RMIT UNIVERSITY

Course code: AERO2253

Course name: Aerospace Dynamics and Control

……………/90

Assignment 1

Jay Dickson

RMIT, Bundoora, Australia

RMIT UNIVERSITY

TABLE OF CONTENTS

1 Introduction (5 points)

Considered herein are a set of methods that can be used to solve for the transient response of a 1 degree of freedom system. Of particular concern is the advantages and disadvantages of each method as well as the accuracy. In this report the three methods considered are the Laplace, Numerical Convolution and MATLAB ode45. In each section a MATLAB implementation of the relevant method will be constructed to ensure rigour and repeatability.

Further a discussion of the general approach taken when solving and modelling dynamic systems is also included along with explanations of the various properties and characteristics of these systems such as the general mathematical forms that can be used to describe a dynamic system and how to construct this model from using Newtons second law.

2 Technical characteristics of 1DOF system (5 points)

We are considering a 1 degree of freedom mass-spring-damper system. This can be modeled using a second order differential equation the form of which is presented in *section 2.1*. As this system has only one degree of freedom it is capable of only translational motion along a single axis. We will assume that to be the x axis.

The mass provides an inertial component to the system and its effect on the system is described with F = ma or $F = m\ddot{x}$.

The damping of the system could be provided by friction or a damping device this component is described with $F = B\dot{x}$.

The final component the stiffness could be provided by some type of spring this component is described with $F = kx$.

In this case the complete equation of motion can be gained by combining these descriptions and equating them to an input function.

Table 1 – Characteristics of the system derived from student number (s3719855)

Mass (Kg)	Damping (Ns/m)	Stiffness (N/m)
38	855	37800

Calculations of characteristics

Student # = 3719855

Mass = 3+7+1+9+8+5+5 = 38 Kg
Damping = Last 3 Digits = 855 Ns/m
Stiffness = 3*7*9*8*5*5 = 37800 N/m

2.1 Linear homogeneous differential equation in general form (5 points)

A linear homogeneous differential equation is an equation that relates the rate of change for connected functions to each other. The linear component refers to the structure of the equation where it contains only addition or subtraction between functions or can be written this way. The homogenous part requires the equation be equated to zero when all derivatives are on one side of the equation.

A linear second order equation can nicely represent a dynamic system as the properties of the system can be written in terms of first and second order derivatives of the position or the position itself. Such as $F = kx$ or $F = m\ddot{x}$. As such a second order equation can model these systems.

General Form:

$$m\,\ddot{x} + c\,\dot{x} + kx = 0$$

Where,

m: mass,
c: damping coefficient,
k: spring coefficient

So,

$$38\,\ddot{x} + 855\,\dot{x} + 37800x = 0$$

Standard form:

$$\ddot{x} + 2\zeta\omega_n\dot{x} + \omega_n^2 x = 0$$

Where,

ζ: Damping Ratio,
ω_n: Natural Frequency

So,

$$\ddot{x} + 22.5\dot{x} + 994.74x = 0$$

This form is useful as it allows for simple calculation of the systems properties.

Find characteristic equation

The characteristic equation allows for assessment of some aspects of the total systems behaviour such as its stability.

Assume solution:

$$\mathrm{x} = \mathrm{e}^{\mathrm{st}}$$

Differentiate, substitute and factorise:

$$e^{st}(s^2 + 22.5s + 994.74) = 0$$

Characteristic equation:

$$s^2 + 22.5s + 994.74 = 0$$

Find Roots using quadratic equation

The roots tell us about the stability of the system, for example if the real parts of the roots are negative the system is stable which in this case they are. This can be seen again in *section 2.4* where poles in the left (negative) half of the graph imply stability.

$$\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Where,

B = Damping (855 Ns/m),
A = Mass (38 Kg),
C = Spring coefficient (37800 N/m)

