# UNIVERSIDADE FEDERAL DE UBERLÂNDIA

## Roteiro 01a - Modelagem e Simulação de Sistemas Dinâmicos

Sistemas e Controle

**Rodrigo Henrique Alves Ferreira - 11811ECP001**

# 1 Atividades

1. As equações diferenciais eram ferramentas essenciais para descrever o movimento de sistemas dinâmicos, como planetas em órbita. Poincaré percebeu que as soluções dessas equações muitas vezes exibiam comportamentos imprevisíveis, desafiando a visão predominante da época de que sistemas físicos eram completamente previsíveis. Ele introduziu a ideia de sensibilidade às condições iniciais e trouxe o conceito de atratores - regiões no espaço de fase onde as trajetórias tendiam a se agrupar. Através do problema dos três corpos, que explora as interações gravitacionais entre três corpos celestes, Poincaré demonstrou que, devido à complexidade matemática envolvida, encontrar soluções analíticas exatas era uma tarefa praticamente impossível. Em vez disso, ele adotou uma abordagem geométrica e topológica para compreender as características qualitativas das órbitas, estabelecendo as bases para a teoria das órbitas periódicas e a análise topológica de sistemas dinâmicos. Através de suas incursões nas equações diferenciais e no problema dos três corpos, Poincaré foi um dos pioneiros na formação da área de sistemas dinâmicos. Seu trabalho ajudou a lançar as bases para nossa compreensão moderna da complexidade, do caos e da não-linearidade presentes em sistemas naturais e artificiais. Ao investigar a imprevisibilidade e a riqueza de padrões nos movimentos, Henri Poincaré desempenhou um papel crucial no desenvolvimento dessa disciplina interdisciplinar que continuou a influenciar matemáticos, físicos e cientistas ao longo do século XX e além.

2. a) Equações Diferenciais Ordinárias (ODEs) são utilizadas para descrever sistemas que mudam em apenas uma dimensão, geralmente com relação ao tempo. Por outro lado, Equações Diferenciais Parciais (PDEs) são empregadas quando lidamos com sistemas que variam em múltiplas dimensões, como espaço e tempo. ODEs são aplicadas em situações unidimensionais, como o movimento de partículas individuais. Já PDEs são usadas para modelar fenômenos espaciais, como propagação de calor ou ondas em um meio.

   b) Um gráfico de espaço de fase representa as relações entre as variáveis de estado de um sistema dinâmico em um espaço multidimensional. Ele traça as trajetórias ao longo do tempo, mostrando como as variáveis interagem e evoluem. Esse tipo de gráfico oferece informações essenciais sobre o comportamento qualitativo do sistema. Ele nos permite identificar pontos de equilíbrio, periodicidades, ciclos, comportamentos caóticos e atratores, além de fornecer uma visão clara das tendências do sistema ao longo do tempo. Em resumo, o gráfico de espaço de fase é uma ferramenta valiosa para visualizar e entender o comportamento complexo e dinâmico dos sistemas ao mapear suas diferentes trajetórias e possíveis estados ao longo do tempo.

c) A exponenciação de matriz $e^A$ possui significado matemático e aplicações físicas. Ela descreve a evolução de sistemas sob transformações lineares ao longo do tempo. Por exemplo, considerando a matriz $A$, $e^{At}$ modela a evolução temporal do sistema representado por $A$, sendo aplicado em áreas como mecânica quântica para entender a dinâmica de sistemas complexos.

Suponha que tenhamos a matriz $A$ dada por:

$$A = \begin{bmatrix} 2 & 1 \\ 1 & 3 \end{bmatrix}$$

Ao elevar $e^A$ a um tempo $t$, obtemos a matriz $e^{At}$ que descreve a evolução do sistema representado por $A$ ao longo do tempo. Esse processo é particularmente útil em mecânica quântica, onde matrizes exponenciais modelam a evolução temporal de sistemas quânticos e operadores lineares.

3. Um sistema de controle é um conjunto organizado de componentes interconectados que trabalham juntos para regular ou manipular o comportamento de um sistema dinâmico. O objetivo principal de um sistema de controle é garantir que as saídas do sistema atinjam os valores desejados, mesmo quando ocorrem perturbações ou variações nas entradas.

