UNIVERSIDADE ESTADUAL DE CAMPINAS

FACULDADE DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO

EA619R – LABORATÓRIO DE ANÁLISE LINEAR

PROF. RENATO DA ROCHA LOPES

PROF. RICARDO CORAÇÃO DE LEÃO FONTOURA DE OLIVEIRA

RELATÓRIO DO EXPERIMENTO 4:
SIMULAÇÃO DE SISTEMAS DINÂMICOS

Bryan Wolff	**RA: 214095**
João Luís Carvalho de Abreu	**RA: 175997**

Campinas

Junho de 2021

VÍDEOS DEMONSTRATIVOS:

- **[João Luís Carvalho de Abreu]**
- **[Bryan Wolff]**

INTRODUÇÃO E OBJETIVOS:

Este experimento objetiva analisar determinados sistemas dinâmicos a partir de equações diferenciais lineares e não-lineares. Os estudos foram realizados a partir do software Matlab, no qual foi simulado os comportamentos desses sistemas ao decorrer do tempo e os *scripts* implementados e utilizados estão disponibilizados em arquivos anexados a este relatório, assim como seus valores computados serão demonstrados no decorrer do desenvolvimento teórico.

1. ATIVIDADE 1

A partir do sistema dinâmico representado pela equação diferencial abaixo:

$$\dddot{y}(t) + (2+6p)\ddot{y}(t) + (9+12p)\dot{y}(t) + 18y(t) = 18x(t)$$

em que p é um parâmetro desconhecido pertencente à faixa [0.1,1.2], foi implementado um diagrama em Simulink do Matlab utilizando como base os integradores. O diagrama está disponibilizado em anexos com o nome "*sistemaLinearOrdem3.slx*", juntamente com o *script* do dessa atividade dada como *"atividade1.m",* também disponível no Anexo A. O objetivo dessa atividade é avaliar o valor do máximo sobressinal Mp, dado por (y_{max} - y_{final}) / y_{final} , por meio de simulações para vários valores de p.

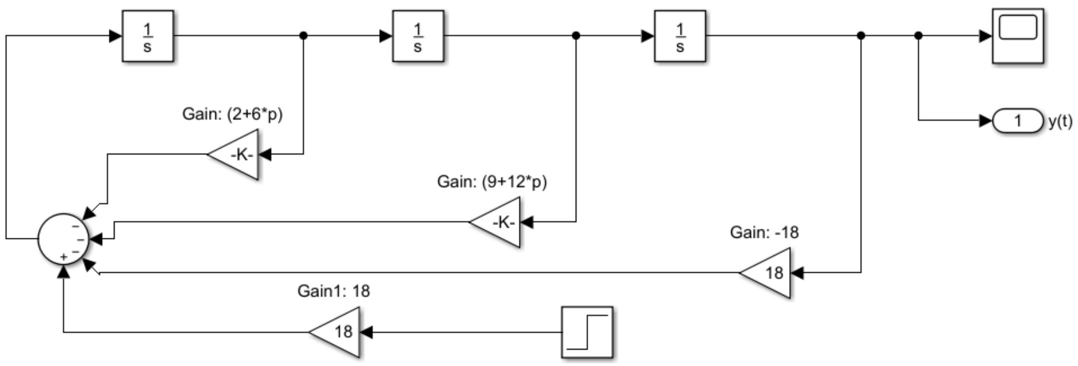

Figura 1: Diagrama Simulink do sistema dinâmico para condições iniciais nulas.

Para esse sistema, em função da incerteza do parâmetro p, a determinação do máximo sobressinal M_p pode ser feita por meio de simulações exaustivas, testando diversos valores de p, e para isso, vamos computar a solução da equação para 50 valores igualmente espaçados na faixa de p. Assim, foi gerado o seguinte gráfico:

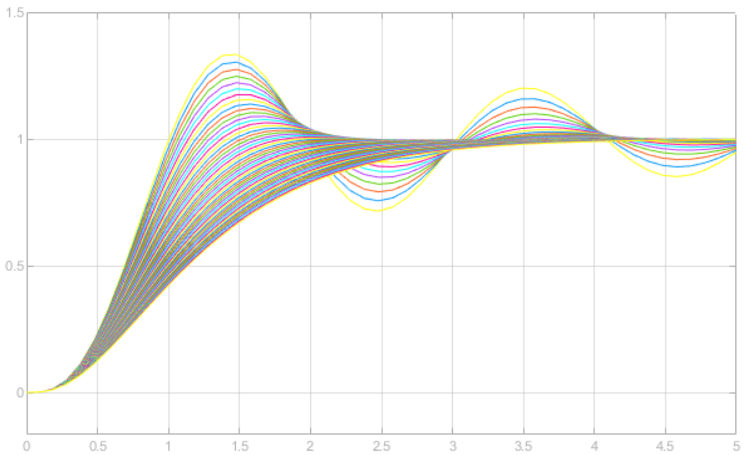

Figura 2: Resposta no tempo do sistema para as várias simulações exaustivas.

Por meio desta simulação e de todos os valores de $y(t)$ das curvas obtidas, é possível analisar o comportamento do sobressinal do sistema dinâmico estudado e obter os seguintes resultados:

1.1. VALOR MÁXIMO PARA O SOBRESSINAL

O sobressinal assume seu valor máximo quando M_p= 0.335, para p = 0.1, dado pelo instante de tempo 1.479 s.

1.2. TEMPO DE SUBIDA T_S:

O tempo necessário para a trajetória alcançar 100% do valor de referência aplicado na entrada, isto é, y(t) = 1, foi de 1.079 s.

1.3. TEMPO DE ACOMODAÇÃO T_A:

O instante de tempo a partir do qual a trajetória permanece dentro de uma faixa de 2% de seu valor final, isto é, dentro da faixa [0.98,1.02], foi de 9.179 s.