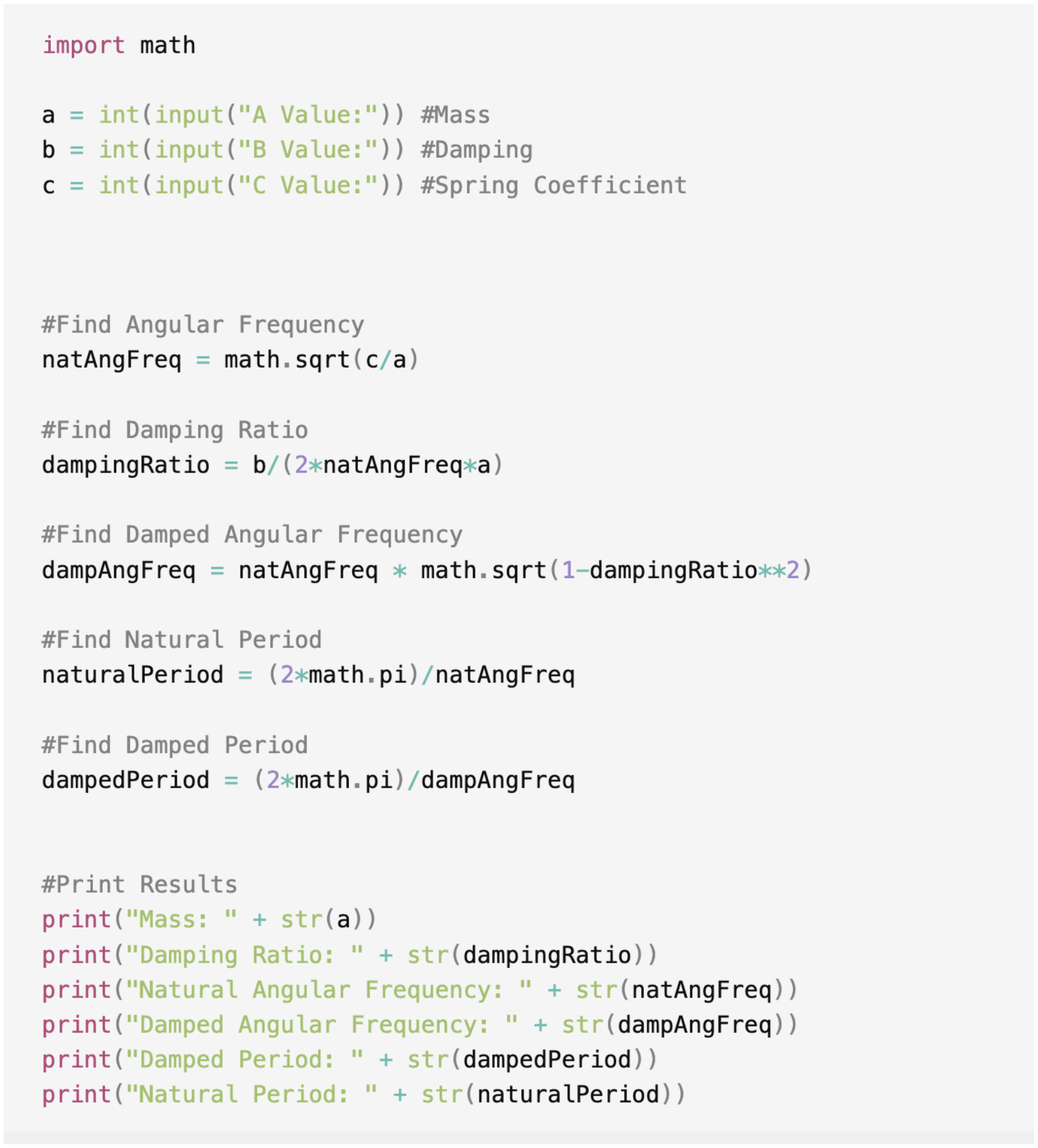
Roots:

$$s1 = -11.25 + i29.46$$
$$s2 = -11.25 - i29.46$$

2.2 Find natural frequency, damping ratio and damped frequency if applicable (5 points)

Using the python script[*] bellow *Figure-1*, we can take advantage of the relationship between significant system properties and the standard form of second order differential equation. As per this equation:

$$\ddot{x} + 2\zeta\omega_n\dot{x} + \omega_n^2 x = 0$$



```python
import math

a = int(input("A Value:")) #Mass
b = int(input("B Value:")) #Damping
c = int(input("C Value:")) #Spring Coefficient



#Find Angular Frequency
natAngFreq = math.sqrt(c/a)

#Find Damping Ratio
dampingRatio = b/(2*natAngFreq*a)

#Find Damped Angular Frequency
dampAngFreq = natAngFreq * math.sqrt(1-dampingRatio**2)

#Find Natural Period
naturalPeriod = (2*math.pi)/natAngFreq

#Find Damped Period
dampedPeriod = (2*math.pi)/dampAngFreq


#Print Results
print("Mass: " + str(a))
print("Damping Ratio: " + str(dampingRatio))
print("Natural Angular Frequency: " + str(natAngFreq))
print("Damped Angular Frequency: " + str(dampAngFreq))
print("Damped Period: " + str(dampedPeriod))
print("Natural Period: " + str(naturalPeriod))
```

Figure 1 – Python Script to find properties of the system

[*] Could of course have used MATLAB but I already had written this script before starting this assignment, so I elected to use this instead of re-writing it.

Table 2 – Properties of the system

Property	Value
Damping Ratio	0.36
Natural Frequency	31.54 rad/s
Damped frequency	29.46 rad/s

Damping Ratio – Property that describes how the system acts to reduce or prevent occultations

Natural Frequency – The frequency of the system without damping

Damped frequency – The frequency of the system under the effect of damping

2.3 Define transfer function (5 points)

The transfer function relates the systems output and input more specifically the ratio of input to output. This is useful as it allows for computation of a variety of properties, notably it allows for the plotting of the pole-zero graph which presents a useful visual way to inspect the systems properties.