   Tópicos fundamentais em sistemas de controle incluem:

   - **Sinal de Referência:** É o valor desejado da saída do sistema, também chamado de "setpoint".
   - **Erro:** A diferença entre o sinal de referência e a saída real do sistema, que é medida e usada para ajustar o sistema.
   - **Controlador:** Componente responsável por calcular e aplicar um sinal de controle com base no erro. Pode ser proporcional, integral, derivativo ou uma combinação (PID).
   - **Planta ou Processo:** Representa o sistema físico que está sendo controlado, podendo ser uma máquina, um processo industrial, um veículo, entre outros.
   - **Sensor:** Dispositivo que mede a saída real do sistema e fornece feedback ao controlador.
   - **Atuador:** Componente que executa a ação de controle no sistema, geralmente com base no sinal de controle do controlador.

- **Malha Aberta e Malha Fechada:** Na malha aberta, o sistema age sem levar em conta o feedback da saída. Na malha fechada, o feedback é usado para ajustar o sistema e alcançar a saída desejada.
- **Estabilidade:** Um sistema de controle é estável quando suas saídas permanecem limitadas em resposta a perturbações, em vez de divergirem.
- **Resposta Transitória e Permanente:** A resposta transitória é a reação inicial do sistema após uma mudança na entrada. A resposta permanente é o estado final estabilizado após as oscilações iniciais.

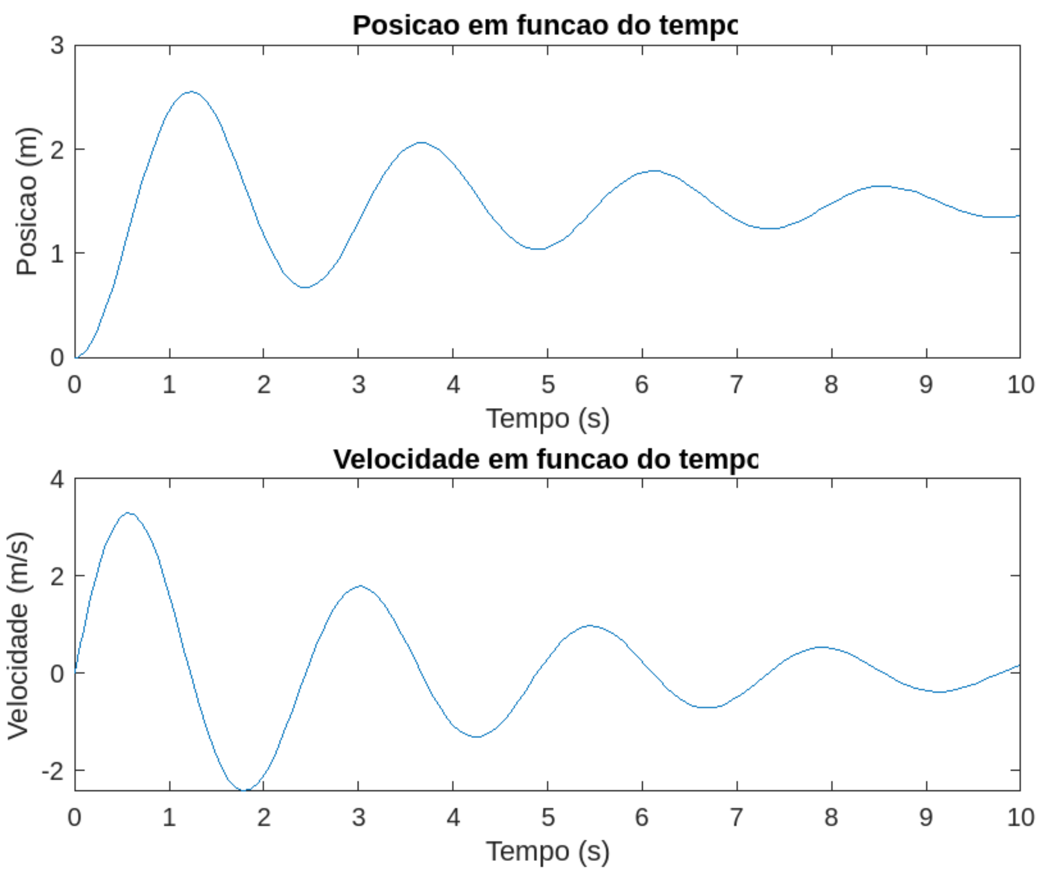
Em suma, um sistema de controle busca manipular a saída de um sistema dinâmico para atingir um objetivo desejado. Para isso, é crucial compreender e ajustar os elementos fundamentais, como o controlador, a planta, o feedback e outros componentes, para alcançar a estabilidade, eficiência e precisão necessárias.