1.4. SOBRESSINAL MENOR QUE 7%

A subfaixa para a qual o sobressinal é menor ou igual a 7% é obtido quando o valor de p está compreendido entre 0.392 e 1.200. Além disso, os pólos da função pertencentes ao intervalo de p obtido anteriormente são dados por: -1.1755 + 2.7601i; -1.1755 - 2.7601i; -2.0000 + 0.0000i.

1.5. FAIXA EM QUE NÃO OCORRE SOBRESSINAL

A faixa de p para a qual não há sobressinal, isto é, o sobressinal é menor que 1% ($Mp \leq 0.01$) é dado por 0.549 e 1.200.

2. ATIVIDADE 2

O objetivo dessa atividade é estudar o movimento de um pêndulo simples, dado pela Figura 3, sob ação da aceleração da gravidade g com uma haste rígida de comprimento l que balança uma massa m, que é regido pela seguinte equação diferencial:

$$\ddot{\theta}(t) + \beta\dot{\theta}(t) + \frac{g}{\ell}\mathrm{sen}\theta(t) = 0$$

em que β é um coeficiente de amortecimento proporcional à velocidade angular do pêndulo. Adotou-se um pêndulo não amortecido, isto é, $\beta = 0$.

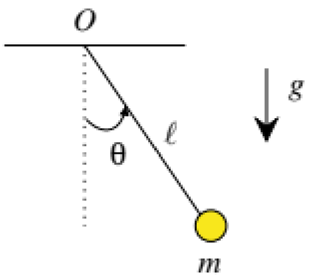

Figura 3: Pêndulo Simples.

O *script* dessa atividade está disponível no Anexo B, e também nos arquivos enviados com o nome *"atividade2.m"*.

2.1. REPRESENTAÇÃO EM ESPAÇO DE ESTADOS E ANÁLISE DOS PONTOS DE EQUILÍBRIO

Como foi adotado um pêndulo não amortecido, isto é, $\beta = 0$, a equação a ser analisada pode ser reescrita como:

$$\theta''(t) + \frac{g}{l}sen(\theta(t)) = 0$$

Para obter a representação de espaço de estados, definiu-se as variáveis x_1 e x_2 como:

$$x_1 = \theta(t);\ x_2 = \theta'(t)$$

Então,

$$x_1' = x_2$$

$$x_2' = -\frac{g}{l}sen(x_1)$$

Dessa forma, é possível verificar os pontos de equilíbrio do sistema, isto é, verificar o comportamento da função quando $x_1' = x_2 = 0$.

$$-\frac{g}{l}sen(x_1) = 0 \quad \Rightarrow \quad sen(x_1) = 0$$

Logo,

$$x_1 = n\pi \qquad n = 0, \pm1, \pm2, ...$$

Portanto, os pontos de equilíbrio são dados pelos pares ordenados ($n\pi$, 0) , onde n é um número inteiro (0,±1,±2,...).

2.2. MODELO LINEARIZADO EM TORNO DOS PONTOS DE EQUILÍBRIO

Adotando-se o comprimento da haste como $l = 1\,m$ e a aceleração da gravidade como $g = 9,81\,m/s^2$, pode-se obter o modelo linearizado em torno dos pontos de equilíbrio (0,0) e (π,0). Vale ressaltar que o modelo linearizado é definido por $x' = Ax + Bu$ em que, neste caso, $B \cdot u = 0$ devido à ausência de entrada. Dessa forma, definiu-se as funções f_1e f_2 para a construção da matriz A da seguinte forma:

$$f_1 = x_1; \qquad f_2 = -gsen(x_1)$$

$$A = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{bmatrix}$$

Então,

$$A = \begin{bmatrix} 0 & 1 \\ -gcos(x_1) & 1 \end{bmatrix}$$

Dessa forma, o modelo linearizado pode ser representado como:

$$\begin{bmatrix} x_1' \\ x_2' \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -gcos(x_1) & 0 \end{bmatrix} * \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

No sistema linearizado em torno do ponto de equilíbrio $(x_1,x_2) = (0,0)$, é dado por:

$$\begin{bmatrix} x_1' \\ x_2' \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -g & 0 \end{bmatrix} * \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

Já no ponto de equilíbrio $(x_1,x_2) = (\pi,0)$, o sistema linearizado é dado por:

$$\begin{bmatrix} x_1' \\ x_2' \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ g & 0 \end{bmatrix} * \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

2.3. ESTABILIDADE DO SISTEMA

Para calcular a estabilidade do sistema obtido no item anterior, é necessário calcular os autovalores associados à matriz A em cada ponto de equilíbrio. Para isso, primeiramente foram calculados e analisados os autovalores do ponto de equilíbrio $(x_1,x_2) = (0,0)$:

$$det(\lambda I - A) = det \begin{pmatrix} \lambda & -1 \\ g & \lambda \end{pmatrix} = 0$$

Então,

$$\lambda^2 + g = 0 \Rightarrow \quad \lambda = \pm j\sqrt{g}$$

Dessa forma, como o autovalor não apresenta parte real, trata-se de um comportamento oscilatório estável.

Agora, foram analisados os autovalores do ponto de equilíbrio $(x_1,x_2) = (\pi,0)$:

$$det(\lambda I - A) = det\begin{pmatrix} \lambda & -1 \\ -g & \lambda \end{pmatrix} = 0$$

Então,

$$\lambda^2 - g = 0 \Rightarrow \lambda = \pm\sqrt{g}$$

Dessa forma, como o sistema apresenta um autovalor positivo e um autovalor negativo, ele é caracterizado por um comportamento oscilatório instável.

