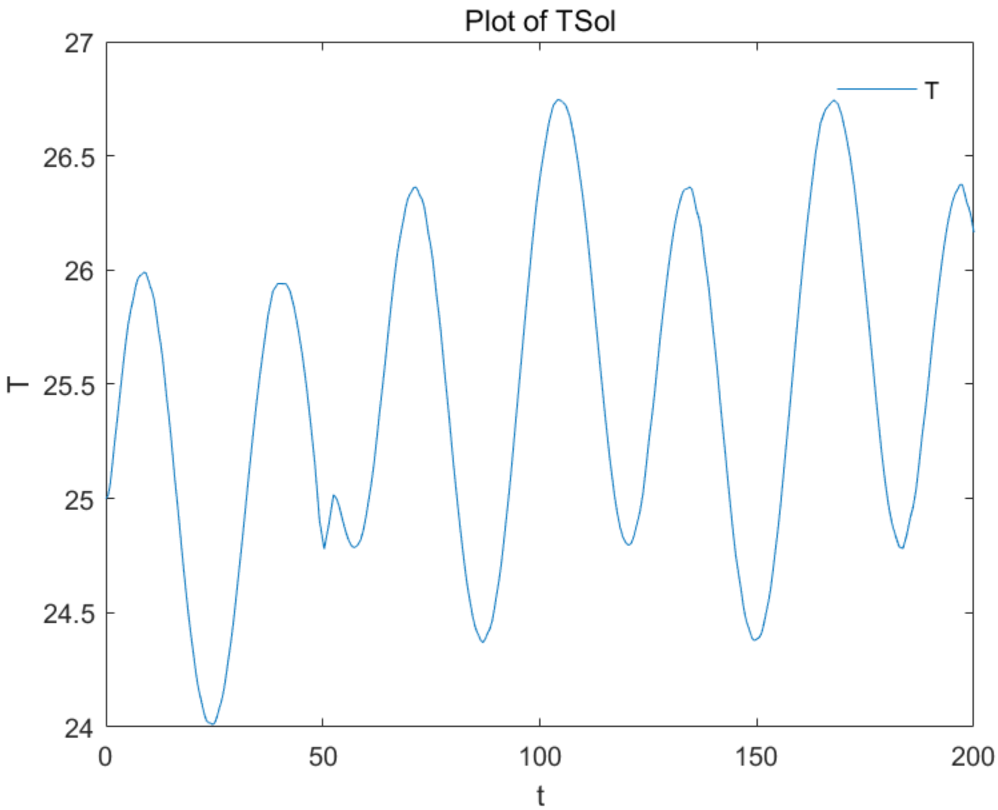
3(b)

Add code here to carry out the `ode45` simulation, including relevant plots.

```
fprintf('3(b) \n');
cp = 1;
mz = 100;
cz = 20;
u1H = @(t) 1000*(2 + sin(0.1*t));
u2H = @(t) 25 + sin(0.2*t);
qH = @(t) 1000*(t>50);
Tinit=25;
tSpan=[0 200];
%bldgHTMModel=@(t,T) (qH(t)+cp*u1H(t).*(u2H(t)-T))/(mz+cz);
[tSol11,TSol11]= ode45(@(t,T) bldgHTM(T, u1H(t), u2H(t), qH(t), mz,
 cz, cp),tSpan,Tinit);
figure;
f10=figure(10);
set(f10,'name','ode45 and bldgHTM  3(b)','Numbertitle','off');
hold on;
hold on;
plot(tSol11,TSol11);
%plot(tSol,d_vSol,'b-o','LineWidth',2);
%axis([-Inf Inf -1 1]);
```

```
xlabel('t');
ylabel('T');
title('Plot of TSol');
legend('T');
box on;
legend('boxoff');
hold off;
```

3(b)