4. Exemplo 1.3

   Código em matlab

```matlab
% Parametros do sistema
m = 3;      % massa em kg
k = 20;     % constante da mola em N/m
b = 1.5;    % coeficiente de amortecimento em N*s/m
g = 9.81;   % aceleracao devida a gravidade em m/s^2

% Equacao diferencial do sistema: m*x'' + b*x' + k*x = m*g
% Convertendo para um sistema de equacoes de primeira ordem:
% x1' = x2
% x2' = (1/m) * (m*g - b*x2 - k*x1)

% Funcao para as equacoes diferenciais
func = @(t, y) [y(2); (1/m) * (m*g - b*y(2) - k*y(1))];

% Condicoes iniciais
t0 = 0;
x0 = 0;
v0 = 0;
initial_conditions = [x0; v0];

% Tempo de simulacao
tspan = [t0, 10];  % Tempo de 0 a 10 segundos

% Resolvendo as equacoes diferenciais usando ode45
[t, sol] = ode45(func, tspan, initial_conditions);

% Obtendo posicoes e velocidades
positions = sol(:, 1);
```

```
velocities = sol(:, 2);

% Plotando os resultados
figure;
subplot(2, 1, 1);
plot(t, positions);
xlabel('Tempo (s)');
ylabel('Posicao (m)');
title('Posicao em funcao do tempo');

subplot(2, 1, 2);
plot(t, velocities);
xlabel('Tempo (s)');
ylabel('Velocidade (m/s)');
title('Velocidade em funcao do tempo');
```