2.4. SIMULAÇÃO DO PÊNDULO SIMPLES PARA CONDIÇÃO INICIAL (θ(0),ω(0)) = (π/10,0)

A partir do comando *ode45* do Matlab, e dada a condição inicial $(\theta(0), \omega(0)) = (\frac{\pi}{10}, 0)$, o comportamento do sistema pode ser obtido através da simulação de um gráfico de $\theta(t)$ e $\omega(t)$. Além disso, é possível visualizar o vídeo da simulação através deste link. Dessa forma, foi possível obter as seguintes simulações:

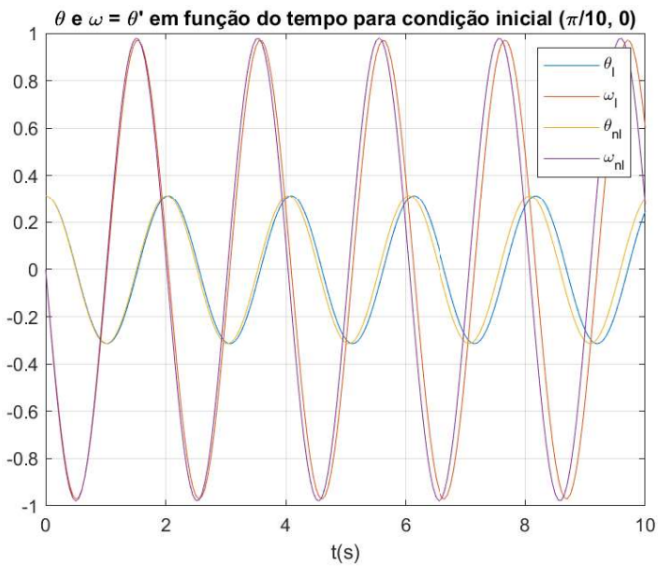

Figura 4: Simulação de $\theta(t)$ e $\omega(t)$ para os casos linear (l) e não linear (nl) dada a condição inicial

$(\theta(0), \omega(0)) \ = \ (\frac{\pi}{10}, 0)$.

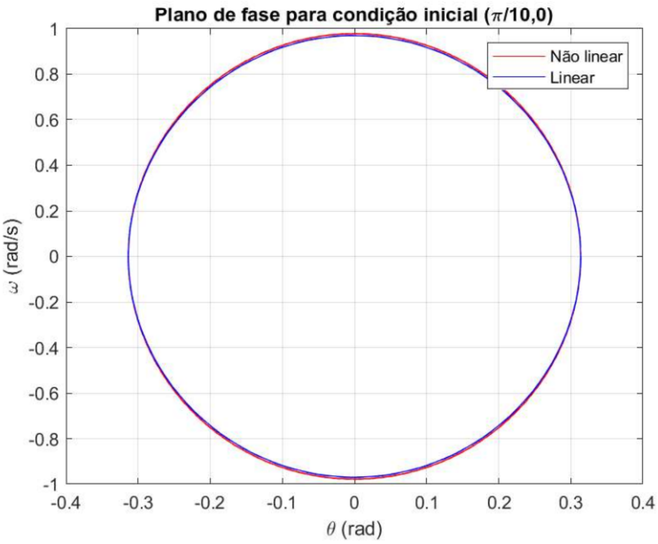

Figura 5: Plano de fase para a condição inicial $(\theta(0), \omega(0)) \ = \ (\frac{\pi}{10}, 0)$.

2.4.1. DEFASAGEM ENTRE AS SIMULAÇÕES PARA CONDIÇÃO INICIAL (θ(0),ω(0)) = (π/10,0)

Por fim, implementou-se uma função para o cálculo da defasagem angular no tempo entre os modelos de simulação não linear e linear e uma outra função para a verificação das defasagens superiores a 2° - ambas disponíveis no Anexo B deste relatório. O cálculo da defasagem representa a diferença modular entre $\theta_{nao\_linear}(t)$ e $\theta_{linear}(t)$ dado um mesmo instante de tempo.

Para a condição inicial $(\theta(0), \omega(0)) \ = \ (\frac{\pi}{10}, 0)$ e tendo em vista a defasagem superior a 2°, os seguintes intervalos de tempo foram obtidos e o seguinte gráfico foi esboçado:

Intervalos de tempo com defasagem superior a 2°
[3.404 , 3.804]
[4.354 , 4.904]
[5.304 , 5.954]
[6.254 , 7.004]
[7.254 , 8.004]

[8.254 , 9.054]
[9.254 , 10.000]

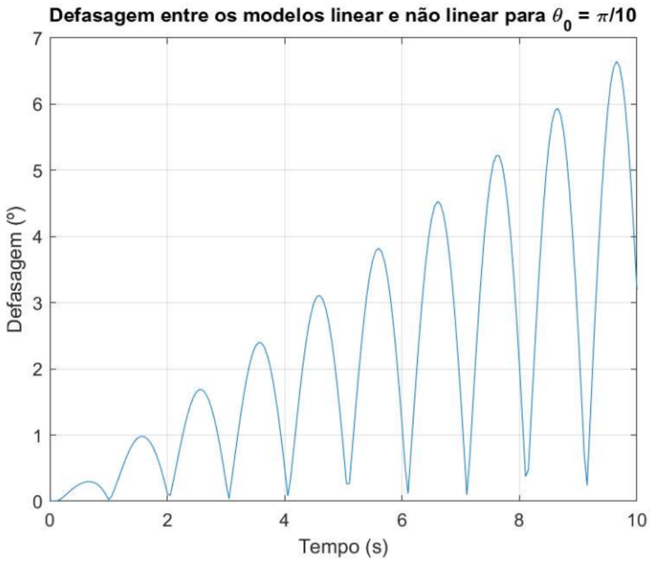

Figura 6: Simulação da defasagem entre os dois modelos implementados (linear e não linear) para a condição inicial $(\theta(0), \omega(0)) = (\frac{\pi}{10}, 0)$.