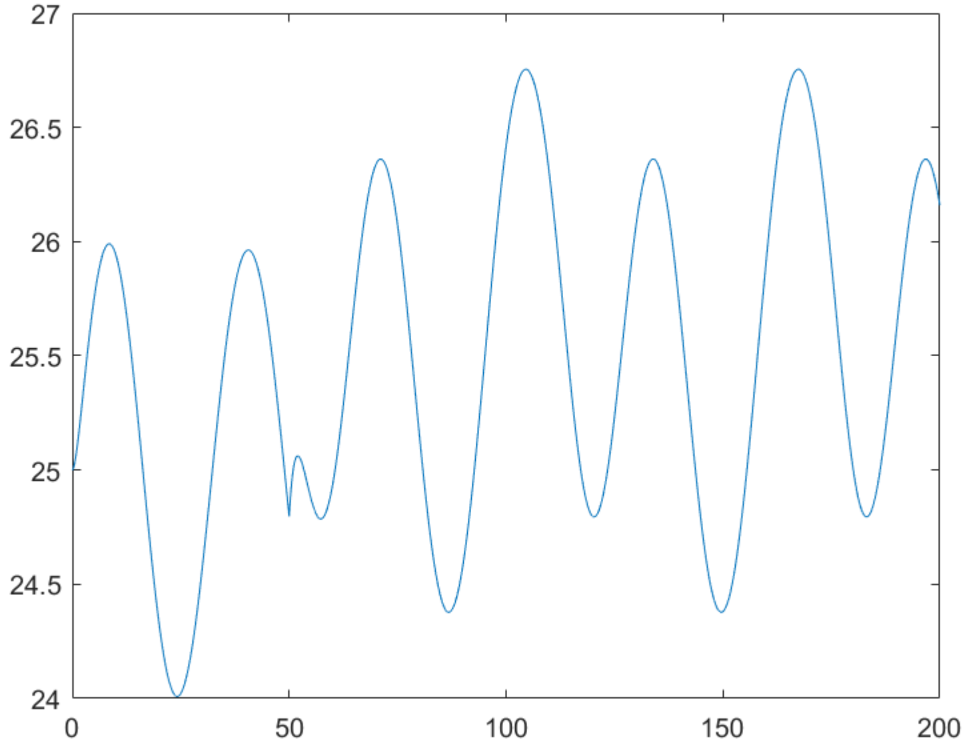
3(c)

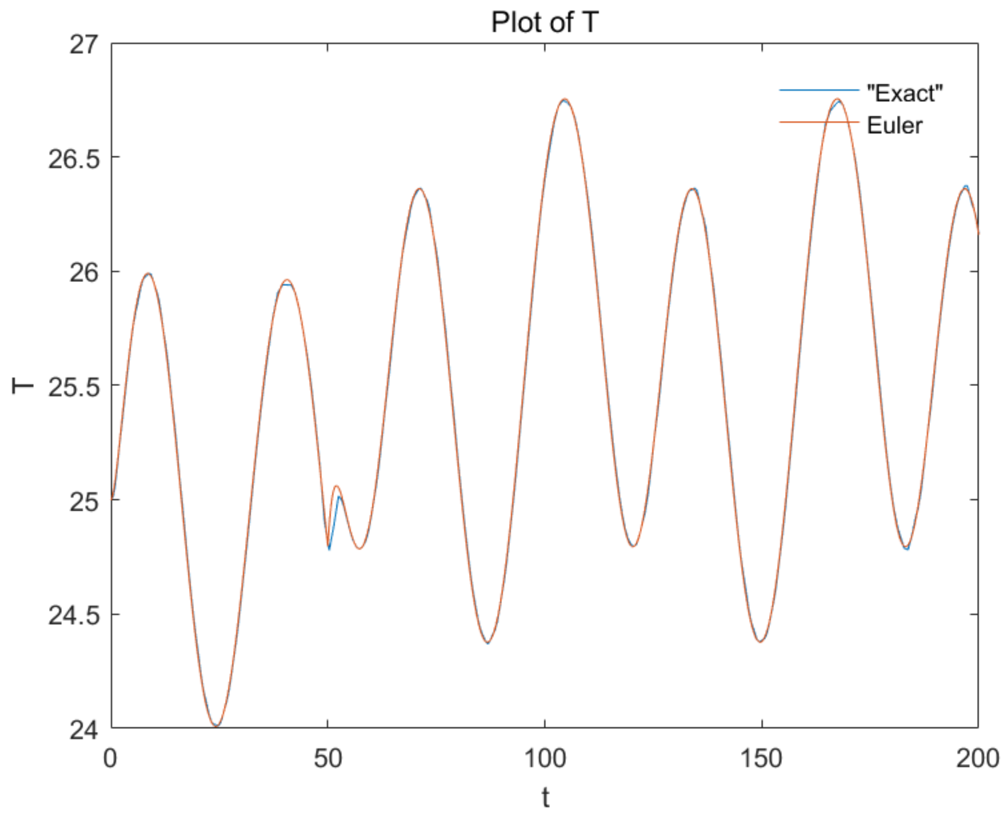
Add code here to carry out the first-order, forward Euler simulation, including relevant plots, and comparison to `ode45` simulation.

```
fprintf('3(c) \n');
TS=0.1;
[tESol,TESol]=recall_xEsol(200/0.1,0.1);
figure;
f11=figure(11);
set(f11,'name','3(b)','Numbertitle','off');
hold on;
plot(tSol11,TSol11,tESol,TESol);

%plot(tSol,d_vSol,'b-o','LineWidth',2);
%axis([-Inf Inf -1 1]);
```

```
xlabel('t');
ylabel('T');
title('Plot of T');
legend('"Exact"','Euler');
box on;
legend('boxoff');
hold off;
```