Function describing complete motion

$$\ddot{x} + 22.5\dot{x} + 994.74x = f(t)$$

Take the Laplace transform of both sides

Initial Conditions:

$$x(0) = 0$$
$$x'(0) = 0$$

Laplace:

$$s^2X(s) + 22.5sX(s) + 994.74X(s) = F(s)$$

Factorise:

$$X(s)(s^2 + 22.5s + 994.74) = F(s)$$

Transfer function:

$$\frac{X(s)}{F(s)} = \frac{1}{s^2 + 22.5s + 994.74}$$

2.4 Plot poles and zeros (5 points)

Using the calculated transfer function, we can plot the zeros of which this system has none and the poles of which this system has three. Pole-zero plots are useful as they describe qualities such as the stability and degree of oscillation for a system. We can use MATLAB to plot these values for a given transfer function.

```
%Find Poles and Zeros

f = tf(1,[1 22.5 994.74]) %Define Transfer Function

%Plot Poles and Zeros
pzmap(f)
grid, axis([-15 0 -35 35]) %Set Range and Domain
```

Figure 2 – MATLAB code to plot poles and zeros

The below plot shows no zeros and two poles. These poles are in the left (negative) half of the graph and so indicate stability. The poles also have imaginary component indicating oscillation and do not lie on the imaginary axis so this system will oscillate and decay exponentially.

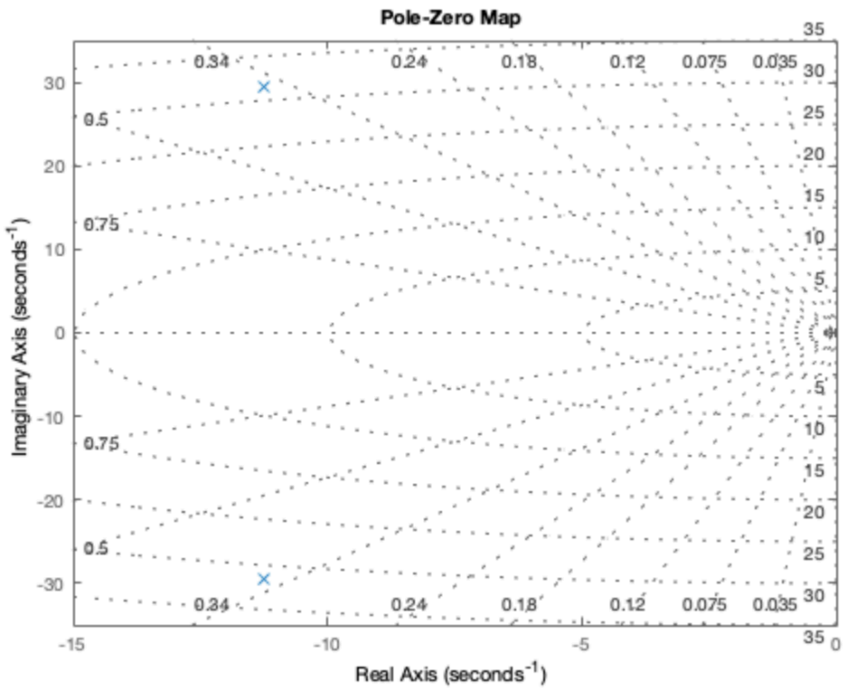


Figure 3 – Poles and Zeros for transfer function

3 Transient response in Laplace (3 points)

The Laplace method takes advantage of the simplicity associated with solving a differential equation in the s-domain as opposed to the t-domain. By taking the Laplace transform of a differential equation. We can then solve the problem algebraically and take the inverse Laplace to return to the t-domain with a solved equation.

The strongest advantage of this approach is its simplicity, once an equation describing the motion of the system has been defined the Laplace transform can be applied for many inputs using a table or computer package with relatively little variation needed in the approach. It also makes computation by hand possible which is quite difficult with other approaches.

Method is continuous

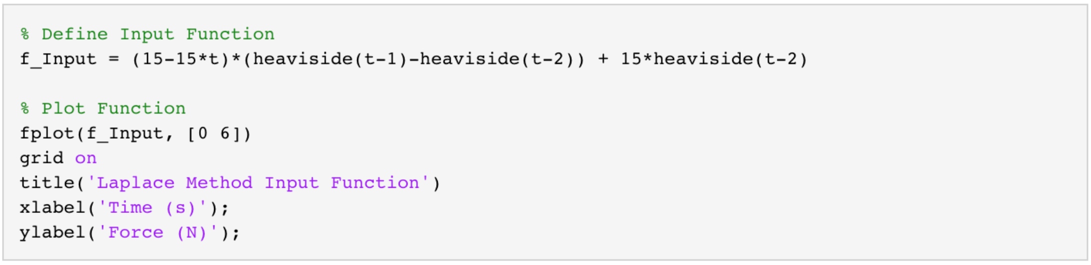
3.1 Input Force for Laplace (2 points)

Using the Heaviside function to restrict the expression of functions to sections of the t-domain, we can define the input force mathematically.

