Report for the 313 Computer Project – Team 30

09/22/2021

Team members	PSU ID	Contributions
Jim Ervin	jhe5074	Coding, Numerical solutions
Chris Fischer	cef5444	Numerical solutions, Analytical Solution
Akhilesh Mulgund	azm6412	Analytical solution, Report writing
Ralph Quartiano	rjq5038	Numerical Solution, Coding
Jeremy Tobin,	jwt5471	Analytical Solution, Report Writing

Section 1: Analytical Solutions

Computer Project 1.

1.1)

$x(0) = x_0 \quad \dot{x}(0) = \dot{x}_0$

$\ddot{x} + 2\zeta\omega\dot{x} + \omega^2 x = 0 \quad \rightarrow$ Goal: $x(t) = e^{-\zeta\omega t}\left[c_1\cos(\omega_d t) + c_2\sin(\omega_d t)\right]$

$s^2X(s) - sx_0 - \dot{x}_0 + 2\zeta\omega(sX(s) - x_0) + \omega^2X(s) = 0$

$$X(s) = \frac{2\zeta\omega x_0 + sx_0 + \dot{x}_0}{s^2 + 2\zeta\omega s + \omega^2 + \zeta^2\omega^2 - \zeta^2\omega^2}$$

$$= \frac{2\zeta\omega x_0 + sx_0 + \dot{x}_0}{(s+\zeta\omega)^2 + \underbrace{\omega^2(1-\zeta^2)}_{\omega_d^2}} \qquad \omega_d = \omega\sqrt{1-\zeta^2}$$

$$\frac{\zeta\omega x_0 + \zeta\omega x_0 + \dot{x}_0 + sx_0}{(s+\zeta\omega)^2 + \omega_d^2}$$

$$\frac{x_0(s+\zeta\omega)}{(s+\zeta\omega)^2 + \omega_d^2} + \frac{\overbrace{\zeta\omega x_0 + \dot{x}_0}^{C_2}}{\omega_d}\left(\frac{\omega_d}{(s+\zeta\omega)^2 + \omega_d^2}\right) = X(s)$$

(C_1)

Taking inverse Laplace:

$$x(t) = \underbrace{x_0}_{C_1}\, e^{-\zeta\omega t}\cos(\omega_d t) + \underbrace{\frac{\zeta\omega x_0 + \dot{x}_0}{\omega_d}}_{C_2} e^{-\zeta\omega t}\sin(\omega_d t)$$

$$\boxed{x(t) = e^{-\zeta\omega t}\left[C_1\cos(\omega_d t) + C_2\sin(\omega_d t)\right]}$$

$C_1 = x_0$

$C_2 = \dfrac{\zeta\omega x_0 + \dot{x}_0}{\omega_d}$

1.2)

$x(0) = \dot{x}(0) = 0$

$$\ddot{x} + 2\zeta\omega\dot{x} + \omega^2 x = T_0 u(t-a)$$

$$s^2 X(s) + 2\zeta\omega s X(s) + \omega^2 X(s) = \mathcal{L}\{T_0 u(t-a)\} = \frac{T_0 e^{-as}}{s}$$

$$X(s^2 + 2\zeta\omega s + \omega^2) = \frac{T_0 e^{-as}}{s}$$

$$X(s) = \frac{T_0 e^{-as}}{s(s^2 + 2\zeta\omega s + \omega^2)}$$

Partial Fractions:

$$\left[\frac{A}{s} + \frac{Bs + C}{s^2 + 2\zeta\omega s + \omega^2}\right] = 1$$

$$As^2 + 2A\zeta\omega s + A\omega^2 + Bs^2 + Cs = 1$$

$s=0 \quad A\omega^2 = 1 \quad A = \frac{1}{\omega^2}$

$s=1 \quad \frac{1}{\omega^2} + \frac{2\zeta}{\omega} + 1 + B + C = 1$

$s=-1 \quad \frac{1}{\omega^2} - \frac{2\zeta}{\omega} + 1 + B - C = 1$

(+)

