**Control 01 ECUACIONES DIFERENCIALES**

Parte 1 (70%)

Resolver las siguientes ecuaciones diferenciales de primer orden con condiciones iniciales.

* La ecuación se debe colocar en su forma canónica
* Se debe determinar la solución homogénea de forma analítica
* Se debe determinar la solución particular de forma analítica
* Se debe a partir de la condición inicial las constantes de forma analítica si corresponde

## Ecuación Diferencial 1.1 (10pts)

## Ecuación Diferencial 1.2 (10pts)

## Ecuación Diferencial 1.3 (10pts)

## Ecuación Diferencial 1.4 (10pts)

Parte 2 (15%)

## Serie de Fourier 2.1 (10pts)

Resolver la Series de Fourier Compleja de forma analítica escribir el procedimiento y graficar con un programa de Octave/MATLAB

## Serie de Fourier 2.2 (10pts)

Graficar el Resultado de la solución para dada por la ecuación con un programa de Octave/MATLAB

Parte 3 (15%)

## Ecuación Diferencial 3.1 (10pts)

Resuelva usando diferencias centrales y grafique los resultados incluyendo, desplazamiento, velocidad y diagrama de fase

%matriz de masa

M = [1, 1, 0, 0, 0;

1, 2, 1, 0, 0;

0, 1, 3, 1, 0;

0, 0, 1, 2, 1;

0, 0, 0, 1, 1];

%matriz de rigidez

K = [ 400,-200, 0, 0, 0;

-200, 400,-200, 0, 0;

0,-200, 400,-200, 0;

0, 0,-200, 400,-200;

0, 0, 0,-200, 200];

%matriz de amortiguamiento

C = [20, -10, 0, 0, 0;

-10, 20, -10, 0, 0;

0, -10, 20, -10, 0;

0, 0, -10, 20, -10;

0, 0, 0, -10, 10];

%% condiciones iniciales

x0 = [0;0;0;0;0]; %desplazamiento inicial

v0 = [0;0;0;0;0]; %velocidad inicial

%componentes

f1(1:N) = 0; %inicialización de fuerza

f2(1:N) = 0; %inicialización de fuerza

f3(1:N) = 0; %inicialización de fuerza

f4(1:N) = 0; %inicialización de fuerza

f5(1:N) = 0; %inicialización de fuerza

tf1 = 10; %tiempo1

tf2 = 50; %tiempo2

tf3 = 90; %tiempo2

nf1 = round(tf1/dt); %muestra1

nf2 = round(tf2/dt); %muestra2

nf3 = round(tf3/dt); %muestra2

f1(nf1:nf2) = 50;

f1(nf2+1:nf3) = -50;

## Ecuación Diferencial 3.2 (10pts)

Resuelva la ecuación diferencial ordinaria de segundo orden no lineal, con valores iniciales mediante el método de Euler, incluyendo, desplazamiento, velocidad, diagrama de fase y espacio de fase

%datos

delta = 1.0;

beta = 1.0;

w0 = -1.0;

gama = 1.0;

phi0 = 0.0;

w = 0.1;

%tiempo incial/final

tini = 0;

tfin = 200;

## Ecuación Diferencial 1.1 (10pts)

Ecuación Homogénea – Solución Complementaria

Solución Particular

Solución Completa

Condición Inicial

Finalmente

## Ecuación Diferencial 1.2 (10pts)

Ecuación Homogénea – Solución Complementaria

Solución Particular

En nuestro caso podemos aislar un término genérico e intercambiar la integral por la sumatoria

Solución completa

Condiciones Iniciales

Solución

## Ecuación Diferencial 1.3 (10pts)

Ecuación Homogénea – Solución Particular

Polinomio característico

Raíces

Solución Complementaria – Método de los coeficientes Indeterminados

Derivamos

Reemplazamos

Solución Completa

Derivadas

Condiciones Iniciales

Ordenamos

Sistema de Ecuaciones

Finalmente

## Ecuación Diferencial 1.4 (10pts)

Ecuación Homogénea – Solución Complementaria

Polinomio característico

Raíces

Solución Complementaria

Solución Particular

Condiciones Iniciales

Sistema de Ecuaciones

Parte 2 (15%)

