Geração dos sinais, definição do sistema e execução das simulações

Antes de mais nada, é importante fechar possíveis figuras abertas, apagar as variáveis instanciadas e limpar o prompt

```
clear; close all; clc
```

Constantes

Parâmetros para geração dos gráficos (tamanho da fonte, grossura das linhas, etc.).

```
set(0,'DefaultLineLineWidth', 1.5, ...
    'DefaultAxesFontName', 'Latin Modern Math', ...
    'DefaultAxesFontSize', 20, ...
    'DefaultFigurePosition', [403 914 900 600]);

save_plots = true;
rng(21);  % Importante deixar a seed padrao para ter reprodutibilidade
```

Parâmetros da simulação

```
% dt = 1E-5;  % Para que a gente consiga visualizar bem
% dt = 1E-3;  % Para ir rapido --> diminuir para melhorar resolucao
dt = 1E-4;  % Meio termo
f_sr = 1/dt;
% T = 32;  % Deve ser par
T = 40;  % Deve ser par
% CI = [1; 1; 1; 1];
CI = [0; 0; 0; 0];
```

Parâmetros da planta

```
mv = 2.45;  % [mv] = kg --> Massa de 1/4 do veiculo
% mv = 1.45;  % [mv] = kg --> Massa de 1/4 do veiculo
ks = 900;   % [ks] = N/m --> Constante elastica da suspensao
bs = 7.5;   % [bs] = Ns/m --> Coeficiente de amortecimento da suspensao
mr = 1;     % [mr] = kg --> Massa do eixo-roda
kp = 1250;  % [kp] = N/m --> Constante elastica do pneu
bp = 5;     % [bp] = Ns/m --> Coeficiente de amortecimento do pneu
```

Parâmetros dos sinais de chirp

```
% Parametros gerais do sinal de chirp
T0 = T/2;
f0 = 1/T0;
k1 = 0*(1/f0);  % Fica mais claro do que k1 = 0*T0
k2 = 10*(1/f0);
t = 0:dt:T0;
N = 2;  % Numero de periodos
```

```
% Especificos para cada um
Aw = 0.1;  % [Aw] = m/s
Au = 5;  % [Au] = N
```

Parâmetros do ruídos

```
% Ruido de medicao
% SNR = 20;  % 20dB --> Sinal eh 100x mais potente que o ruido
SNR = 30;  % 30dB --> Sinal eh 1000x mais potente que o ruido

% Ruido de processo
p_max_pn = 0.01;
```

Velocidade dos polos dos observador de Luenberger

Entrada $\omega(t)$

```
pu_r_mult_luenberger_w = [7 8.75 7 8.75];
pu_i_mult_luenberger_w = [0 0 0 0];

p1_r_mult_luenberger_w = [4 4 4 4];
p1_i_mult_luenberger_w = [4 4 4 4];

p2_r_mult_luenberger_w = [4 4 4 4];
p2_i_mult_luenberger_w = [4 4 4 4];
```

Entrada $u(t)$

```
pu_r_mult_luenberger_u = [7 8.75 7 8.75];
pu_i_mult_luenberger_u = [0 0 0 0];

p1_r_mult_luenberger_u = [5.5 5.5 5.5 5.5];
p1_i_mult_luenberger_u = [2.5 2.5 2.5 2.5];

p2_r_mult_luenberger_u = [5.5 5.5 5.5 5.5];
p2_i_mult_luenberger_u = [2.5 2.5 2.5 2.5];
```

Parâmetros do filtro passa-baixas

```
N_filter = 2;   % Ordem do filtro
wc_filter = 75;  % Frequencia de corte; [wc] = rad/s
```

Parâmetros para melhor estimador (obtido após várias análises)

```
best_sim = 'luenberger_u';
best_exct_pos = 'u';
l_mul = 1.375;
r1_limit = l_mul*1.29E-04;  % Obtido no observer_design e multiplicado por
um algum fator...
r2_limit = l_mul*8.72E-02;
```

Scripts

Para facilitar, criou-se funções auxiliares para algumas partes do código. Vamos adicioná-las ao path.

```
addpath(genpath('./functions'))
```

Definição das matrizes do sistema

```
fprintf('Obtendo matrizes do sistema...\n')
```

```
Obtendo matrizes do sistema...
```

```
[A, B1, B2, C, D] = get_matrices(mv, ks, bs, mr, kp, bp);
```