$$\frac{2}{\omega^2} + 2 + 2B = 2$$

$$B = -\frac{1}{\omega^2}$$

$$\frac{4\zeta}{\omega} + 2C = 0$$

$$C = -\frac{2\zeta}{\omega}$$

$$T_0 e^{-as}\left(\frac{1}{\omega^2 s} - \frac{\frac{2\zeta\omega}{\omega^2}}{s^2 + 2\zeta\omega s + \omega^2} - \frac{s/\omega^2}{s^2 + 2\zeta\omega s + \omega^2}\right)$$

$$\frac{T_0 e^{-as}}{\omega^2}\left(\frac{1}{s} - \frac{\zeta\omega}{(s+\zeta\omega)^2 + \omega_d^2} - \frac{(s+\zeta\omega)}{(s+\zeta\omega)^2 + \omega_d^2}\right)$$

$$s^2 + 2\zeta\omega s + \omega^2 + \zeta^2\omega^2 - \zeta^2\omega^2$$
$$= (s+\zeta\omega)^2 + \omega^2(1-\zeta^2)$$
$$= (s+\zeta\omega)^2 + \omega_d^2$$

$$\frac{\zeta\omega}{\omega_d}\left(\frac{\omega_d}{(s+\zeta\omega)^2 + \omega_d^2}\right)$$

Taking inverse Laplace

$$\frac{T_0 e^{-as}}{\omega^2}\ \mathcal{L}^{-1}\left[\frac{1}{s} - \frac{\zeta\omega}{\omega_d}\left(\frac{\omega_d}{(s+\zeta\omega)^2+\omega_d^2}\right) - \frac{(s+\zeta\omega)}{(s+\zeta\omega)^2+\omega_d^2}\right]$$

$$= \frac{T_0 u(t-a)}{\omega^2}\left[1 - \frac{\zeta\omega}{\omega_d}\left(e^{-\zeta\omega(t-a)}\sin(\omega_d(t-a))\right) - e^{-\zeta\omega(t-a)}\cos(\omega_d(t-a))\right]$$

$$z(t) = \frac{T_0 u(t-a)}{\omega^2}\left[1 - e^{-\zeta\omega(t-a)}\left(\cos(\omega_d(t-a)) + \frac{\zeta\omega}{\omega_d}\sin(\omega_d(t-a))\right)\right]$$

$y' = f(t,y)$

$x'' + 2\zeta\omega x' + \omega^2 x = 0$

$x'' = -(2\zeta\omega y_2 + \omega^2 y_1)$

$$y = \begin{bmatrix} x \\ x' \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$$

$$y' = \begin{bmatrix} x' \\ x'' \end{bmatrix} = \begin{bmatrix} y_2 \\ -(2\zeta\omega y_2 + \omega^2 y_1) \end{bmatrix}$$

$$y' = \begin{bmatrix} y_2 \\ -(2\zeta\omega y_2 + \omega^2 y_1) \end{bmatrix}$$

$$y(0) = \begin{bmatrix} x(0) \\ x'(0) \end{bmatrix} = \begin{bmatrix} x_0 \\ \dot{x}_0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Section 2: Numerical Solutions