Mathematical Method:

$$f(t) = (15 - 15t) * (H(t-1) - H(t-2)) + 15H(t-2)$$

Using MATLAB, we can plot this function and compare it to the original plot to confirm our model is accurate.



```
% Define Input Function
f_Input = (15-15*t)*(heaviside(t-1)-heaviside(t-2)) + 15*heaviside(t-2)

% Plot Function
fplot(f_Input, [0 6])
grid on
title('Laplace Method Input Function')
xlabel('Time (s)');
ylabel('Force (N)');
```

Figure 4 - MATLAB code for input function plot

This plot accurately represents the given input and as such we can continue with the mathematical model.

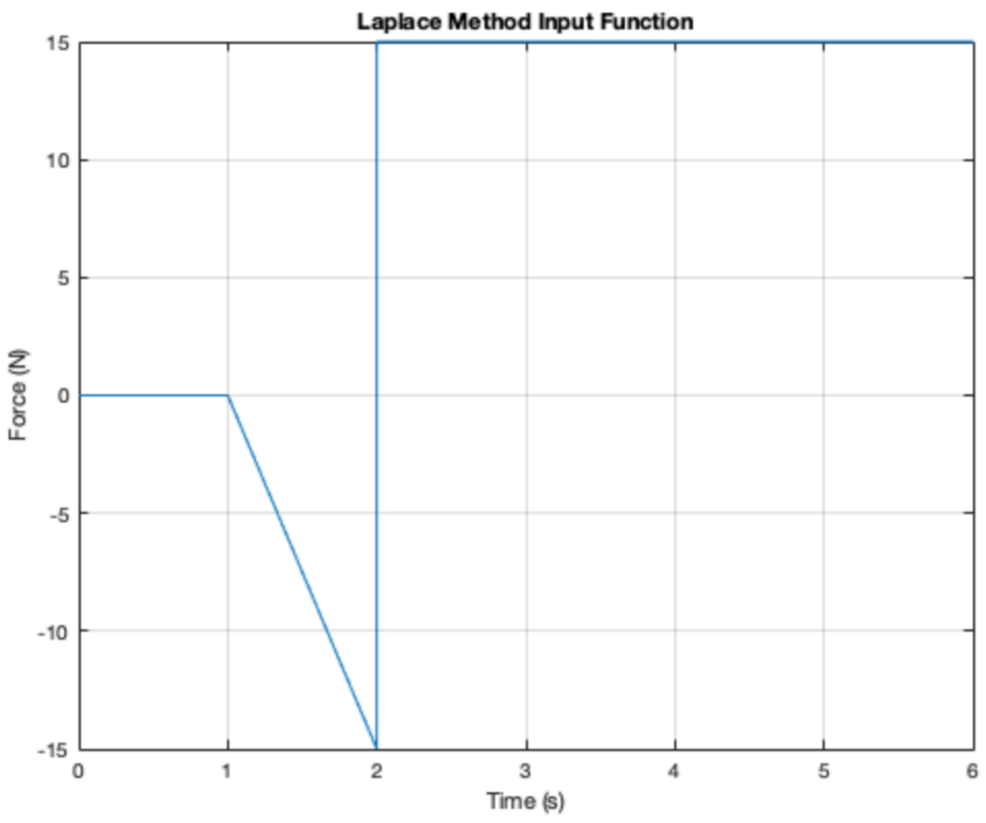


Figure 5 – Plot of input function for Laplace method

3.2 Laplace transform and Inverse Laplace Transform (5 points)

Taking the Laplace of the ode converting the function to the s-domain, initial conditions are assumed to be zero.

Take the Laplace of the equation

Initial Conditions:

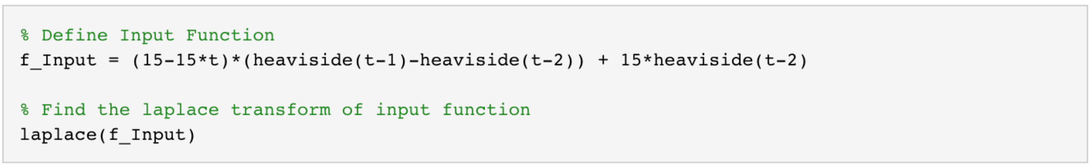
$$x(0) = 0$$
$$x'(0) = 0$$

Laplace of output part of Function:

$$s^2X(s) + 22.5sX(s) + 994.74X(s) = F(s)$$

Taking the Laplace of the Input function, this is done with MATLAB due to the added complexity.

Laplace of Input part of Function:



```
% Define Input Function
f_Input = (15-15*t)*(heaviside(t-1)-heaviside(t-2)) + 15*heaviside(t-2)

% Find the laplace transform of input function
laplace(f_Input)
```

Figure 6 - MATLAB code to find Laplace transform of Input Function f(t)

Output:

$$F(s) = \frac{15e^{-2s}}{s} + \frac{15e^{-2s}(s - e^{s} + 1)}{s^2}$$

Simplified:

$$F(s) = \frac{30e^{-2s}s - 15e^{-s} + 15e^{-2s}}{s^2}$$

We can combine the two Laplace results together to get the entire descriptive equation. This will then be subject to some algebraic re-arrangement to solve for X(s) before we convert the function back to the t-domain.