Geração do sinal de *chirp*

```
fprintf('Gerando sinais de chirp...\n')
```

```
Gerando sinais de chirp...
```

Chirp para $\omega(t)$

```
% Sinal gerado
[w, tw] = generate_chirp(Aw, T0, k1, k2, t, N);
W = cov(w);

% Exportando para arquivo
wt = timeseries(w, tw);
save('./data/chirp_w_data.mat', 'wt', '-v7.3')

% Salvando as figuras
%%% No tempo
plot_chirp(tw, w, '\omega(t)', 'm/s', [tw(1), tw(end), -Aw*1.1, Aw*1.1],
save_plots, './figs/chirp/chirp_w.png');
%%% Na frequencia
ii = ceil(length(tw)/2);
w_ = w(1:ii);  % Fazendo assim pois dois concatenados da bug...
plot_fspec(w_, length(w_)*10, dt, [k1*f0, k2*f0+2, -20, 1], true,
save_plots, './figs/chirp/chirp_w_mag.png');
```

Chirp para $u(t)$

```
% Sinal gerado
[u, tu] = generate_chirp(Au, T0, k1, k2, t, N);
% U = cov(u);  % NAO FAZ SENTIDO REALIAR ISTO POIS O SINAL EH CONHECIDO!

% Exportando para arquivo
ut = timeseries(u, tu);
save('./data/chirp_u_data.mat', 'ut', '-v7.3')

% Salvando as figuras
%%% No tempo
```

```
plot_chirp(tu, u, 'u(t)', 'N', [tu(1), tu(end), -Au*1.1, Au*1.1],
save_plots, './figs/chirp/chirp_u.png');
%%% Na frequencia
ii = ceil(length(tu)/2);
u_ = u(1:ii);  % Fazendo assim pois dois concatenados da bug...
plot_fspec(u_, length(u_)*10, dt, [k1*f0, k2*f0+2, -20, 1], true,
save_plots, './figs/chirp/chirp_u_mag.png');
```

Ruído de medição e processo

```
fprintf('Gerando sinais de ruido...\n')
```

```
Gerando sinais de ruido...
```

Ruído de medição para saídas quando entrada $\omega(t)$

```
% Pegando y e os valores maximos para w(t)
sys_w = ss(A, B1, C, 0);  % B1 pois queremos estudar apenas w(t)
y_w = lsim(sys_w, w, tw);
y1_w = y_w(:,1);
y2_w = y_w(:,2);

% Procedimento para gerar matriz de variancia
[v_y1_w, v_var_y1_w] = generate_noise(SNR, y1_w);
[v_y2_w, v_var_y2_w] = generate_noise(SNR, y2_w);

% Gerando a matriz de covariancia
V_w = cov(v_y1_w, v_y2_w);
v_w = [v_y1_w; v_y2_w];
vt_w = timeseries(v_w, tw);
save('./data/measurement_noise_w_data.mat', 'vt_w', '-v7.3')

% Salvando as figuras
%%% Ruido 1
plot_noise(tw, v_w(1,:), [0, 32, 1.1*min(v_w(1,:)), 1.1*max(v_w(1,:))], '1',
'm', 'v', ...
           save_plots, './figs/noise/noise_v1_w.png');
plot_noise_dist(v_w(1,:), 30, '1', 'm', save_plots, './figs/noise/
noise_v1_w_dist.png');
%%% Ruido 2
plot_noise(tw, v_w(2,:), [0, 32, 1.1*min(v_w(2,:)), 1.1*max(v_w(2,:))], '2',
'm/s^2', 'v', ...
           save_plots, './figs/noise/noise_v2_w.png');
plot_noise_dist(v_w(2,:), 30, '2', 'm/s^2', save_plots, './figs/noise/
noise_v2_w_dist.png');
```