## Serie de Fourier 2.1 (10pts)

Resolver la Series de Fourier Compleja de forma analítica escribir el procedimiento y graficar con un programa de Octave/MATLAB

%% control 01 parte serie de fourier compleja 2.1

%

% f(t) = (t^4 -1) -1 <= t <= 1

%

clearvars;

close all;

clc;

%% definiciones

Tmin = -10;

Tmax = 10;

dt = 0.05;

%%tiempo

t = Tmin:dt:Tmax;

Nt = length(t);

%%función

f2t = zeros(1,Nt);

%% serie de fourier

figure(1)

plot(t,f2t);

title('serie de fourier');

xlabel('t (s)');

ylabel('f(t) (Pa)');

grid on;

box on;

%sumatoria

N = 201;

for n = 1:N

nn(n) = -round(N/2) + n;

%coeficiente

%valor c(n)

if nn(n) ~= 0

cn(n) = (4/(nn(n)^4\*pi^4 ))\*(-1)^nn(n)\*(nn(n)^2\*pi^2 - 6 );

else

cn(n) = -8/5;

end

%funcion

f2t = f2t + cn(n)\*exp(j\*nn(n)\*t);

figure(1)

plot(t,f2t);

title('serie de fourier');

xlabel('t (s)');

ylabel('f(t) (Pa)');

legend(['f(t) n = ',num2str(nn(n))]);

axis([Tmin,Tmax,-2,-0.5]);

grid on;

box on;

pause(0.05);

end

%% figura final

figure(2)

plot(t,f2t);

title('serie de fourier');

xlabel('t (s)');

ylabel('f\_2(t) (Pa)');

legend(['f\_2(t) n = ',num2str(nn(n))]);

axis([Tmin,Tmax,-2,-0.5]);

grid on;

box on;

%% figura final parte real e imaginaria

figure(3)

subplot(2,1,1)

plot(t,real(f2t));

title('parte real f\_2(t)');

xlabel('t');

ylabel('real(f\_2(t))');

legend('real(f\_2(t))');

axis([Tmin,Tmax,-2,-0.5]);

grid on;

box on;

subplot(2,1,2)

plot(t,imag(f2t));

title('parte imaginaria f\_2(t)');

xlabel('n');

ylabel('imag(f\_2(t))');

legend('imag(f\_2(t))');

axis([Tmin,Tmax,1.1\*min(imag(f2t)),1.1\*max(imag(f2t))]);

grid on;

box on;

%% figura coeficientes

figure(4)

subplot(2,1,1)

stem(nn,real(cn));

title('coeficientes parte realc\_n');

xlabel('n');

ylabel('real(c\_n)');

legend('real(c\_n)');

grid on;

box on;

subplot(2,1,2)

stem(nn,imag(cn));

title('coeficientes parte imaginaria c\_n');

xlabel('n');

ylabel('imag(c\_n)');

legend('imag(c\_n)');

grid on;

box on;

figure(5)

subplot(2,1,1)

stem(nn,abs(cn));

title('amplitud c\_n');

xlabel('n');

ylabel('abs(c\_n)');

legend('abs(c\_n)');

grid on;

box on;

subplot(2,1,2)

stem(nn,angle(cn));

title('fase c\_n');

xlabel('n');

ylabel('fase(c\_n)');

legend('fase(c\_n)');

grid on;

box on;









## Serie de Fourier 2.2 (10pts)

Graficar el Resultado de la solución para dada por la ecuación con un programa de Octave/MATLAB

Solución

%% control 01 parte serie de fourier 2.2

clearvars;

close all;

clc;