Sub F(s) to obtain complete descriptive equation:

$$s^2X(s) + 22.5sX(s) + 994.74X(s) = \frac{30e^{-2s}s - 15e^{-s} + 15e^{-2s}}{s^2}$$

Solve for X(s):

$$X(s)=\frac{30e^{-2s}s-15e^{-s}+15e^{-2s}}{(s^2+22.5s+994.74)s^2}$$

Inverse Laplace of function X(s):

```
% Find and plot inverse Laplace transform of X(s)

func_total = (30*exp(-2*s)*s-15*exp(-s)+15*exp(-2*s))/((s^(2)+22.5*s+994.74)*s^(2)) %Define Function X(s)

sol = ilaplace(func_total) %Find inverse laplace

fplot(sol, [0,6]) %Plot response to input function
```

Figure 7 – MATLAB code to find the inverse Laplace of X(s) to get solution to ode x(t)

We can then obtain the final function describing the transient response to the input function we are considering.

Calculated function x(t):

```
x(t) =

15*H(t - 2)*((50*t)/49737 + (6250*exp(45/2 - (45*t)/4)*(cos((347271^(1/2)*(t
- 2))/20) - (49441*347271^(1/2)*sin((347271^(1/2)*(t -
2))/20))/26045325))/274863241 - 1676650/824589723) - 15*H(t -
1)*((50*t)/49737 + (6250*exp(45/4 - (45*t)/4)*(cos((347271^(1/2)*(t -
1))/20) - (49441*347271^(1/2)*sin((347271^(1/2)*(t -
1))/20))/26045325))/274863241 - 847700/824589723) - 30*H(t -
2)*((50*exp(45/2 - (45*t)/4)*(cos((347271^(1/2)*(t - 2))/20) +
(75*347271^(1/2)*sin((347271^(1/2)*(t - 2))/20))/115757))/49737 - 50/49737)
```

3.3 Graph (5 points)

Graphing the transient response function calculated above gives the system response to the input function. The response is stable as it settles into a steady state by t > 3. The system also demonstrates some oscillation. Only minor oscillations are seen across the first ramp function. Perhaps due to the magnitude of the input and/or the resolution of the graph.

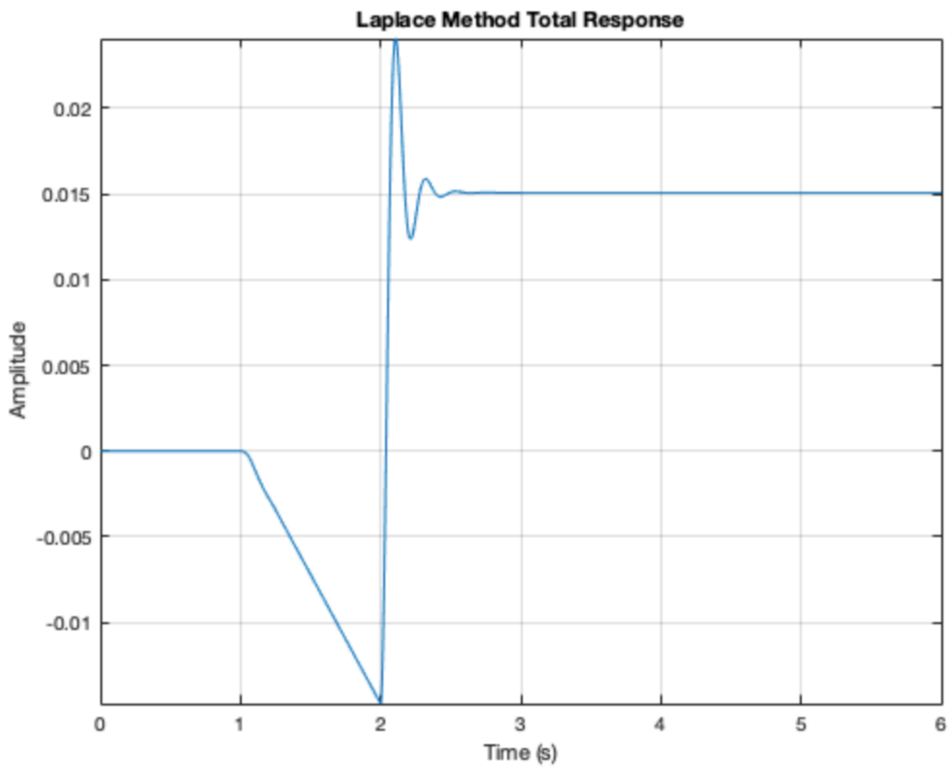


Figure 8 – Transient response to the input function

4 Transient Response via Numerical Convolution (3 points)

Numerical Convolution is a powerful tool can allow for the calculation of the response to any input provided the impulse response can be calculated. It takes advantage of the principals of superposition and linearity to map the input onto the zero-input system response. Essentially describing how the shape of the response can be affected by shape of the input function

Its advantages lie in how effective it is at solving for any input function provided the function can be described in terms of impulse response.

Its disadvantages are its complexity by hand frequently requiring integral calculation and a considerable amount of set up.

Method is discrete due to the conversion of the input into discrete impulse response sections.