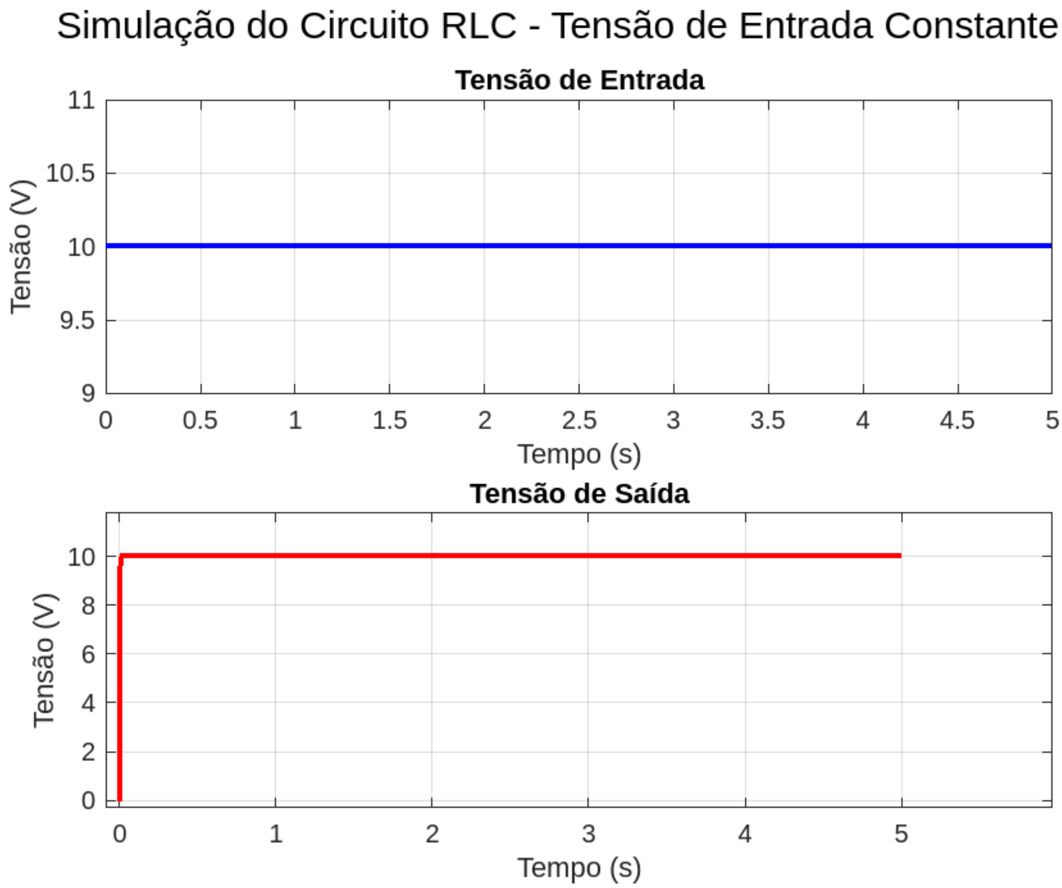



Exemplo 1.4

a) Simulação da tensão de entrada constante em 10V:

```
% Par metros do circuito
R = 1e3;   % Resist ncia em ohms
L = 1e-3;  % Indut ncia em henrys
C = 1e-6;  % Capacit ncia em farads

% Tempo de simula  o
```

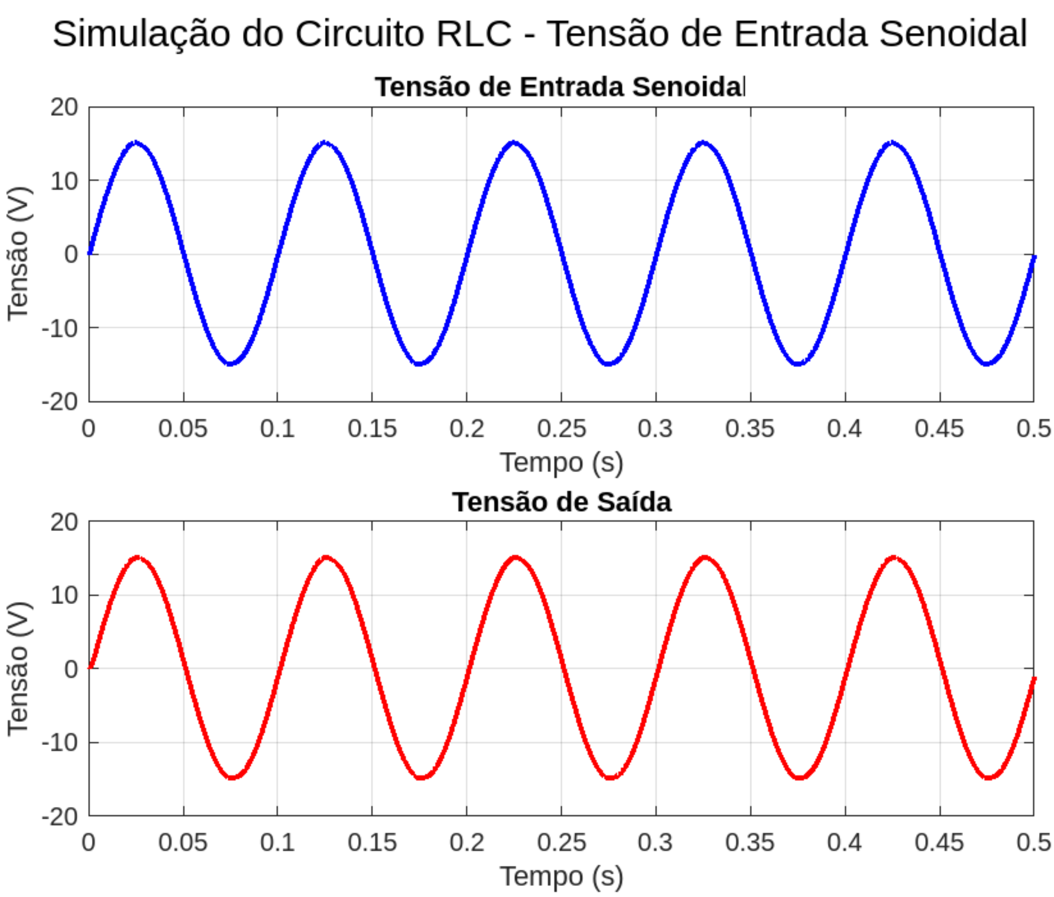
```
t = 0:0.001:5;  % Vetor de tempo de 0 a 5 segundos, passo de 1 ms

% Tens o de entrada constante
Vin = 10;  % Tens o constante de entrada

% Simula  o do circuito RLC
sys = tf(1, [L*C, R*C, 1]);  % Fun  o de transfer ncia do circuito
Vout = lsim(sys, Vin * ones(size(t)), t);

% Plot dos resultados
figure;
subplot(2, 1, 1);
plot(t, Vin * ones(size(t)), 'b', 'LineWidth', 2);
title('Tens o de Entrada');
xlabel('Tempo (s)');
ylabel('Tens o (V)');
grid on;

subplot(2, 1, 2);
plot(t, Vout, 'r', 'LineWidth', 2);
title('Tens o de Sa da');
xlabel('Tempo (s)');
ylabel('Tens o (V)');
grid on;

sgtitle('Simula  o do Circuito RLC - Tens o de Entrada Constante');
```