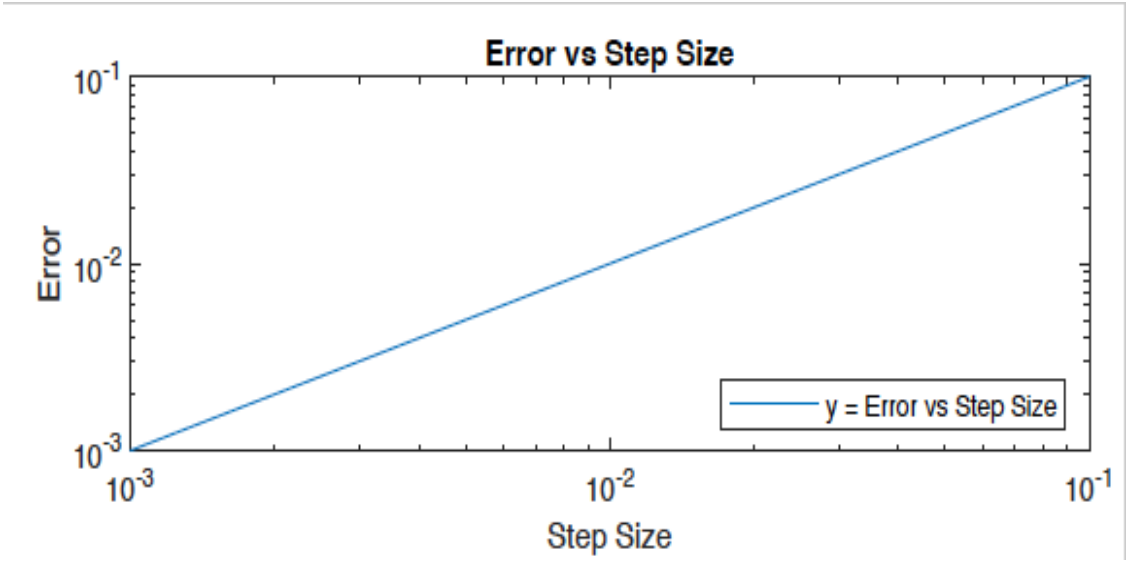

Figure 1: Error vs. Step Size plot shows increase in error with increase in step size. This result is consistent with our coursework.