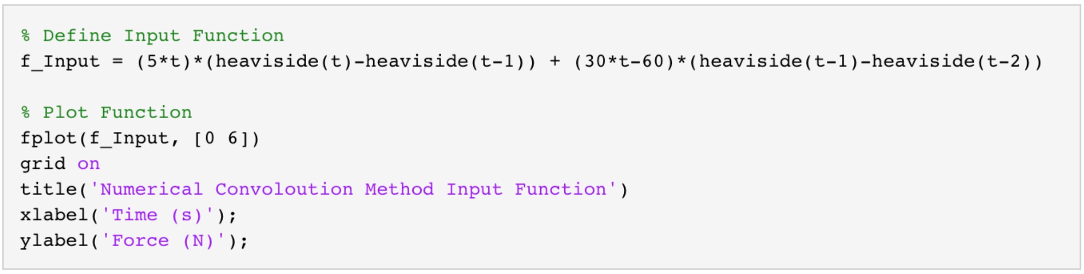
4.1 Input Force for Numerical Convolution (2 points)

Using the Heaviside function to restrict the expression of functions to sections of the t-domain, we can define the input force mathematically.

Mathematical Method:

$$f(t) = (5t) * \big(H(t) - H(t-1)\big) + (30t - 60) * (H(t-1) - H(t-2))$$

Using MATLAB, we can plot this function and compare it to the original plot to confirm our model is accurate.



```
% Define Input Function
f_Input = (5*t)*(heaviside(t)-heaviside(t-1)) + (30*t-60)*(heaviside(t-1)-heaviside(t-2))

% Plot Function
fplot(f_Input, [0 6])
grid on
title('Numerical Convoloution Method Input Function')
xlabel('Time (s)');
ylabel('Force (N)');
```

Figure 9 – MATLAB code for input function plot

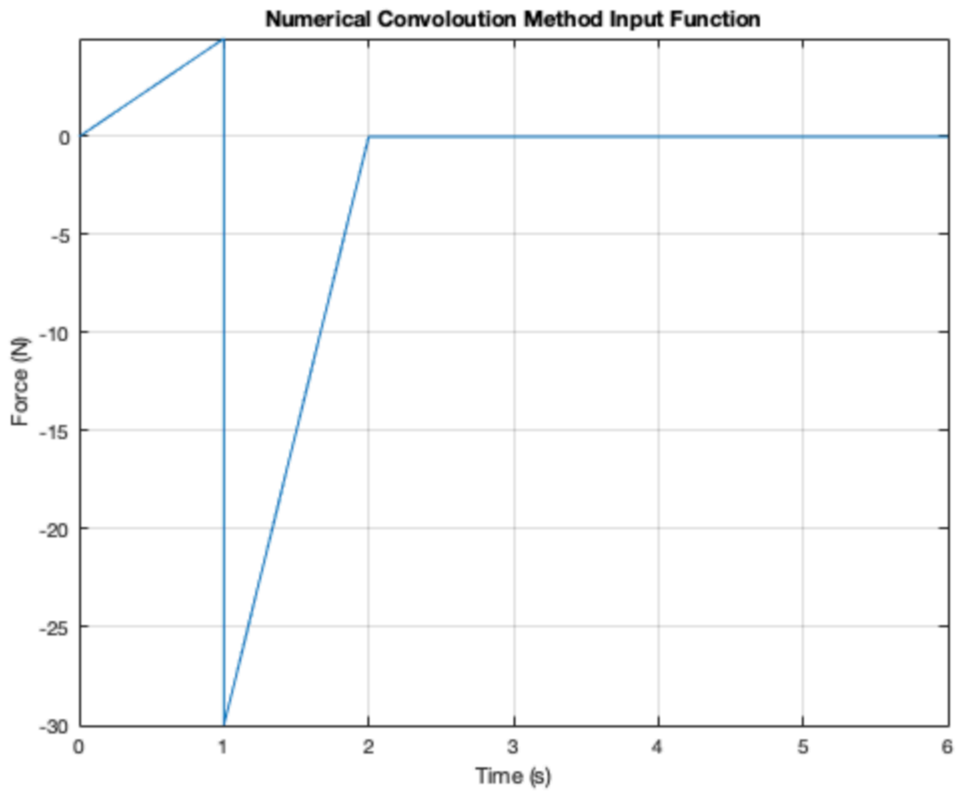


Figure 10 – Input function plot for Numerical Convolution

4.2 Define Unit Impulse response (5 points)

To determine the Unit Impulse response, we first have to determine the damping condition of the system. We found the damping ratio above as described in *Table 2* to be 0.36.

