**System Dynamics**

**Fall 2017**

**Project**



**Department of Engineering**

**Indiana University Purdue University – Fort Wayne**

**By: Nguyen Vo**

**Date: 06 December 2017**

**To: Dr. Kang**

**Question 1**

Given:

Taking the Laplace Transform L[

%% Question 1 - Nguyen Vo

% Finding the closed-form solution

syms s; % Define s as a symbolic variable

X=((6/(s^2+9))+s+3.5)/(s^2+4\*s+6); % X(s)

x=ilaplace(X); % Inverse Laplace to find x(t)

pretty(x) % Simplify x(t)

Answer:

**Question 2**

%% Question 2 - Nguyen Vo

% Transfer function of 1/(s^2 + 4s + 9)

num=[1]; % numerator of G(s)

den=[1 4 9]; % denominator of G(s)

sys=tf(num,den); % G(s)

%% Part i - tf() and impulse()

% Impulse function with input of 5\*Dirac\_Delta\_t

[x t]=impulse(sys); % MATLAB will determine time steps and span

% Plotting of x

figure(1);

plot(t,5\*x); % 5\*x is 5\*Dirac\_Delta\_t

legend('x(t)');

xlabel('Time, t [s]');

ylabel('Response, x [length]');

grid on;



Figure 1. Response of the system as a function of time *t*

%% Part ii - tf() and step()

% Step function

t=0:0.01:10;

[x,tt]=step(sys,t);

% Plotting of x

figure(2);

plot(tt,5\*x);

legend('x(t)');

xlabel('Time, t [s]');

ylabel('Response, x [length]');

grid on;



Figure 2. Response of the system as a function of time *t*

%% Part iii - tf() and lsim()

t=0:0.01:10; % time span

% The following two lines create the input function

unitstep=@(x)(x>=0); % Define the unit step function. True is 1, False is 0

u=5\*t.\*(1-unitstep(t-2))+2\*sin(6\*t); % The input function

% Solve for the output given the input function using lsim()

x=lsim(sys,u,t);

% Plotting of x

figure(3);

plot(t,x);

legend('x(t)');

xlabel('Time, t [s]');

ylabel('Response, x [length]');

grid on;

****

Figure 3. Response of the system as a function of time *t*

**Question 3**

Deriving from the problem:

**Taking the Laplace Transform:**

Transfer Function:

%% Question 3 - Nguyen Vo

%% Part i

% Transfer function of (7s + 5)/(2s^2 + 7s + 20) with input of 5u(t)

num=[7 5]; % numerator of G(s)

den=[2 7 20]; % denominator of G(s)

sys=tf(num,den); % G(s)

[x,tt]=step(sys,t); % Matlab will determine time steps for accuracy

% Plotting of x

figure(1);

plot(tt,5\*x);

legend('x(t)');

xlabel('Time t [s]');

ylabel('Response x [length]');

grid on;



Figure 4. Response of the mass *m (m=2)* as a function of time *t*

%% Part ii

t=0:0.01:10; % Time Span

% The following two lines create the input function

unitstep=@(x)(x>=0); % Define the unit step function. True is 1, False is 0

u=t.\*(1-unitstep(t-1))+(-t+2).\*(unitstep(t-1)-unitstep(t-3))+(t-4).\*(unitstep(t-3)-unitstep(t-4)); % The input

% Solve for the output given the input function using lsim()

x=lsim(sys,u,t);

% Plotting of x

figure(2);

plot(t,x,t,u);

legend('x(t)','Input u(t)');

xlabel('Time t [s]');

ylabel('Response x [length]');

grid on;

****

Figure 5. Response of the mass *m (m=2)* as a function of time *t*

**Question 4**

%% Question 4 - Nguyen Vo

% Transfer function of s/(2s^2 + 4s + 10) with input of u(t)

num=[1 0]; % numerator of G(s)

den=[2 4 10]; % denominator of G(s)

sys=tf(num,den); % G(s)

% Step function

[x,tt]=step(sys,t); % Matlab will determine time steps for accuracy

% Plotting of x

figure(1);

plot(tt,x);

legend('x');

xlabel('Time t [s]');

ylabel('Response x [length]');

grid on;