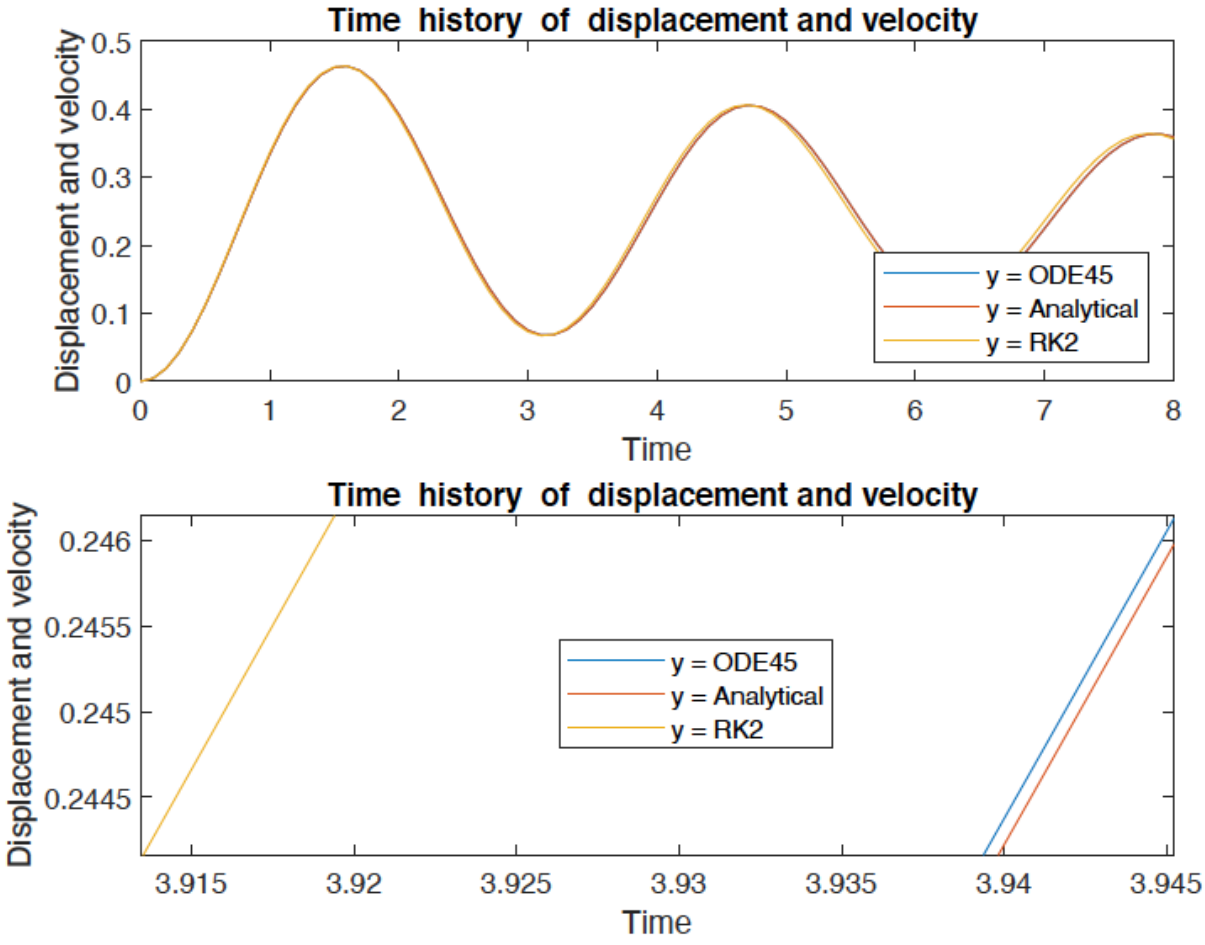

Figure 2: The time history of displacement and velocity of the three different solutions the team used. The three solutions utilized by the team were ode45, Improved Euler and the Analytical solution. The bottom graph shows a magnified version of the top graph. These graphs show that two of the solutions are almost perfectly aligned on the graph.

Throughout the completion of this project, the team noticed that ode45 solution was very similar to the analytical solution. The team had to magnify the top graph in Figure 2 to see the difference between these two solutions. The ode45 solution uses a version of the RK4 algorithm and as a result, it has greater accuracy than the RK2 algorithm.

Section 3: Application of the Computer Program

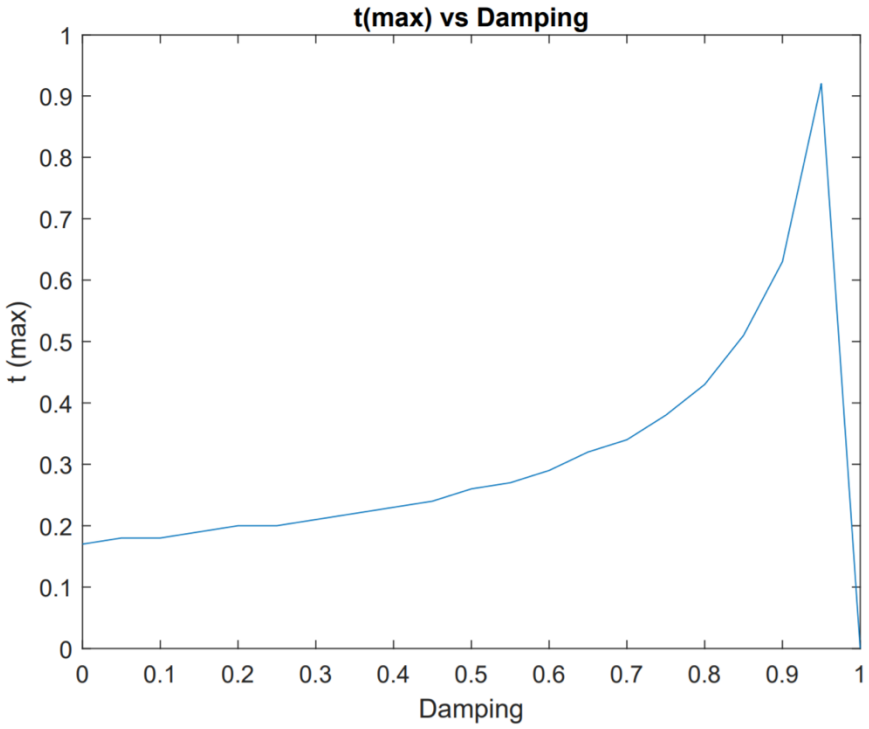

Figure 3: Variation of t(max) as Damping constant increases

As seen in Figure 3, as the Damping constant increases, the t(max) also increases. However, as the Damping constant is set to 1, the value for t(max) is 0. This means that the target altitude is not reached in the time span.

- 1(2) The drone will stay at the height defined by T_0 / w^2 when t is sufficiently large.
- 2(2) At a glance, ode45, analytical, and improved all appear to be nearly identical. Upon closer examination, improved is less accurate than ode45 and the analytical solution. Ode45 and the analytical solution appear nearly identical and are both much more accurate than the improved solution.

- 2(3) The log-log plot of error vs step size shows that as step size increases, error also increases linearly. These results, shown in Figure 1, are consistent with our understanding of differential equation approximations.
- 3(2) The sweep of zeta values shown in Figure 3 that as zeta increases, so too does t(max). In other words, a drone with a stronger damping constant will take longer to reach the desired altitude. The range of zeta values that reaches the desired altitude of 0.1 within t(max) = 0.20 is 0 to 0.20.