%% definiciones

Tmin = 0;

Tmax = 6\*pi;

dt = 0.01\*pi;

%%tiempo

t = Tmin:dt:Tmax;

Nt = length(t);

%%función

f2t = zeros(1,Nt);

y2t = zeros(1,Nt);

%% serie de fourier

figure(1)

plot(t,f2t);

title('serie de fourier');

xlabel('t (s)');

ylabel('f(t) (Pa)');

grid on;

box on;

%sumatoria

N = 50;

for n = 1:N

nn(n) = n;

%coeficiente f(t)

fn(n) = 4/(pi\*(2\*nn(n) - 1));

%funcion f(t)

f2t = f2t + fn(n)\*sin( (2\*nn(n) - 1)\*t );

%coeficiente y(t)

yn(n) = 4/(pi\*(2\*nn(n)-1));

yn(n) = yn(n)/( 4\*nn(n)^2-4\*nn(n)+16\*pi^2+1 );

%funcion y(t)

y2t = y2t + yn(n)\*( 4\*pi\*sin((2\*nn(n)-1)\*t)+(1-2\*nn(n))\*cos((1-2\*nn(n))\*t) );

figure(1)

plot(t,f2t);

title('serie de fourier');

xlabel('t (s)');

ylabel('f(t) (Pa)');

legend(['f(t) n = ',num2str(nn(n))]);

axis([Tmin,Tmax,-1.5,1.5]);

grid on;

box on;

figure(2)

plot(t,y2t);

title('serie de fourier');

xlabel('t (s)');

ylabel('y(t) (Pa)');

legend(['y(t) n = ',num2str(nn(n))]);

axis([Tmin,Tmax,-0.1,0.1]);

grid on;

box on;

pause(0.05);

end

%% figura final

figure(3)

plot(t,f2t);

title('serie de fourier');

xlabel('t (s)');

ylabel('f\_2(t) (Pa)');

legend(['f\_2(t) n = ',num2str(nn(n))]);

axis([Tmin,Tmax,-1.5,1.5]);

grid on;

box on;

figure(4)

plot(t,y2t);

title('serie de fourier');

xlabel('t (s)');

ylabel('y(t) (Pa)');

legend(['y(t) n = ',num2str(nn(n))]);

axis([Tmin,Tmax,-0.1,0.1]);

grid on;

box on;





Parte 3 (15%)

## Ecuación Diferencial 3.1 (10pts)

Resuelva usando diferencias centrales y grafique los resultados incluyendo, desplazamiento, velocidad y diagrama de fase

%matriz de masa

M = [1, 1, 0, 0, 0;

1, 2, 1, 0, 0;

0, 1, 3, 1, 0;

0, 0, 1, 2, 1;

0, 0, 0, 1, 1];

%matriz de rigidez

K = [ 400,-200, 0, 0, 0;

-200, 400,-200, 0, 0;

0,-200, 400,-200, 0;

0, 0,-200, 400,-200;

0, 0, 0,-200, 200];

%matriz de amortiguamiento

C = [20, -10, 0, 0, 0;

-10, 20, -10, 0, 0;

0, -10, 20, -10, 0;

0, 0, -10, 20, -10;

0, 0, 0, -10, 10];

%% condiciones iniciales

x0 = [0;0;0;0;0]; %desplazamiento inicial

v0 = [0;0;0;0;0]; %velocidad inicial

%componentes

f1(1:N) = 0; %inicialización de fuerza

f2(1:N) = 0; %inicialización de fuerza

f3(1:N) = 0; %inicialización de fuerza

f4(1:N) = 0; %inicialización de fuerza

f5(1:N) = 0; %inicialización de fuerza

tf1 = 10; %tiempo1

tf2 = 50; %tiempo2

tf3 = 90; %tiempo2

nf1 = round(tf1/dt); %muestra1

nf2 = round(tf2/dt); %muestra2

nf3 = round(tf3/dt); %muestra2

f1(nf1:nf2) = 50;

f1(nf2+1:nf3) = -50;