A partir das informações acima, é possível afirmar que a diferença na defasagem entre as simulações linear e não linear aumenta com a passagem do tempo. Apesar disso, o comportamento dos dois modelos para a condição inicial em questão é relativamente próximo, sendo possível verificar a discrepância de 2° apenas a partir do instante $3,404$ s.

2.5. SIMULAÇÃO DO PÊNDULO SIMPLES PARA CONDIÇÃO INICIAL (θ(0),ω(0)) = (π/4,0)

A partir do comando *ode45* do Matlab, e dada a condição inicial $(\theta(0), \omega(0)) = (\frac{\pi}{4}, 0)$, o comportamento do sistema pode ser obtido através da simulação de um gráfico de $\theta(t)$ e $\omega(t)$. Além disso, é possível visualizar o vídeo da simulação através deste link. Dessa forma, foi possível obter as seguintes simulações:

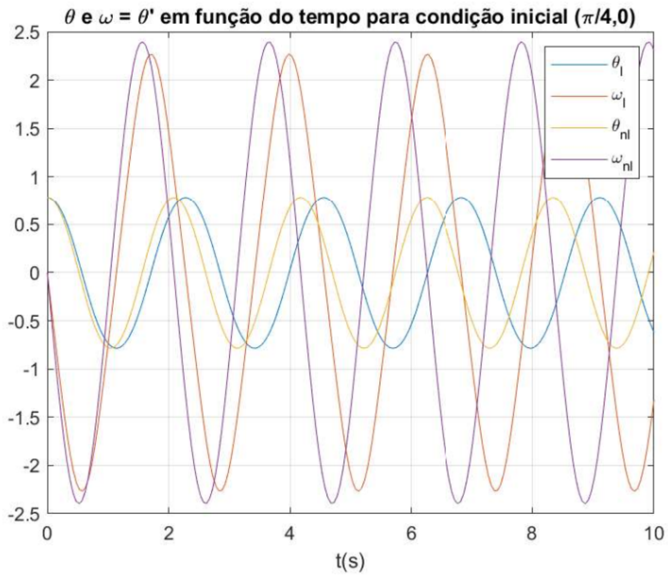

Figura 7: Simulação de $\theta(t)$ e $\omega(t)$ para os casos linear (l) e não linear (nl) dada a condição inicial $(\theta(0), \omega(0)) = (\frac{\pi}{4}, 0)$.

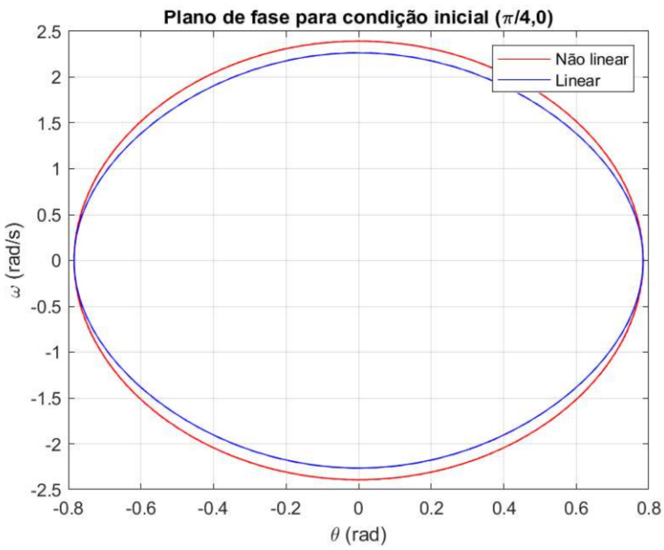

Figura 8: Plano de fase para a condição inicial$(\theta(0), \omega(0)) = (\frac{\pi}{4}, 0)$.

2.5.1. DEFASAGEM ENTRE AS SIMULAÇÕES PARA CONDIÇÃO INICIAL (θ(0),ω(0)) = (π/4,0)

Analogamente ao processo realizado para a condição inicial anterior, neste caso utilizou-se as mesmas funções citadas anteriormente. Para a condição inicial em questão, foram obtidos os seguintes intervalos de tempo e o seguinte gráfico:

Intervalos de tempo com defasagem superior a 2°
[0.279 , 1.029]
[1.179 , 2.179]
[2.229 , 3.279]
[3.329 , 6.529]
[6.579 , 7.629]
[7.579 , 10.000]

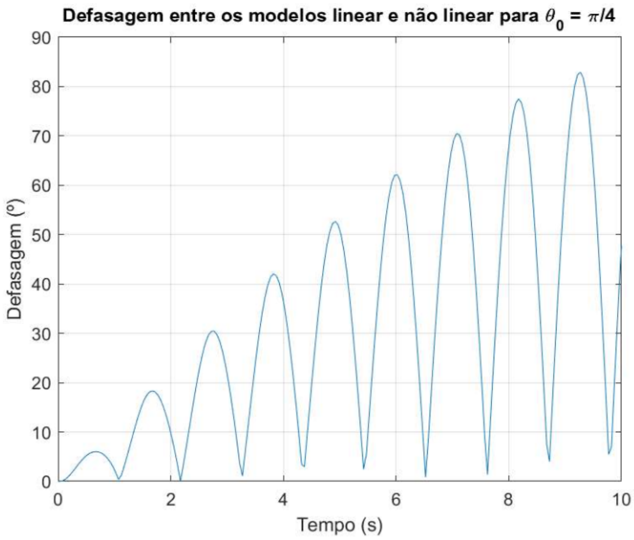

Figura 9: Simulação da defasagem entre os dois modelos implementados (linear e não linear) para a condição inicial $(\theta(0), \omega(0)) \; = \; (\frac{\pi}{4}, 0)$.