Section 4: Summary

This project allowed us to utilize numerical methods to model the flight of a quadrotor. By analytically analyzing flight dynamics and comparing it with computer generated numerical solutions, the team was able to compare accuracies of different mathematical methods, such as RK2 and the Improved Euler Method.

This project allowed us to gain a new understanding of MATLAB and how to utilize it to approximate ODEs, functions, and calculate error. The team learned how to use the ode45 function and how to implement the Euler methods in a real-life application. The team also developed an RK2 algorithm, a second order Runge-Kutta method in MATLAB to compare it with other forms of approximations. In Figure 2, the team used the ode45 method, the analytical method and the rk2 method. The rk2 method was less accurate than the ode45 which utilizes rk4, when compared to the analytical. This is shown in Figure 2 in bottom graph.

The Team observed how changing certain parameters and initial conditions affected the flight of the drone. As the zeta values increased, the time to reach the desired altitude also increased. This makes physical sense, since the damping constant is applied to the velocity, and by increasing the zeta value the max acceleration is reduced. The physical advantage of this is that the amplitude of the first oscillation is reduced, but the downside is that reaching the desired altitude is slower.

Appendix: The Computer Program

```
%AERSP 313 Project 1 Code (PART 2)
%Group Members: James Ervin, Christopher Fischer, Akhilesh Mulgund,
 Jeremy
%Tobin, Ralph Quartiano

clc;
clear;

%Defining drone parameters and initial conditions
z1 = 0.05;  % Zeta for part 2
o1 = 2;     % Omega for part 2
T0 = 1;     % T0 for part 2
t=0:0.1:8;
y0 = [0,0]; %Initial conditions for displacement and velocity

%Part 2-1a: Solving with ode45
[t,y_ode45] = ode45(@myfunction,t,y0); %'myfunction' written below
%Individual plot (commented out for final submission)
% plot(t,y_ode45(:,1));
% xlabel('t')
% ylabel('y')

%Part 2-2a: Analytical solution
y_analytical=zeros(1,81);
a=(T0/(o1*o1));
b=z1*o1;
od=o1*sqrt(1-(z1*z1));
c=((z1*o1)/(od));
for i = 1:81
y_analytical(i)=a*(1-(exp(-b*t(i))*(cos(od*t(i))+c*sin(od*t(i)))));
end
%Individual plot (commented out for final submission)
% plot(t,y_analytical)
% xlabel('t');
% ylabel('y');
%Part 2-2b: Comparison plot
plot(t,y_ode45(:,1));
hold on
% hold on
plot(t,y_analytical);
hold off

y_rk2=zeros(2,81);

steps = [0.05,0.025,0.001];

y_approx01=rk2(y_rk2,0.1,z1)
y_approx005=rk2(y_rk2,steps(1),z1)
y_approx0025=rk2(y_rk2,steps(2),z1)
y_approx0001=rk2(y_rk2,steps(3),z1)

subplot(2,1,2)
```

```
plot(t,y_ode45(:,1));
hold on
plot(t,y_analytical);
hold on
plot(t,y_approx01(1,:));
title('Time  history  of  displacement and velocity ')
xlabel('Time')
ylabel('Displacement and velocity')
legend({'y = ODE45','y = Analytical','y =
 RK2'},'Location','southeast')
hold off


subplot(2,1,1)
plot(t,y_ode45(:,1));
hold on
plot(t,y_analytical);
hold on
plot(t,y_approx01(1,:));
title('Time  history  of  displacement and velocity ')
xlabel('Time')
ylabel('Displacement and velocity')
legend({'y = ODE45','y = Analytical', 'y =
 RK2'},'Location','southeast')
hold off


% Time vs Displacement and velocity for varying step sizes. Since only
% error was requested in problem statement, this plot has been
 commented
% out for the final submission.
% subplot(2,1,2)
% size(y_approx01,2);