b) Simulação da tensão de entrada senoidal:

```
% Par metros do circuito
R = 1e3;    % Resist ncia em ohms
L = 1e-3;   % Indut ncia em henrys
C = 1e-6;   % Capacit ncia em farads

% Tempo de simula  o
t = 0:0.0001:0.5;  % Vetor de tempo de 0 a 0.5 segundos, passo de 0.1 ms

% Tens o de entrada senoidal
Vin = 15 * sin(20 * pi * t);  % Tens o de entrada senoidal

% Simula  o do circuito RLC
sys = tf(1, [L*C, R*C, 1]);  % Fun  o de transfer ncia do circuito
Vout = lsim(sys, Vin, t);

% Plot dos resultados
figure;
subplot(2, 1, 1);
plot(t, Vin, 'b', 'LineWidth', 2);
title('Tens o de Entrada Senoidal');
```

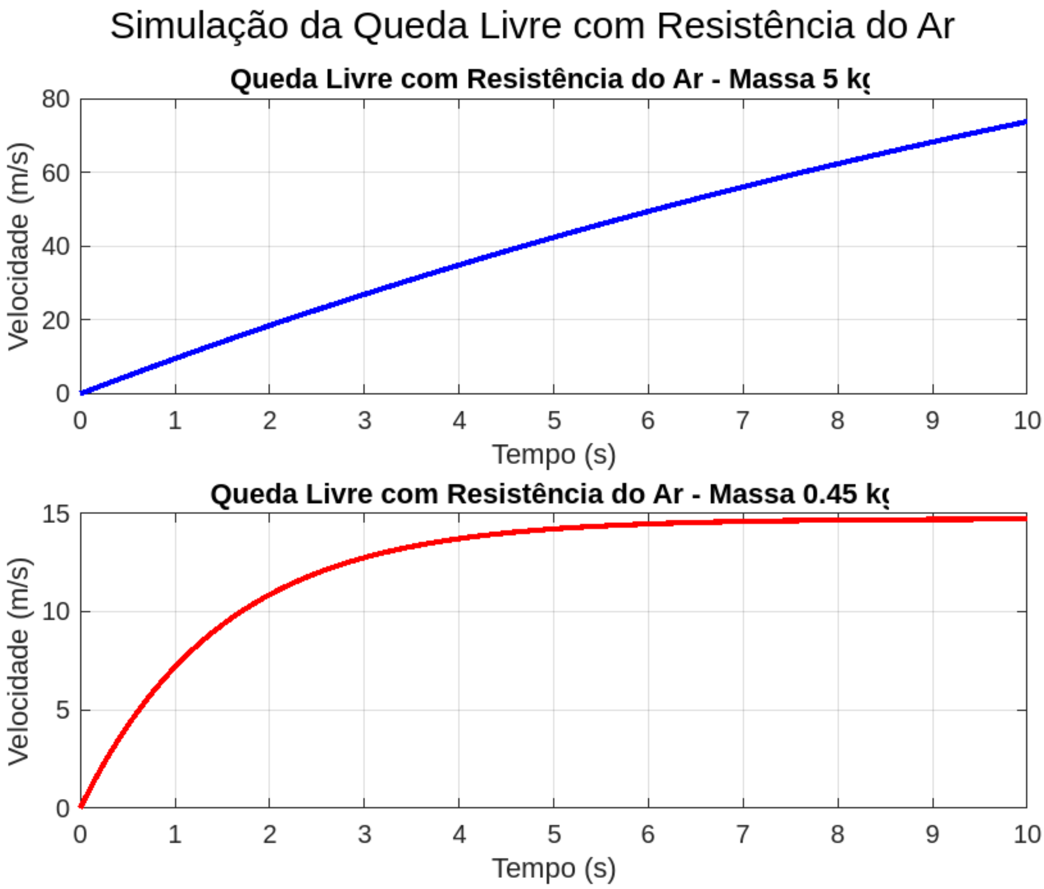
```
xlabel('Tempo (s)');
ylabel('Tens o (V)');
grid on;

subplot(2, 1, 2);
plot(t, Vout, 'r', 'LineWidth', 2);
title('Tens o de Sa da');
xlabel('Tempo (s)');
ylabel('Tens o (V)');
grid on;

sgtitle('Simula  o do Circuito RLC - Tens o de Entrada Senoidal');
```