Property	Value
Damping Ratio	0.36
Natural Frequency	31.54 rad/s
Damped frequency	29.46 rad/s

This indicates an underdamped system as such the Unit impulse response is given by:

$$h_{(t)} = \frac{1}{m\omega_n\sqrt{1-\zeta^2}} e^{-\zeta\omega_n t} \sin\left(\omega_n\sqrt{1-\zeta^2}\, t\right)$$

Substituting values from Table 1 and 2. We get:

$$h_t = \frac{1}{38 * 31.54 * \sqrt{1-0.36^2}} e^{0.36*31.34t} \sin\left(31.54 * \sqrt{1-0.36^2 t}\right)$$

Giving:

$$h_t = 8.94 * 10^{-4} * e^{11.28t} \sin(29.43t)$$

RMIT UNIVERSITY

4.3 Program Numerical convolution (5 points)

| DID NOT COMPLETE |

4.4 Graph (5 points)

| DID NOT COMPLETE |

5 Transient Response via MATLAB function ode45 (3 points)

The ode45 approach relies on solving the differential equation directly. As opposed to the Laplace or convolution approach. A system of first order differential equations can be defined that represents the second order system accurately. The MATLAB function can then solve the complete differential equation computationally to arrive at a solution.

The disadvantages of this approach are that it requires a computational device as solving the complete equation by hand would be difficult.

The advantages are the lack of a need to transform the function between domains simply describing the function is all the work needed on our part. The ode45 function takes care of the computation.

Method is continuous

5.1 Input Force for ODE45 (2 points)

Using the Heaviside function to restrict the expression of functions to sections of the t-domain, we can define the input force mathematically.

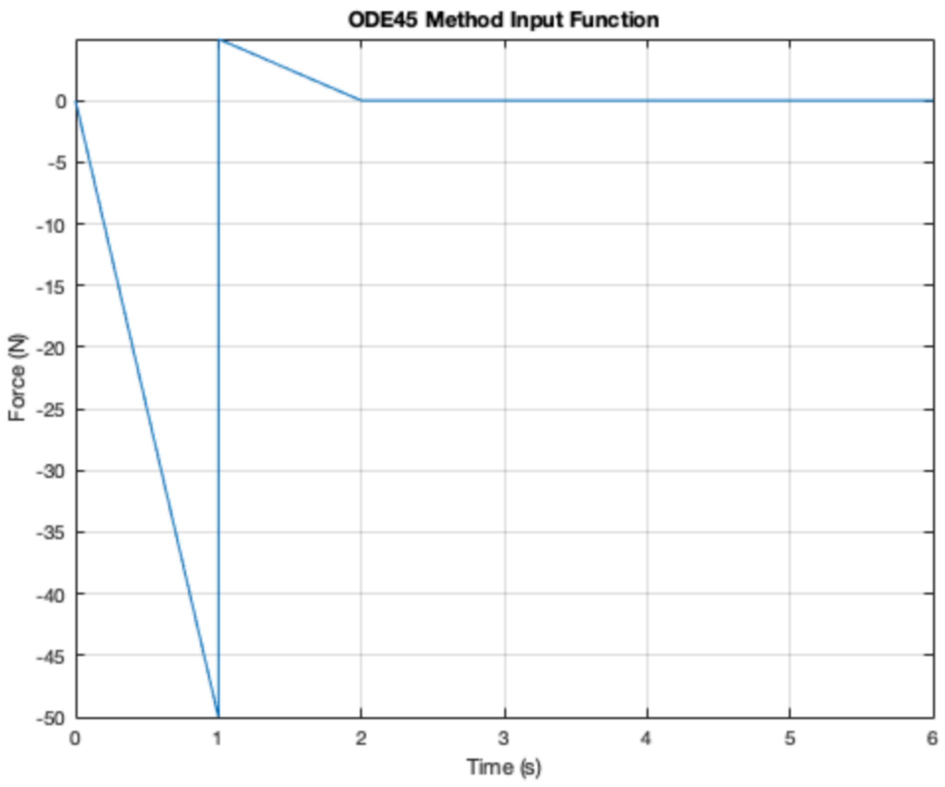
Mathematical Method:

$$f(t) = (-50t) * \big(H(t) - H(t-1)\big) + (-5t + 10) * (H(t-1) - H(t-2))$$

Using MATLAB, we can plot this function and compare it to the original plot to confirm our model is accurate.

```
% Define Input Function
f_Input = (-50*t)*(heaviside(t)-heaviside(t-1)) + (-5*t+10)*(heaviside(t-1)-heaviside(t-2))

% Plot Function
fplot(f_Input, [0 6])
grid on
title('ODE45 Method Input Function')
xlabel('Time (s)');
ylabel('Force (N)');
```



5.2 Solution with ode45 (5 points)

Using the MATLAB function ode45 we can solve the differential equation directly to get a function describing the system response. We first however have to convert the second order equation into a system of first order equations.