Apesar das semelhanças no comportamento do gráfico acima com relação à primeira condição inicial, principalmente pelo aumento da defasagem com a passagem do tempo, neste caso é possível afirmar que o instante de tempo em que a discrepância excede os 2° é menor do que o caso anterior. Para a condição $(\theta(0), \omega(0)) \; = \; (\frac{\pi}{4}, 0)$, essa diferença passa a ser verificada mais especificamente a partir de $0,279$s.

3. ATIVIDADE 3

O objetivo dessa atividade é estudar o comportamento de um pêndulo duplo ilustrado na figura abaixo, em que m_1 e m_2 são massas dos corpos, o valor l_1 é a distância da origem O ao corpo de massa m_1 e l_2 é a distância entre o corpo de massa m_1 e o corpo de massa m_2.

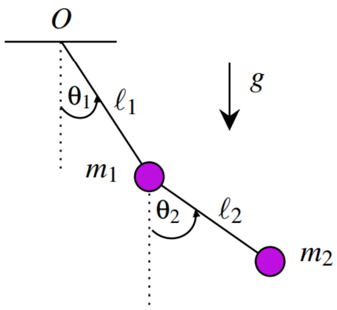

Figura 10: Pêndulo duplo.

A dinâmica do movimento do pêndulo duplo pode ser modelada por meio do seguinte conjunto de equações diferenciais não lineares:

$$\dot{\theta}_1 = \omega_1$$

$$\dot{\omega}_1 = \frac{-g(2m_1+m_2)\mathrm{sen}(\theta_1) - m_2 g\,\mathrm{sen}(\theta_1-2\theta_2) - 2\mathrm{sen}(\theta_1-\theta_2)m_2(\omega_2^2\ell_2+\omega_1^2\ell_1\cos(\theta_1-\theta_2))}{\ell_1(2m_1+m_2-m_2\cos(2\theta_1-2\theta_2))}$$

$$\dot{\theta}_2 = \omega_2$$

$$\dot{\omega}_2 = \frac{2\mathrm{sen}(\theta_1-\theta_2)(\omega_1^2\ell_1(m_1+m_2)+g(m_1+m_2)\cos(\theta_1)+\omega_2^2\ell_2 m_2\cos(\theta_1-\theta_2))}{\ell_2(2m_1+m_2-m_2\cos(2\theta_1-2\theta_2))}$$

Os parâmetros são escolhidos arbitrariamente ao início, desde que estejam dentro da faixa [0.5,2]. Dessa forma, foram escolhidos os seguintes parâmetros para efetuar a simulação:

$$l_1 = 1; \qquad m_1 = 2; \qquad l_2 = 0,5; \qquad m_2 = 0,5.$$

Agora, a partir do comando *ode45* do Matlab, desenvolveu-se um *script* - disponível no Anexo C e nos arquivos enviados com o nome *"atividade3.m"* - de forma a estudar o comportamento do sistema, que pode ser obtido através da simulação de um gráfico de $\theta(t)$ e $\omega(t)$. Além disso, é possível visualizar o vídeo da simulação através deste link. Dessa forma, foi possível obter as seguintes simulações:

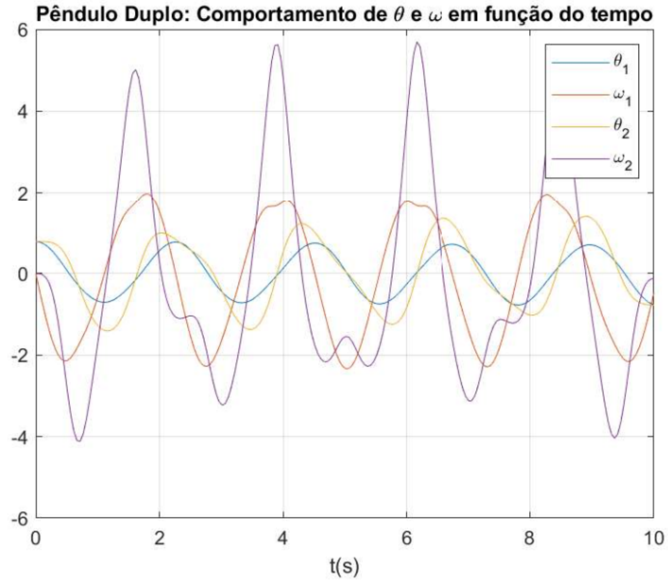

Figura 11: Simulação de $\theta(t)$ e $\omega(t)$ do Pêndulo Duplo.

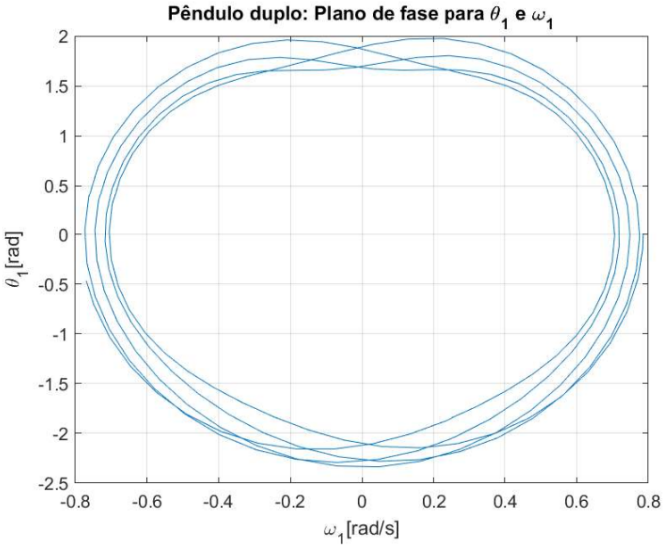

Figura 12: Plano de fase do Pêndulo Duplo para $\theta_1(t)$ e $\omega'_1(t)$.