Exemplo 1.6

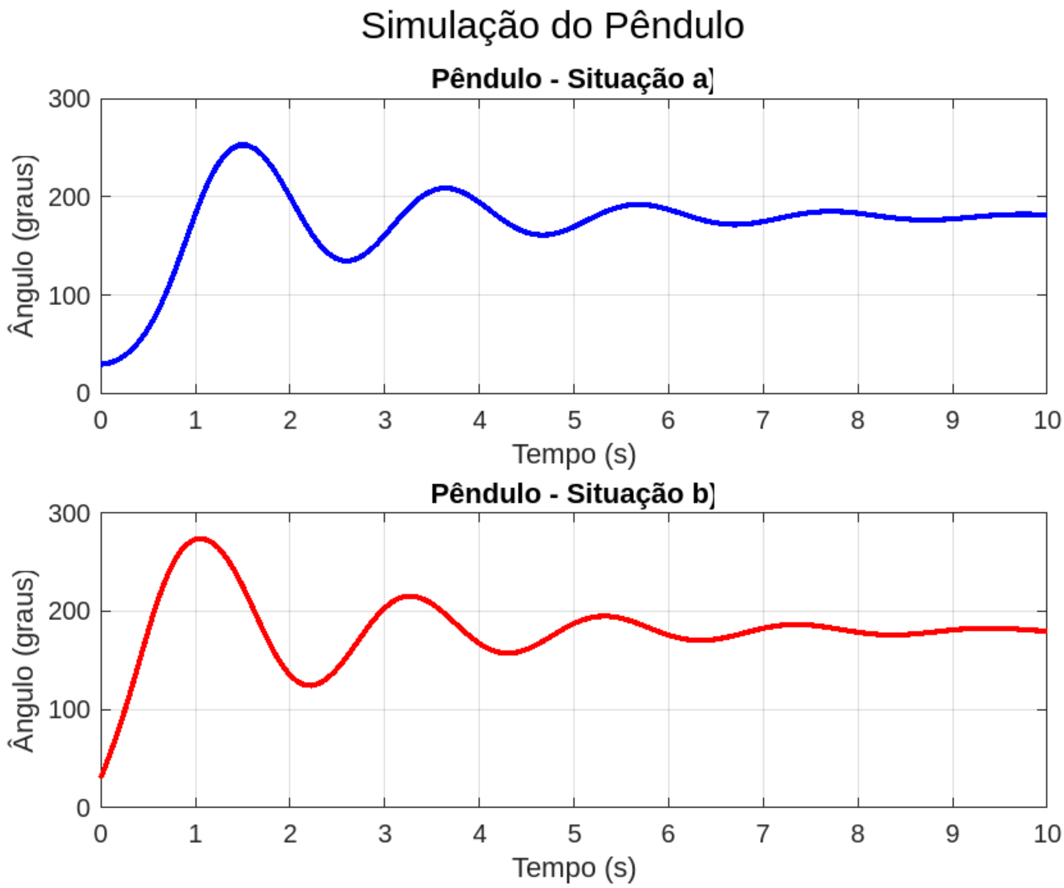
```
% Par metros do sistema
b = 0.3;    % Coeficiente de resist ncia do ar em kg*tm
g = 9.81;   % Acelera  o devido   gravidade em m/s^2

% Tempo de simula  o
t = 0:0.01:10;   % Vetor de tempo de 0 a 10 segundos, passo de 0.01 s