%% prog metodo de diferencias centrales control 01 parte 3.1

%este programa resuelve la EDO de segundo orden con condiciones inciales

%usando diferencias finitas

clearvars;

close all;

clc;

%% datos inciales

%matriz de masa

M = [1, 1, 0, 0, 0;

1, 2, 1, 0, 0;

0, 1, 3, 1, 0;

0, 0, 1, 2, 1;

0, 0, 0, 1, 1];

%matriz de rigidez

K = [ 400,-200, 0, 0, 0;

-200, 400,-200, 0, 0;

0,-200, 400,-200, 0;

0, 0,-200, 400,-200;

0, 0, 0,-200, 200];

%matriz de amortiguamiento

C = [20, -10, 0, 0, 0;

-10, 20, -10, 0, 0;

0, -10, 20, -10, 0;

0, 0, -10, 20, -10;

0, 0, 0, -10, 10];

%% tiempo

dt = 0.01; %intervalo de muestreo

tini = 0; %tiempo incial

tfin = 200; %tiempo final

t = tini:dt:tfin; %vector de tiempo

N = length(t); %numero de muestras total

%% condiciones inciales

%% condiciones iniciales

x0 = [0;0;0;0;0]; %desplazamiento inicial

v0 = [0;0;0;0;0]; %velocidad inicial

%% fuerza externa

%componentes

f1(1:N) = 0; %inicialización de fuerza

f2(1:N) = 0; %inicialización de fuerza

f3(1:N) = 0; %inicialización de fuerza

f4(1:N) = 0; %inicialización de fuerza

f5(1:N) = 0; %inicialización de fuerza

tf1 = 10; %tiempo1

tf2 = 50; %tiempo2

tf3 = 90; %tiempo2

nf1 = round(tf1/dt); %muestra1

nf2 = round(tf2/dt); %muestra2

nf3 = round(tf3/dt); %muestra2

f1(nf1:nf2) = 50;

f1(nf2+1:nf3) = -50;

f = [f1;f2;f3;f4;f5];

%% desplazamiento

%inicializamos el desplazamiento

X = zeros(5,N);

%paso 1 determinar la aceleracion

f0 = f(:,1);

a0 = M\(f0 - x0 - v0);

%paso 2 determinar el desplazamiento x1 ubicado en la segunda columna de X

X(:,2) = x0 + v0\*dt + (1/2)\*a0\*dt^2;

%paso 3 determinamos el resultado mediante recurrencia

X(:,1) = x0;

C1 = M/(dt^2) + C/(2\*dt);

C2 = K - (2\*M)/(dt^2);

C3 = M/(dt^2) - C/(2\*dt);

for n = 2:N-1

X(:,n+1) = C1\( f(:,n) - C2\*X(:,n) - C3\*X(:,n-1) );

end;

%separamos la solucion por facilidad

x1(1:N) = 0; %incializacion de desplazamiento

x2(1:N) = 0; %incializacion de desplazamiento

x3(1:N) = 0; %incializacion de desplazamiento

x4(1:N) = 0; %incializacion de desplazamiento

x5(1:N) = 0; %incializacion de desplazamiento

for n = 1:N

x1(n) = X(1,n);

x2(n) = X(2,n);

x3(n) = X(3,n);

x4(n) = X(4,n);

x5(n) = X(5,n);

end;

%% velocidad

v1(1:N) = 0; %incializacion de la velocidad

v2(1:N) = 0; %incializacion de la velocidad

v3(1:N) = 0; %incializacion de la velocidad

v4(1:N) = 0; %incializacion de la velocidad

v5(1:N) = 0; %incializacion de la velocidad

for n = 2:N-1

v1(n) = ( x1(n+1) - x1(n-1) )/(2\*dt);

v2(n) = ( x2(n+1) - x2(n-1) )/(2\*dt);

v3(n) = ( x3(n+1) - x3(n-1) )/(2\*dt);

v4(n) = ( x4(n+1) - x4(n-1) )/(2\*dt);

v5(n) = ( x5(n+1) - x5(n-1) )/(2\*dt);

end;