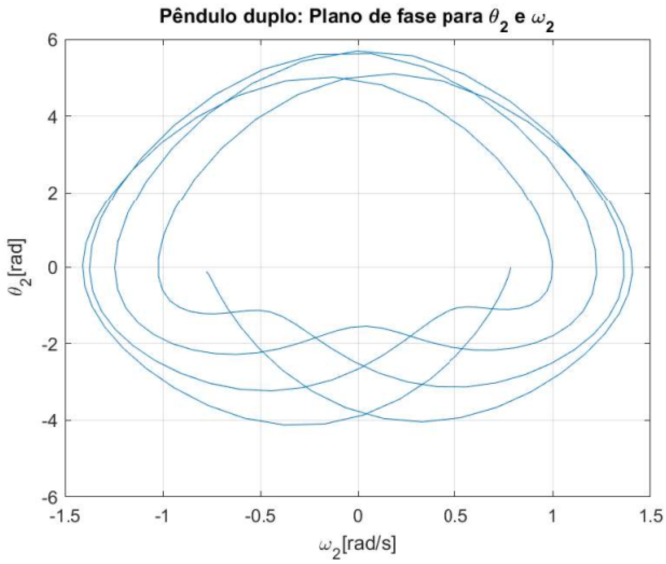

Figura 13: Plano de fase do Pêndulo Duplo para $\theta_2(t)$ e $\omega'_2(t)$

4. ANEXO A: *SCRIPT* DA ATIVIDADE 1

```
% Atividade 1

close all; clear all; clc;

% Equação diferencial:
%     y'''(t) + (2+6p)y''(t) + (9+12p)y'(t) + 18y(t) = 18x(t)
% em que p é um parâmetro desconhecido, pertencente à faixa p ? [0.1,1.2]

%---------------------------------------------------------------------------%
%---------------------------------------------------------------------------%

% Questão 3
p = linspace(0.1,1.2,50);

% Script para a chamada do diagrama desenvolvido no item 1
for i = 1:50
    simOut = sim('sistemaLinearOrdem3','SrcWorkspace','current','maxstep','0.1');
end

%---------------------------------------------------------------------------%
%---------------------------------------------------------------------------%

%Questão 4

% (a)
fprintf('Item (a)\n');
% valor final em que y estabiliza
y_f = simOut.yout{1}.Values.Data(end);

% [maior valor de y, posição do maior valor de y]
[m,pm] = max(simOut.yout{1}.Values.Data);

% [valor máximo para o sobressinal, índice do valor máximo]
[y_max, i] = max(m);
m_max = (y_max - y_f)/y_f;
fprintf("Valor máximo para o sobressinal (Mp): %.3f \n", m_max);

% valor de p para Mp máximo
p_max = p(i);
fprintf('Valor de p para o máximo Mp: %.3f \n', p_max);

% valor de pm para Mp máximo
pm_max = pm(i);
t_mp_max = simOut.tout(pm_max);
fprintf('Instante de tempo para Mp máximo: %.3f s\n', t_mp_max);
```

```matlab
% t_s: tempo de subida
% valor de referência na entrada: degrau unitário (= 1)

len_simOut = length(simOut.yout{1}.Values.Data);
for k = 1:len_simOut
    y(k) = simOut.yout{1}.Values.Data(k,1);
end

for j = 1:len_simOut
    j_index = j;
    if y(j_index) > 1
        break
    end
end
t_s = simOut.tout(j_index);
fprintf('Tempo de subida: %.3f s \n', t_s);

% t_a: tempo de acomodação
% valor de referência na entrada: degrau unitário (= 1)
% (1 - 0.02*1) = 0.98
% (1 + 0.02*1) = 1.02

k_in = 0;
k_aux = 0;
for k = 1:len_simOut
    if (y(k) >= (1-0.02*1)) && (y(k) <= (1+0.02*1))
        if (k_aux == 0)
            k_in = k;
            k_aux = 1;
        end
    else
        k_aux = 0;
    end
end
t_a = simOut.tout(k_in);
fprintf('Tempo de acomodação: %.3f s\n', t_a);

fprintf('---------------------------------------------------\n');

%-----------------------------------------------------------------------------%
% (b)
fprintf('Item (b)\n');
% mp < 0.07 --> mp = (m - y_f)/y_f < 0.07
contador = 1;
p7 = [];
for t = 1:50
  if (((m(t) - y_f)/y_f) <= 0.07)
      p7(contador) = p(t);
      contador = contador + 1;
  end
end
```

```
fprintf('Subfaixa de p para Mp <= 0.07: (%.3f , %.3f)\n', p7(1),p7(end));

fprintf('-------------------------------------------------------\n');

%-----------------------------------------------------------------------------------%
% (c)
fprintf('Item (c)\n');

% função para definir o p que gera Mp mais próximo de 7%
prox = 1;
for n = 1:50
  if (abs(((m(n) - y_f)/y_f) - 0.07) < prox) %se diferença for menor que a anterior
      prox = abs(((m(n) - y_f)/y_f) - 0.07); %redefine prox
      p_prox = p(n); %redefine p que gera o sinal
      mp_p_prox = m(n) - y_f;
  end
end

% função de transferência
func_transf = tf(18,[1 (2+6*p_prox) (9+12*p_prox) 18]);
polos = pole(func_transf);
fprintf('Polos da função: \n');
disp(polos);

fprintf('-------------------------------------------------------\n');
```

```

%-----------------------------------------------------------------------------------%
% (d)
fprintf('Item (d)\n');

contador = 1;
p1 = [];
for t = 1:50
  if (((m(t) - y_f)/y_f) <= 0.01)
      p1(contador) = p(t);
      contador = contador + 1;
  end
end

fprintf('Subfaixa de p para Mp <= 0.01: (%.3f , %.3f)\n', p1(1),p1(end));
```