```

```
% Massas a serem consideradas
mass_a = 5;    % Massa A em kg
mass_b = 0.45; % Massa B em kg

% Fun  o de acelera  o considerando resist ncia do ar
acceleration = @(m, v) g - (b/m) * v;

% Simula  o para a massa A (5 kg)
v_a = zeros(size(t));  % Vetor de velocidade para a massa A
v_a(1) = 0;  % Condi  o inicial de velocidade

for i = 2:length(t)
    v_a(i) = v_a(i - 1) + acceleration(mass_a, v_a(i - 1)) * (t(i) - t(i - 1));
end

% Simula  o para a massa B (0.45 kg)
v_b = zeros(size(t));  % Vetor de velocidade para a massa B
v_b(1) = 0;  % Condi  o inicial de velocidade

for i = 2:length(t)
    v_b(i) = v_b(i - 1) + acceleration(mass_b, v_b(i - 1)) * (t(i) - t(i - 1));
end

% Plot dos resultados
figure;
subplot(2, 1, 1);
plot(t, v_a, 'b', 'LineWidth', 2);
title('Queda Livre com Resist ncia do Ar - Massa 5 kg');
xlabel('Tempo (s)');
ylabel('Velocidade (m/s)');
grid on;

subplot(2, 1, 2);
plot(t, v_b, 'r', 'LineWidth', 2);
title('Queda Livre com Resist ncia do Ar - Massa 0.45 kg');
xlabel('Tempo (s)');
ylabel('Velocidade (m/s)');
grid on;

sgtitle('Simula  o da Queda Livre com Resist ncia do Ar');
```




```
% Par metros do p ndulo
m = 1.2;     % Massa em kg
r = 0.10;    % Dist ncia do centro de massa ao eixo de rota  o em
    m
l = 1.5;     % Comprimento da haste em m
b = 1.5;     % Coeficiente de amortecimento em kg/m
g = 9.81;    % Acelera  o devido   gravidade em m/s^2
J = (m * r^2 + 2 * m * l^2) / 3;  % Momento de in rcia

% Condi  es iniciais
theta0_deg = 30;  %  ngulo  inicial em graus
theta0_rad = deg2rad(theta0_deg);  %  ngulo  inicial em radianos
omega0 = 0;  % Velocidade angular inicial

% Tempo de simula  o
tspan = [0, 10];  % Intervalo de tempo de 0 a 10 segundos

% Fun  o de equa  es diferenciais
pendulum_eqs = @(t, y) [y(2); (m * g * l * sin(y(1)) - b * y(2)) /
    J];

% Condi  es iniciais para theta e omega
initial_conditions = [theta0_rad, omega0];

```

```
% Simula  o para a situa  o a)
[T_a, Y_a] = ode45(pendulum_eqs, tspan, initial_conditions);

% Condi  es iniciais para a situa  o b)
T_b = [0, 10];
theta0_deg_b = 30;
theta0_rad_b = deg2rad(theta0_deg_b);
omega_b = 7.7 / J;  % Calculando a velocidade angular inicial correspondente

initial_conditions_b = [theta0_rad_b, omega_b];

% Simula  o para a situa  o b)
[T_b, Y_b] = ode45(pendulum_eqs, tspan, initial_conditions_b);

% Plot dos resultados
figure;
subplot(2, 1, 1);
plot(T_a, rad2deg(Y_a(:, 1)), 'b', 'LineWidth', 2);
title('P ndulo - Situa  o a)');
xlabel('Tempo (s)');
ylabel(' ngulo  (graus)');
grid on;

subplot(2, 1, 2);
plot(T_b, rad2deg(Y_b(:, 1)), 'r', 'LineWidth', 2);
title('P ndulo - Situa  o b)');
xlabel('Tempo (s)');
ylabel(' ngulo  (graus)');
grid on;

sgtitle('Simula  o do P ndulo');
```


Simulação do Pêndulo
Pêndulo - Situação a]
Ângulo (graus)
0
100
200
300
0
1
2
3
4
5
6
7
8
9
10
Tempo (s)
Pêndulo - Situação b]
Ângulo (graus)
0
100
200
300
0
1
2
3
4
5
6
7
8
9
10
Tempo (s)