3(c)





Problem 4

Kinematic Bicycle model, simulation and animation

4(a)

`bikeFE` is autograded. Illustrate its behavior here.

```
fprintf('4(a) all good \n');
```

4(a) all good

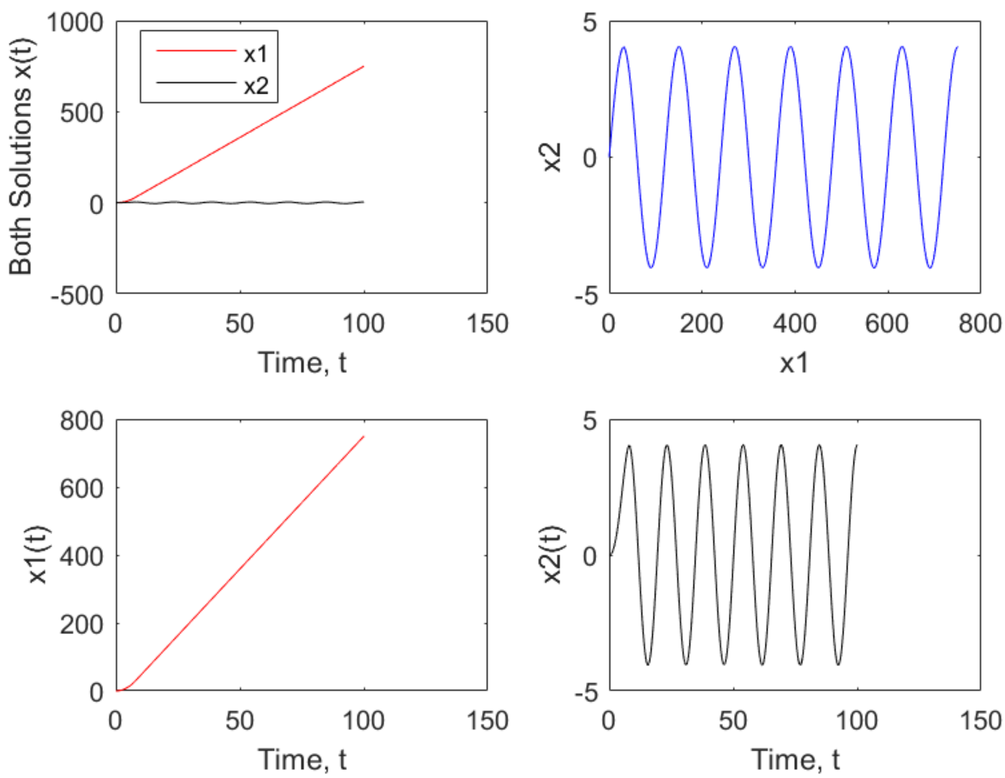
4(b)

`bikeFEsim` is autograded. Illustrate its behavior here, including a few plots. Data produced in this section will be used in the next section to build the animation.

```
fprintf('4(b) all good \n');
initState=[0 0 0 0];
TS=0.1;
[xE, yE, vE, psiE] = bikeFEsim(a, deltaF, initState, TS);
tSol12=zeros(length(xE),1);
for i=2:length(xE)
    tSol12(i)=tSol12(i-1)+TS;
```

```
end
figure;
f12=figure(12);
set(f12,'name','4(b)','Numbertitle','off');
hold on;
subplot(2,2,1)
plot(tSol12, xE, 'r', tSol12, yE, 'k');
legend('x1','x2','location','Best');
xlabel('Time, t')
ylabel('Both Solutions x(t)')
subplot(2,2,2)
plot(xE, yE, 'b');
xlabel('x1')
ylabel('x2')
subplot(2,2,3)
plot(tSol12, xE, 'r');
xlabel('Time, t')
ylabel('x1(t)')
subplot(2,2,4)
plot(tSol12,  yE, 'k');
xlabel('Time, t')
ylabel('x2(t)')
box on;
hold off;
```

4(b) all good



4(c)

Include code here to produce the animation, mimicing the ideas in the `movingLineMovieDemo.m` file introduced in Lab.

```
fprintf('4(c) \n');
%run movingLineMovieDemo;
```

```
4(c)
```