Convert the ode to a system of first order equations

$$x_1 = x(t)$$

$$x_1' = x_2$$

$$x'_2 = f(t) - 22.5x_2 - 994.74x_1$$

These can be then be passed to the function ode45 to be solved and the response can be graphed.

```
odesolve()

function odesolve
t=0:0.001:6; %Define the domain

%Initial Condtions
initial_x = 0;
initial_dxdt = 0;

[t,x] =ode45(@rhs, t, [initial_x initial_dxdt]); %Solve ode

plot(t,x(:,1)) %Plot the ode
grid on
title('ODE45 Method Total Response')
xlabel('Time (s)');
ylabel('Amplitude');


    function dxdt=rhs(t,x)
        f_input = (-50*t)*(heaviside(t)-heaviside(t-1)) + (-5*t+10)*(heaviside(t-1)-heaviside(t-2)); % Define the input force equation

        % Convert to a system of first order equations
        dxdt_1 = x(2);
        dxdt_2 = f_input - 22.5*x(2) - 994.74 * x(1);

        dxdt=[dxdt_1; dxdt_2];
    end
end
```

Figure 11 – MATLAB code to solve the set of first order equations using ode45

5.3 Graph (5 points)

The graph shows a stable response with the system entering a steady state after about t > 2. Some oscillations can be seen. Interestingly oscillations are not seen for the first ramp input between t (0 to 1). Perhaps due to the magnitude of the force, which is much larger than in the Laplace section and so may obscure the oscillations.

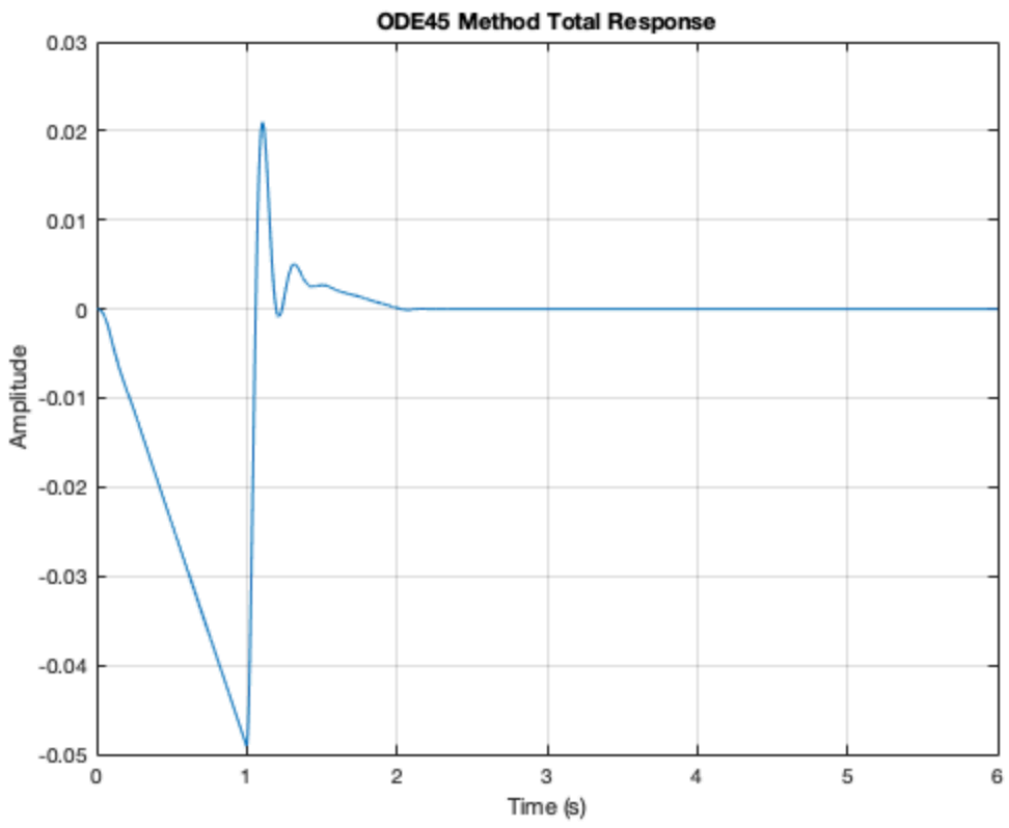


Figure 12 – Graph of the transient response due to the input function

6 Conclusions and recommendations (5 points)

This report considers three methods for assessing a systems response to a given input. These methods are Laplace, Numerical Convolution and ode45. The Laplace and ode45 have the advantage of simplicity regarding calculation effort. Both can be setup relatively quickly and calculated rapidly with the assistance of a computer. The Laplace method has a further advantage in this regard as it can be performed completely by hand if required. The Numerical Convolution method in contrast requires extensive setup but can be solved by hand although with more difficulty in comparison to the Laplace method due to the inclusion of integral calculations.

7 References

Reference list

Desmos 2022, *Desmos Graphing Calculator*, Desmos Graphing Calculator.

Plot expression or function - MATLAB fplot - MathWorks Australia n.d., au.mathworks.com, viewed 8 April 2022, <https://au.mathworks.com/help/matlab/ref/fplot.html>.

Pole and Zero Locations - MATLAB & Simulink - MathWorks Australia n.d., au.mathworks.com, viewed 8 April 2022, <https://au.mathworks.com/help/control/ug/pole-and-zero-locations.html>.

Solve nonstiff differential equations — medium order method - MATLAB ode45 - MathWorks Australia n.d., au.mathworks.com.

Symbolab Math Solver - Step by Step calculator 2017, Symbolab.com.

x-engineer.org n.d., *How to find the transfer function of a system – x-engineer.org*, X Engineer.