Figure 6. Response of the mass *m* as a function of time *t*

**Question 5**

%% Question 5 – Nguyen Vo

%% Part i - expm() and int()

% Matrices

A=[0 1;-10 -4]; % State Matrix A

B=[0;5]; % Input Matrix B

C=[1 0]; % Output Matrix C

x0=[1;0.5]; % Initial Condition

% Finding the closed-form solution

syms t tau

stm1=expm(A\*t); % State transition matrix

stm2=subs(stm1,t,t-tau); % Replaced t with (t-tau)

y=stm1\*x0 + int((stm2\*B\*sin(2\*tau)),tau,0,t); % 2x1 matrix solution

x=C\*y; % Final solution of x

pretty(x) % Simplify x(t)

x1 = subs(x);

Answer:

%% Part ii - ode45()

% Define the function matrix with z-variables and z-values

zdot=@(t,z)([ z(2);

-10\*z(1)-4\*z(2)+5\*sin(2\*t)]);

% Define Time Span, ICs, and Relative Errors/Tolerance

tspan=[0 7]; % Time Span, Time Steps will be determined by Matlab

z0=[1;0.5]; % Initial Conditions

options=odeset('reltol',1e-6,'abstol',1e-8); % Set Solver Options (optional)

% Solving the non-stiff ode

[t sol]=ode45(zdot,tspan,z0,options); % Function, Time span, ICs, Tolerance

% Data extraction

x2=sol(:,1); % Extract x

%% Part iii - ss() and lsim()

t3=0:0.01:7; % Time span and fixed time step, have to specify time steps to use lsim

x0=[1;0.5]; % Initial Conditions

A=[0 1;-10 -4]; % State Matrix A

B=[0;5]; % Input Matrix B

C=[1 0]; % Output Matrix C

D=0; % Direct Transmission Matrix D

unitstep=@(x)(x>=0); % Define the Unit Step Function

u=sin(2\*t3); % Input function

% lsim() function

x=lsim(ss(A,B,C,D),u,t3,x0);

x3=x(:,1); % extract x

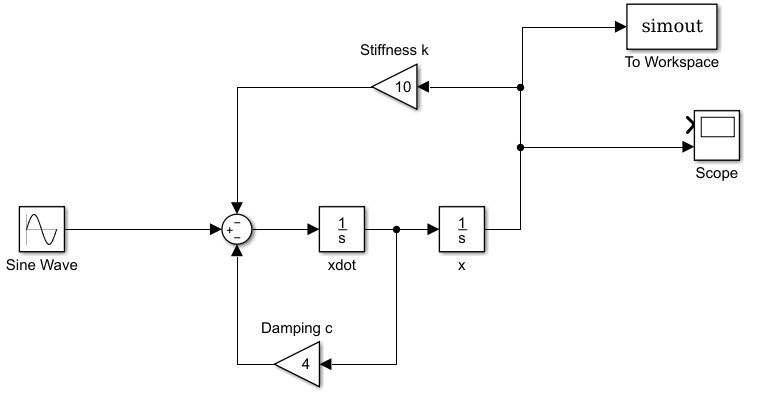


Figure 7. Simulink Setup

%% Part v - Plotting of x with intentional different heights for comparison

figure(1);

fplot(x1,[0 7]);

hold on;

plot(t,x2+0.25,t3,x3+0.5,tout,simout+0.75); % Add value (+0.25, etc.) to see the plots better

hold off;

legend('expm() and int()','ode45()','ss() and lsim()','simulink');

xlabel('Time t [s]');

ylabel('Response x [l]');

grid on;

****

Figure 8. Responses of the system with multiple testing methods as a function of time *t*

**Question 6**

(1)

(2)

Let z1 = ia

Z2 =

We will have (with :

%% Question 6 – Nguyen Vo

% Define constants

KT=0.05;

Kb=KT;

La=2e-3;

Ra=0.5;

IM=9e-5;

c=1e-4;

% Define the function matrix with z-variables and z-values

zdot=@(t,z)([ -Ra/La\*z(1)-Kb/La\*z(2)+10/La;

KT/IM\*z(1)-c/IM\*z(2)]);