V = [v1;v2;v3;v4;v5];

%% resultados

figure(1)

plot(t,f1,'b',t,f2,'k',t,f3,'r',t,f4,'g',t,f5,'c');

title('fuerza');

xlabel('tiempo t(s)');

ylabel('fuerza f(t) (N)');

legend('f\_1(t)','f\_2(t)','f\_3(t)','f\_4(t)','f\_5(t)');

axis([tini, tfin, -1.1\*max(abs(f1)), 1.1\*max(abs(f1))]);

grid on;

box on;

figure(2)

plot(t,x1,'b',t,x2,'k',t,x3,'r',t,x4,'g',t,x5,'c');

title('desplazamiento diferencias centrales');

xlabel('tiempo t(s)');

ylabel('desplazamiento x(t) (m)');

legend('x\_1(t)','x\_2(t)','x\_3(t)','x\_4(t)','x\_5(t)');

axis([tini, tfin, -1.1\*max(max(abs(X))), 1.1\*max(max(abs(X)))]);

grid on;

box on;

figure(3)

plot(t,v1,'b',t,v2,'k',t,v3,'r',t,v4,'g',t,v5,'c');

title('velocidad diferencias centrales');

xlabel('tiempo t(s)');

ylabel('velocidad v(t) (m/s)');

legend('v\_1(t)','v\_2(t)','v\_3(t)','v\_4(t)','v\_5(t)');

axis([tini, tfin, -1.1\*max(max(abs(V))), 1.1\*max(max(abs(V)))]);

grid on;

box on;

figure(4)

plot(x1,v1,'b',x2,v2,'k',x3,v3,'r',x4,v4,'g',x5,v5,'c');

title('diagrama de fase diferencias centrales');

xlabel('desplazamiento x(t) (m)');

ylabel('velocidad v(t) (m/s)');

legend('x\_1(t) vs v\_1(t)','x\_2(t) vs v\_2(t)','x\_3(t) vs v\_3(t)','x\_4(t) vs v\_4(t)','x\_5(t) vs v\_5(t)');

axis([-1.1\*max(max(abs(X))), 1.1\*max(max(abs(X))), -1.1\*max(max(abs(V))), 1.1\*max(max(abs(V)))]);

grid on;

box on;

Gráfico, Gráfico de líneas

Descripción generada automáticamente

Gráfico, Histograma

Descripción generada automáticamente

Gráfico, Histograma

Descripción generada automáticamente

Imagen que contiene Diagrama

Descripción generada automáticamente

## Ecuación Diferencial 3.2 (10pts)

Resuelva la ecuación diferencial ordinaria de segundo orden no lineal, con valores iniciales mediante el método de Euler, incluyendo, desplazamiento, velocidad, diagrama de fase y espacio de fase

%% prog metodo de euler control 01 parte 3.2

clearvars;

close all;

clc;

%% incializacion

%ecuacion diferencial

% y'' + delta\*y' + beta\*y^3 + w0\*y = gama\*cos(w\*t + phi0)

%uso de variable auxiliar

% y' = u

% u = y'

% y'' = -delta\*y' - beta\*y^3 - w0\*y + gama\*cos(w\*t + phi0)

% u' = -delta\*y' - beta\*y^3 - w0\*y + gama\*cos(w\*t + phi0)

%datos

delta = 1.0;

beta = 1.0;

w0 = -1.0;

gama = 1.0;

phi0 = 0.0;

w = 0.1;

%tiempo incial/final

tini = 0;

tfin = 200;

%valores inciales%

y0 = 0.5; %posición incial

u0 = -0.5; %velocidad incial u(0) = y'(0)