5. ANEXO B: *SCRIPT* DA ATIVIDADE 2

```
% Atividade 2
clear all; close all; clc;

% p^2theta + g/l*sen(theta) = 0
% x1 = theta
% x2 = theta'
% A = [df1/x1  df1/x2] = [0              1]    B = [0]    u = 0
%     [df2/x1  df2/x2]   [(-g/l)cos(x1)  0]        [0]

% variáveis globais
global g; % aceleração da gravidade
global l; % comprimento da haste

l = 1;
g = 9.81;

tempoTotal = 10;
optOde = odeset('maxStep', 0.05);

% condições iniciais
x0 = [pi/10 0];

% sistema linear
outOde_linear10 = ode45(@linSys, [0 tempoTotal], x0, optOde);

% sistema não linear
outOde_nao_linear10 = ode45(@notlinSys, [0 tempoTotal], x0, optOde);

% plot das funções
figure();
plot(outOde_linear10.x, outOde_linear10.y);
hold on
plot(outOde_nao_linear10.x, outOde_nao_linear10.y)
xlabel t(s); grid;
legend("\theta_{l}","\omega_{l}","\theta_{nl}","\omega_{nl}")
title("\theta e \omega = \theta' em função do tempo para condição inicial (\pi/10, 0)")

x02 = [pi/4 0];
% sistema linear
outOde_linear4 = ode45(@linSys, [0 tempoTotal], x02, optOde);

% sistema não linear
outOde_nao_linear4 = ode45(@notlinSys, [0 tempoTotal], x02, optOde);
```

```
% plot das funções
figure();
plot(outOde_linear4.x, outOde_linear4.y);
hold on
plot(outOde_nao_linear4.x, outOde_nao_linear4.y)
xlabel t(s); grid;
legend("\theta_{l}","\omega_{l}","\theta_{nl}","\omega_{nl}")
title("\theta e \omega = \theta' em função do tempo para condição inicial (\pi/4,0)")

% plano de fases para a primeira condição inicial
figure()
plot(outOde_nao_linear10.y(1,:),outOde_nao_linear10.y(2,:),'r');
title('Plano de fase para condição inicial (\pi/10,0)');
hold on
plot(outOde_linear10.y(1,:),outOde_linear10.y(2,:),'b');
xlabel('\theta (rad)');
ylabel('\omega (rad/s)'); grid;
legend('Não linear','Linear');

% plano de fases para a segunda condição inicial
figure()
plot(outOde_nao_linear4.y(1,:),outOde_nao_linear4.y(2,:),'r');
title('Plano de fase para condição inicial (\pi/4,0)');
hold on
plot(outOde_linear4.y(1,:),outOde_linear4.y(2,:),'b');
xlabel('\theta (rad)');
ylabel('\omega (rad/s)'); grid;
legend('Não linear','Linear');
```

```
% gerando o video 1
geraVideo(outOde_linear10.x,outOde_nao_linear10.y,outOde_linear10.y,tempoTotal,1,'pendSimplesCondicao1');

% gerando o video 2
geraVideo(outOde_linear4.x,outOde_nao_linear4.y,outOde_linear4.y,tempoTotal,1,'pendSimplesCondicao2');

% Defasagem
defasagem10 = dif(outOde_nao_linear10, outOde_linear10);
title('Defasagem entre os modelos linear e não linear para \theta_0 = \pi/10');
defasagem4 = dif(outOde_nao_linear4, outOde_linear4);
title('Defasagem entre os modelos linear e não linear para \theta_0 = \pi/4');

disp("Intervalos de tempo com defasagem superior a 2° para pi/10")
grau_2(defasagem10,outOde_linear10.x)
fprintf("\n\n")
disp("Intervalos de tempo com defasagem superior a 2° para pi/4")
grau_2(defasagem4,outOde_linear4.x)
fprintf("\n")

function defasagem = dif(nao_linear,linear)
    defasagem = abs(rad2deg(nao_linear.y(1,:) - linear.y(1,:)));
    figure()
    plot(nao_linear.x, defasagem); grid;
    ylabel('Defasagem (°)'); xlabel('Tempo (s)');
end
```

```matlab

function intervalo_tempo = grau_2(defasagem, nao_linear)
    %Encontra os intervalos temporais com defasagem superiores a 2 graus
    aux = 0;
    for j = 1:length(defasagem)
        if (defasagem(1,j) >= 2) && (aux == 0)
            aux = 1;
            fprintf("T: [%.3f ,", nao_linear(j))
        elseif (defasagem(1,j) <= 2) && (aux == 1)
            aux = 0;
            fprintf(" %.3f] s\n", nao_linear(j))
        end
    end
end

function dX = linSys(t,x)
global g;
global l;
g = 9.81;
l = 1;

A = [0 1; (-g/l)*cos(x(1)) 0];
B = [0;0];
u = 0;

dX = A*x + B*u;
end
```

```matlab

function dX_nao_linear = notlinSys(t,x)
global g;
global l;
g = 9.81;
l = 1;

dX_nao_linear(1,1) = x(2);
dX_nao_linear(2,1) = -(g/l)*sin(x(1));
end
```

```matlab

function geraVideo(tempo,estadosN,estadosL,tempoTotal,L,titulo)
% entradas:
% tempo: vetor contendo os instantes de tempo da simula ?c~ao
% estadosN: vetor contando os estados ( theta e dot_theta )
% da simulação não linear
% estadosL: vetor contando os estados ( theta e dot_theta )
% da simulação linear
% tempoTotal: tempo total da simulação
% L: comprimento do fio ( metros )
writerObj = VideoWriter (titulo,'Motion JPEG AVI');
writerObj.FrameRate = ceil(size(tempo,2) / tempoTotal);
```

```
open (writerObj);
fig = figure();
ori = [0 0]; % origem do pêndulo
f = 1;
while f <= length(tempo)
    theta_nao_linear = estadosN(1,f);
    % determine aqui as coordenadas do corpo para o caso não linear (use L)
    x_nao_linear = L*sin(theta_nao_linear);
    y_nao_linear = -L*cos(theta_nao_linear);

    % determine aqui as coordenadas do corpo para o caso linearizado(use L)
    theta_linear = estadosL(1,f);
    x_linear = L*sin(theta_linear);
    y_linear = -L*cos(theta_linear);

    axis ([-2 2 -2.5 0.5]);
    hold on ;

    % desenhe os pêndulos aqui
    %não-linear
    plot(x_nao_linear,y_nao_linear,'o','MarkerSize',10,'MarkerFaceColor','r')
    line([0 x_nao_linear],[0 y_nao_linear]);

    %linear
    plot(x_linear,y_linear,'o','MarkerSize',10,'MarkerFaceColor','b')
    line([0 x_linear],[0 y_linear]);

    legend('Massa NL','Haste NL','Massa L','Haste L'); grid;

    hold off;
    F = getframe;
    writeVideo(writerObj,F);
    clf(fig);
    f = f + 1;
end
close(writerObj);
end
```