% Define Time Span, ICs, and Relative Erros/Tolerance

tspan=[0 0.2]; % Time Span, Time Steps will be determined by Matlab

options=odeset('reltol',1e-6,'abstol',1e-8); % Set Solver Options (optional)

% Solving the non-stiff ode

[t sol]=ode45(zdot,tspan,[0;0],options); % Function, Time span, ICs, Tolerance

% Data extraction

current=sol(:,1); % Extract current

omega=sol(:,2); % Extract angular velocity

% Plot results

figure(1);

subplot(2,1,1);

plot(t,KT\*current); % Torque = KT \* Current

grid on;

xlabel('Time, t [sec]');

ylabel('Torque, T [N-m]');

subplot(2,1,2);

plot(t,omega);

grid on;

xlabel('Time, t [sec]');

ylabel('Angular Velocity, \omega [rad/s]');

****

Figure 9. Response of the Torque *T* and Angular Velocity as a function of time *t*

**Question 7**

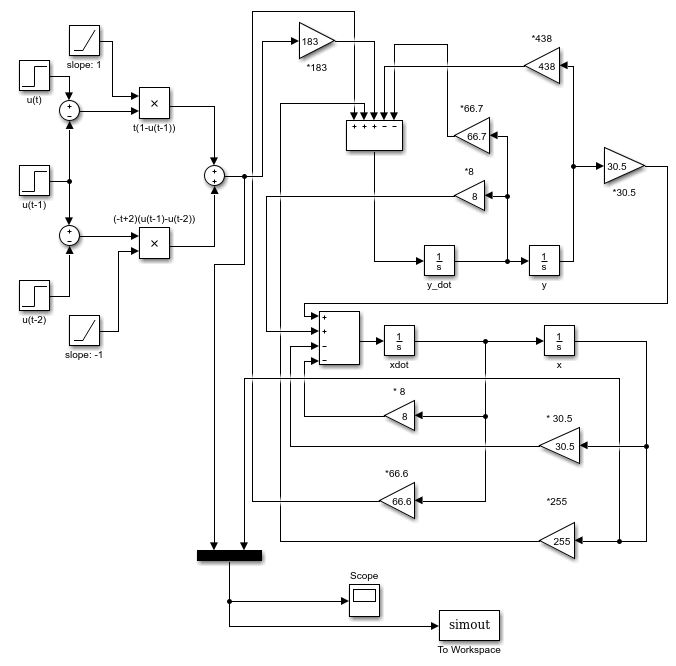
****

Figure 10. Simulink Setup

%% Question 7 - Nguyen Vo

%% Plotting the response of the vehicle's body

figure(1);

plot(tout,simout);

legend('Input','Simulink Response');

xlabel('Time t [s]');

ylabel('Response x [length]');

grid on;



Figure 11. Response of the vehicle’s body as a function of time *t*

**Question 8**

%% Question 8 - Nguyen Vo

%% Part i

% Finding the closed-form solution

syms s; % Define s as a symbolic variable

X=(s+exp(-s)\*3\*s+exp(-3\*s))/(s\*(s+3)\*(s+1));

x=ilaplace(X);

pretty(x)

x1=subs(x);

Answer**:**

%% Part ii - ss() and lsim()

% Transfer function of 1/(s^2+4s+3) with input of

% Dirac\_Delta\_t + 3\*Dirac\_Delta\_(t-1) +u(t-3)

num=[1]; % numerator of G(s)

den=[1 4 3]; % denominator of G(s)

sys=tf(num,den); % G(s)

t=0:0.01:7; % Time Span

% The following two lines create the input function

unitstep=@(x)(x>=0); % Define the unit step function. True is 1, False is 0

deltat=0.01;

u=(unitstep(t)-unitstep(t-deltat))/deltat+3\*(unitstep(t-1)-unitstep(t-1-deltat))/deltat+unitstep(t-3);

% The input(unitstep(t-t0)-unistep(t-t0-deltat))/deltat is the numerical definition of dirac delta function

% Solve for the output given the input function using lsim()

x2=lsim(sys,u,t);