%incrementos

deltat = 0.0001;

%valores de xn por prelocalización

tn = tini:deltat:tfin; %valores de tn

Nn = length(tn); %tamanho de tn

%valores yn por prelocalización

yn = zeros(1,Nn); %incialmente lleno de ceros

%valores un por prelocalización

un = zeros(1,Nn); %incialmente lleno de ceros

%condicion incial de desplazamiento

yn(1) = y0; %condicion incial desplazamiento

%condicion inicial de velocidad

un(1) = u0; %condicion incial velocidad u(0) = y'(0)

%% solucion numérica

for n = 1:Nn-1

yn(n+1) = yn(n) + deltat\*un(n);

un(n+1) = un(n) + deltat\*( -delta\*un(n) - beta\*yn(n)^3 - w0\*yn(n) + gama\*cos(w\*tn(n) + phi0) );

end;

%% espacio de fase

%desplazamiento

Lyini = -1.1\*max(abs(yn));

Lyfin = 1.1\*max(abs(yn));

dy = 0.01;

%velocidad

Luini = -1.1\*max(abs(un));

Lufin = 1.1\*max(abs(un));

du = 0.01;

%tiempo

Tini = -1.1\*max(abs(un));

Tfin = 1.1\*max(abs(un));

dt0 = 0.01;

%espacio de estados inciales

[y0,u0] = meshgrid(Lyini:dy:Lyfin,Luini:du:Lufin);

%tiempo

t0 = meshgrid(Lyini:dy:Lyfin,Luini:du:Lufin);

%aceleracion

v0 = -delta\*u0 - beta\*y0.^3 - w0\*y0 + gama\*cos(w\*t0 + phi0);

%vector

figure(1)

quiver(y0,u0,u0,v0,'r'); %dibuja flechas 2D

title('espacio de fase');

xlabel('y(t) (m) ');

ylabel('dy(t)/dt (m/s)')

legend('espfase');

daspect([1 1 1]);

axis([ -1.1\*max(abs(yn)) 1.1\*max(abs(yn)) 1.1\*min(un) 1.1\*max(un) ]);

grid on;

box on;

%% resultados

figure(2)

plot(tn,yn,'k')

title('solucion euler desplazamiento');

xlabel('t (s) ')

ylabel('y(t) (m)');

legend(['y(t) \Deltat = ' num2str(deltat)])

axis([ tini tfin -1.1\*max(abs(yn)) 1.1\*max(abs(yn))]);

grid on;

box on;

figure(3)

plot(tn,un,'k')

title('solucion euler velocidad');

xlabel('t (s)')

ylabel('dy(t)/dt (m/s)')

legend(['dy(t)/dt \Deltat = ' num2str(deltat)])

axis([ tini tfin 1.1\*min(un) 1.1\*max(un) ]);

grid on;

box on;

figure(4)

plot(yn,un,'k')

title('diagrama de fase');

xlabel('y(t) (m) ');

ylabel('dy(t)/dt (m/s) ');

legend(['y(t) vs dy(t)/dt \Deltat = ' num2str(deltat)])

axis([ -1.1\*max(abs(yn)) 1.1\*max(abs(yn)) 1.1\*min(un) 1.1\*max(un) ]);

grid on;

box on;

figure(5)

quiver(y0,u0,u0,v0,'r'); %dibuja flechas 2D

hold on;

plot(yn,un,'k')

title('espacio de fase y diagrama de fase');

xlabel('y(t) (m)');

ylabel('dy(t)/dt (m/s)');

legend('espfase',['y(t) vs dy(t)/dt \Deltat = ', num2str(deltat)] );

daspect([1 1 1]);

axis([ -1.1\*max(abs(yn)) 1.1\*max(abs(yn)) 1.1\*min(un) 1.1\*max(un) ]);

grid on;

box on;





Imagen que contiene Gráfico

Descripción generada automáticamente