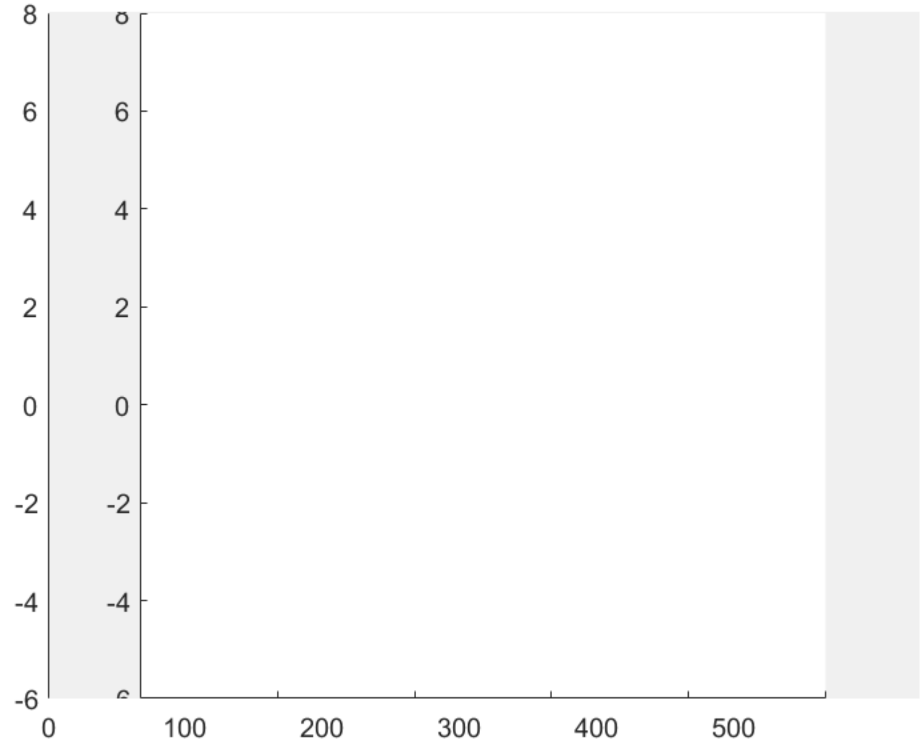
Demo of recording a moving-line Movie

This is a simple script to demonstrate some of the ideas to capturing a sequence of "frames", collecting them into an array, and finally viewing them sequentially as a movie, at a fixed, specified frame rate.

```
figure;
f13=figure(13);
set(f13,'name','4(c)','Numbertitle','off');
clf
% Initial center and orientation of line (uncaptured - see below)
%cx = 1; cy = 1; theta = 0; L = 0.5;
cx = 1; cy = 1; theta = 0; L = 5;
% Create initial line object (xdata, and ydata)
Lh = line([cx-L/2*cos(theta) cx+L/2*cos(theta)],...
    [cy-L/2*sin(theta) cy+L/2*sin(theta)]);
%+++
initState=[0 0 0 0];
TS=0.1;
%[xE, yE, vE, psiE] = bikeFEsim(a, deltaF, initState, TS);
nFrames = length(xE);
%+++
%nFrames = 400;
% Allocate a 1-by-nFrames STRUCT array with 2 fields
F(nFrames) = struct('cdata',[],'colormap',[]);
disp('Creating and recording frames...')
for j = 1:nFrames
    % Change center and angle, in a sensible manner, based on Frame#
    %+++
    cx=xE(j);
    cy=yE(j);
    theta=psiE(j);
    %+++
    %cx = cos(j/nFrames*pi);
    %cy = sin(j/nFrames*pi);
    %theta = 3*pi*j/nFrames;
    % Move the line to new location/orientation
    set(Lh,'xdata',[cx-L/2*cos(theta) cx+L/2*cos(theta)],...
        'ydata', [cy-L/2*sin(theta) cy+L/2*sin(theta)]);
    % Make sure the axis stays fixed (and square)
    axis([0 500 -6 8]); axis square
    % Flush the graphics buffer to ensure the line is moved on screen
    drawnow
    % Capture frame
```

```
    F(j) = getframe;
end
disp('Playing movie...')
Fps = 100;
nPlay = 1;
movie(F,nPlay,Fps)
```

```
Creating and recording frames...
Playing movie...
```



Attribution

Include you name, date and the class number. Zhipeng Yu ,2016/9/4 ME231A

```
fprintf('Zhipeng Yu ,2016/9/4 ME231A');
```

```
Zhipeng Yu ,2016/9/4 ME231A
```

Published with MATLAB® R2015b