6. ANEXO C: *SCRIPT* DA ATIVIDADE 3

```
% Atividade 3
clear all; close all; clc;

global g;
global L1;
global L2;
global m1;
global m2;

% arbitrando valores para as variáveis
g = 9.81; % aceleração da gravidade
L1 = 1; % comprimento da haste 1
L2 = 0.5; % comprimento da haste 2
m1 = 2; % corpo de massa 1
m2 = 0.5; % corpo de massa 2

% condição inicial do sistema
x0 = [pi/4 0 pi/4 0];

tempoTotal = 10;
optOde = odeset('maxStep', 0.05);

% solução a partir da função de pêndulo duplo
outOde = ode45(@penduloDuplo,[0 tempoTotal],x0,optOde);

% plot da função
figure();
plot(outOde.x, outOde.y);
xlabel t(s); grid;
legend("\theta_{1}","\omega_{1}","\theta_{2}","\omega_{2}");
title("Pêndulo Duplo: Comportamento de \theta e \omega em função do tempo");


% planos de fase
figure();
plot(outOde.y(1,:),outOde.y(2,:));
xlabel('\omega_1[rad/s]'); grid;
ylabel('\theta_1[rad]');
title("Pêndulo duplo: Plano de fase para \theta_{1} e \omega_{1}");

figure()
plot(outOde.y(3,:),outOde.y(4,:)); xlabel t;
xlabel('\omega_2[rad/s]'); grid;
ylabel('\theta_2[rad]');
title("Pêndulo duplo: Plano de fase para \theta_{2} e \omega_{2}");

% gerando video
geraVideo(outOde.x,outOde.y,tempoTotal,L1,L2,'penduloDuplo.avi');
```

```
function dX = penduloDuplo(t,x)
global L1;
global L2;
global m1;
global m2;
global g;

% variaveis de estado x1 = theta_1 , x2 = w_1 , x3 = theta_2 , x4 = w_2
dX(1,1) = x(2);
num1 = -g*(2*m1+m2)*sin(x(1))-m2*g*sin(x(1)-2*x(3))-2*sin(x(1)-x(3))*m2*(x(4)^2*L2+x(2)^2*L1*cos(x(1)-x(3)));
den1 = L1*(2*m1+m2-m2*cos(2*x(1)-2*x(3)));
num2 = 2*sin(x(1)-x(3))*(x(2)^2*L1*(m1+m2)+g*(m1+m2)*cos(x(1))+x(4)^2*L2*m2*cos(x(1)-x(3)));
den2 = L2*(2*m1+m2-m2*cos(2*x(1)-2*x(3)));
dX(2,1) = num1/den1;
dX(3,1) = x(4);
dX(4,1) = num2/den2;
end

```

```
function geraVideo(tempo,estadosN,tempoTotal,L1,L2,titulo)
% entradas :
% tempo : vetor contendo os instantes de tempo da simula ?c~ao
% estadosN : vetor contando os estados ( theta e dot_theta )
%da simulação não linear
% estadosL : vetor contando os estados ( theta e dot_theta )
%da simulação linear
% tempoTotal : tempo total da simula ?c~ao
% L: comprimento do fio ( metros )
writerObj = VideoWriter(titulo,'Motion JPEG AVI');
writerObj.FrameRate = ceil(size(tempo,2)/tempoTotal);
open(writerObj);
fig = figure();
ori = [0 0]; % origem do p^endulo
f = 1;
while f <= length(tempo)
    theta1 = estadosN(1,f);
    theta2 = estadosN(3,f);
    % determine aqui as coordenadas do corpo para o corpo 1
    x1 = L1*sin(theta1);
    y1 = -L1*cos(theta1);
    % determine aqui as coordenadas do corpo para o corpo 2
    x2 = x1 + L2*sin(theta2);
    y2 = y1 - L2*cos(theta2);

    axis([-2 2 -2.5 0.5]);
    hold on;
    % desenhe os pêndulos aqui
    plot(x1,y1,'o','MarkerSize',10,'MarkerFaceColor','r')
    line([0 x1], [0 y1])
    plot(x2,y2,'o','MarkerSize',10,'MarkerFaceColor','b')
    line([x1 x2], [y1 y2])
    hold off;
    F = getframe;
```

```
        writeVideo(writerObj,F);
        clf(fig);
        f = f + 1;
    end
    close(writerObj);
end
```