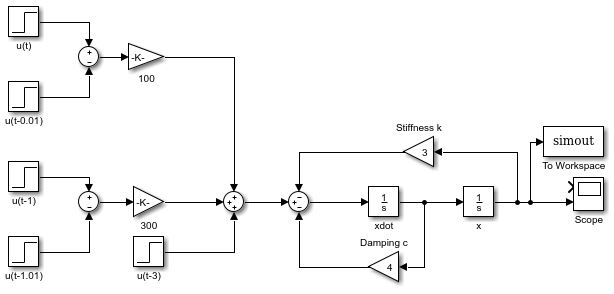
****

Figure 12. Simulink Setup

%% Part iv - Plotting of x with intentional different heights for comparison

figure(1);

plot(t,x1,t,x2+0.25,tout,simout+0.5); %% Add heights to see the graphs better

legend('ilaplace()','tf() and lsim()','simulink');

xlabel('Time t [s]');

ylabel('Response x [m]');

grid on;

****

Figure 12. Responses of the system with multiple testing methods as a function of time *t*

**Question 9**

With linearization, the problem becomes:

With a = 0.12

With a)

With b)

%% Question 9 - Nguyen Vo

a=0.1^2; % Define a

% Define the function matrix with z-variables and z-values

zdot=@(t,z)([z(2);-a\*sin(z(1))]);

%% Part a - theta(0)=0.1 and theta\_dot(0)=0.1

tspan=[0 50]; % Time span

z0=[0.1;0.1]; % Initial Conditions of a)

options=odeset('reltol',1e-6,'abstol',1e-8); % Set Solver Options

% Solving the ode

[t sol]=ode45(zdot,tspan,z0,options);

th=sol(:,1); % Extract theta(non-linear)

% Setting up the linear ode solution

tl=0:0.001:50;

th\_linear=sin(0.1\*tl)+0.1\*cos(0.1\*tl);

% Plotting of theta (linear and exact)

figure(1);

plot(tl,th\_linear,t,th);

legend('Linear','Nonlinear');

xlabel('time t [sec]');

ylabel('\theta [rad]');

grid on;



Figure 13. Response of the pendulum system as a function of time *t*

%% Part b - theta(0)=1 and theta\_dot(0)=1

tspan=[0 50]; % Time span

z0=[1;1]; % Initial Conditions of b)

options=odeset('reltol',1e-6,'abstol',1e-8); % Set solver options

% Solving the ode

[t sol]=ode45(zdot,tspan,z0,options);

th=sol(:,1); % Extract theta(non-linear)

% Setting up the linear ode solution

tl=0:0.001:50;

th\_linear=10\*sin(0.1\*tl)+cos(0.1\*tl);

% Plotting of theta (linear and exact)

figure(2);

plot(tl,th\_linear,t,th);

legend('Linear','Nonlinear');

xlabel('time t [sec]');

ylabel('\theta [rad]');

grid on;



Figure 14. Response of the pendulum system as a function of time *t*

**Discussion:**

It can be seen that the linear approximate solution matches well with the nonlinear exact solution as long as θ is smaller than about 1 radian for a) and 5 radians for b). Also, the larger the initial conditions, the more accurate and longer the accuracy will remain for the linear approximate solutions comparting to the nonlinear exact solutions.

**Question 10**

Given:

Or

With k = 1500, m = 100, , g = 9.81, the problem becomes:

From this, the Simulink setup can be created:

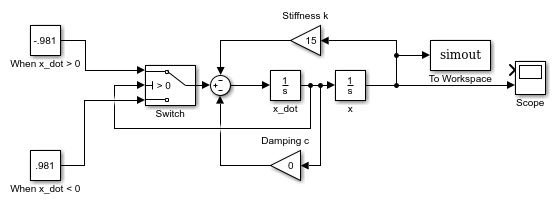


Figure 15. Simulink Setup

%% Question 10 - Nguyen Vo

%% Plotting the response of the mass

figure(1);

plot(tout,simout);

legend('Simulink Response');

xlabel('Time t [s]');

ylabel('Response x [length]');

grid on;



Figure 11. Response of the mass *m* as a function of time *t*

**Discussion:**

It took 4 seconds for the vibration to die out.