% size(t,2)
% plot(t,y_approx01(1,:));
% hold on
% subplot(2,1,2)
% t1=0:steps(1):8;
% plot(t1,y_approx005(1,:));
% hold on
% subplot(2,1,2)
% t2=0:steps(2):8;
% plot(t2,y_approx0025(1,:));
% hold on
% subplot(2,1,2)
% t3=0:steps(3):8;
% plot(t3,y_approx0001(1,:));
% title('Time  history  of  displacement and velocity ')
% xlabel('Time')
% ylabel('Displacement and velocity')
% legend({'y = Step size 0.1','y = Step size 0.05','y = Step size
 0.025','y = Step size 0.001'},'Location','southeast')
% hold off
```

```
y_error01= vecnorm(y_analytical(end)-y_approx01(80))
y_error005 = vecnorm(y_analytical(end)-y_approx005(end-1))
y_error0025 = vecnorm(y_analytical(end)-y_approx0025(end-1))
y_error0001 = vecnorm(y_analytical(end)-y_approx0001(end-1))

stepsfinal= [0.1,0.05,0.025,0.001];




% Error Plot
subplot(2,1,1)
y_errors=[0.1,steps(1),steps(2),steps(3);y_error01,y_error005,y_error0025,y_error0
plot = loglog(stepsfinal,y_errors(1,:));
title('Error vs Step Size')
xlabel('Step Size')
ylabel('Error')
legend({'y = Error vs Step Size'},'Location','southeast')
hold off

%Part 2-1b: Solving with our own program
function y = rk2(x,h,z)
t=0:h:8;
z=0.05;
for i = 1:((8/h))
        t(i)=h*i;
        t(i+1)=h*(i+1);
        k1=myfunction(t(i),x(:,i));
        k2=myfunction(t(i)+0.5*h,x(:,i)+0.5*h*k1);
        x(:,i+1)=x(:,i)+h*k2;
end
y=x;
end
%This function represents the ODE describing drone behavior given in
 the
%project prompt. It is written in vector notation as assigned in
 project
%part 1-3.
function y = myfunction(t,x)
z1 = 0.05;  % Zeta for part 2
o1 = 2;     % Omega for part 2
T0 = 1;
y1= x(2);
y2 = T0-(2*z1*o1*x(2)+(o1*o1)*x(1));

y = [y1; y2];

end


y_approx01 =
```

```
%AERSP 313 Project 1 Code (PART 3)
%Group Members: James Ervin, Christopher Fischer, Akhilesh Mulgund,
%CSWA Certified Jeremy Tobin, Ralph Quartiano
clear;
clc;

%defining parameters
h=0.01;
t=0:h:7.99;
z = 0:0.05:1;
%initializing vectors used in rk2 algorithm. final approximations are
%stored in y_rk2, with the third index storing zeta valuas
y=zeros(2,100);
y_rk2=zeros(2,800,size(z,2));

%implementing rk2 algorithm for different zeta values
for j = 1:size(z,2)
y_rk2(:,:,j)=rk2(y,h,z(j));
end
%plotting approximations
for i = 1:size(y_rk2,3)-1
plot(t,y_rk2(1,:,i))
hold on
end
hold off

%storing and plotting times at which each approximation first hits H=0.01
times = zeros(1,21);
for j = 1:size(y_rk2,3)-1
    times(j)=threshold(y_rk2,j);
end

%Plot of damping constant vs tmax
% plot(z,times)
% title('t(max) vs Damping')
% ylabel('t (max)')
% xlabel('Damping')
% %legend({'y = Damping vs t(max)'},'Location','southeast')
% hold off


%threshold function which detects passing H value in rk2 approximation
function t = threshold(y,j)
h=0.01;
for i = 1:size(y,2)
    if y(1,i,j) >= 0.01
        t=h*i;
        break
    end
end
end
```

```
%rk2 algorithm, based on lecture notes
function y = rk2(x,h,z)
t=0:h:8;
for i = 1:((8/h))-1
        t(i)=h*i;
        t(i+1)=h*(i+1);
        k1=myfunction(t(i),x(:,i),z);
        k2=myfunction(t(i)+0.5*h,x(:,i)+0.5*h*k1,z);
        x(:,i+1)=x(:,i)+h*k2;
end
y=x;
end

%This function represents the ODE describing drone behavior given in the
%project prompt. It is written in vector notation as assigned in project
%part 1-3.
function y = myfunction(t,x,z)
%z1 = 0.05;  % Zeta for part 2
o1 = 10;     % Omega for part 2
T0 = 1;
y1= x(2);
y2 = T0-(2*z*o1*x(2)+(o1*o1)*x(1));

y = [y1; y2];

end
```