Ruído de medição para saídas quando entrada $u(t)$

```
% Pegando y e os valores maximos para u(t)
sys_u = ss(A, B2, C, D);  % B2 pois queremos estudar apenas u(t)
[y_u, ~, x_u] = lsim(sys_u, u, tu);
```

```
y1_u = y_u(:,1);
y2_u = y_u(:,2);
% Procedimento para gerar matriz de variancia
[v_y1_u, v_var_y1_u] = generate_noise(SNR, y1_u);
[v_y2_u, v_var_y2_u] = generate_noise(SNR, y2_u);
% Gerando a matriz de covariancia
V_u = cov(v_y1_u, v_y2_u);
v_u = [v_y1_u; v_y2_u];
vt_u = timeseries(v_u, tu);
save('./data/measurement_noise_u_data.mat', 'vt_u', '-v7.3')

% Salvando as figuras
%%% Ruido 1
plot_noise(tu, v_u(1,:), [0, T, 1.1*min(v_u(1,:)), 1.1*max(v_u(1,:))], '1',
'm', 'v', ...
           save_plots, './figs/noise/noise_v1_u.png');
plot_noise_dist(v_u(1,:), 30, '1', 'm', save_plots, './figs/noise/
noise_v1_u_dist.png');
%%% Ruido 2
plot_noise(tu, v_u(2,:), [0, T, 1.1*min(v_u(2,:)), 1.1*max(v_u(2,:))], '2',
'm/s^2', 'v', ...
           save_plots, './figs/noise/noise_v2_u.png');
plot_noise_dist(v_u(2,:), 30, '2', 'm/s^2', save_plots, './figs/noise/
noise_v2_u_dist.png');
```

Ruído de processo

```
% Gerando ruido de processo
v_pu = (p_max_pn*max(u)).*randn(length(x_u), 1);
vt_pu = timeseries(v_pu, tu);
U = cov(v_pu);
save('./data/process_noise_u_data.mat', 'vt_pu', '-v7.3')

% Figura
plot_noise(tu, v_pu, [0, T, 1.1*min(v_pu), 1.1*max(v_pu)], 'u', 'N', 'v', ...
           save_plots, './figs/noise/process_noise.png');
plot_noise_dist(v_pu, 30, '', 'N', save_plots, './figs/noise/
process_noise_dist.png');
```

Projeto dos observadores e do filtro

```
fprintf('Projetando observadores...\n')
```

```
Projetando observadores...
```

Obtenção dos polos da planta e definição das matrizes específicas para cada observador do banco de observadores

```
p = eig(A);  % Polos da planta
C_obs1 = C(1, :);  % Matriz de saida do observador 1
D_obs1 = D(1, :);
C_obs2 = C(2, :);  % Matriz de saida do observador 2
```

```
D_obs2 = D(2, :);
```

Projeto do observador de Luenberger (polos desejados obtidos anteriomente por meio de tentativa e erro)

```
% Entrada w
%%% Observador unico
pu_luenberger_w = pu_r_mult_luenberger_w.*real(eig(A))' +
1j*pu_i_mult_luenberger_w.*imag(eig(A))';
Kou_luenberger_w = place(A', C', pu_luenberger_w)';
%%% Banco de Observadores
p1_luenberger_w = p1_r_mult_luenberger_w.*real(eig(A))' +
1j*p1_i_mult_luenberger_w.*imag(eig(A))';
p2_luenberger_w = p2_r_mult_luenberger_w.*real(eig(A))' +
1j*p2_i_mult_luenberger_w.*imag(eig(A))';
Ko1_luenberger_w = place(A', C_obs1', p1_luenberger_w)';
Ko2_luenberger_w = place(A', C_obs2', p2_luenberger_w)';

% Entrada u
%%% Observador unico
pu_luenberger_u = pu_r_mult_luenberger_u.*real(eig(A))' +
1j*pu_i_mult_luenberger_u.*imag(eig(A))';
Kou_luenberger_u = place(A', C', pu_luenberger_u)';
%%% Banco de Observadores
p1_luenberger_u = p1_r_mult_luenberger_u.*real(eig(A))' +
1j*p1_i_mult_luenberger_u.*imag(eig(A))';
p2_luenberger_u = p2_r_mult_luenberger_u.*real(eig(A))' +
1j*p2_i_mult_luenberger_u.*imag(eig(A))';
Ko1_luenberger_u = place(A', C_obs1', p1_luenberger_u)';
Ko2_luenberger_u = place(A', C_obs2', p2_luenberger_u)';
```

Projeto do filtro de Kalman

```
% Entrada w
%%%%% Observador unico
[Kou_kalman_w, ~, ~] = lqe(A, B1, C, W, V_w);  % Agora eh V porque tem a
matriz de covariancia...
%%% Banco de Observadores
[Ko1_kalman_w, ~, ~] = lqe(A, B1, C_obs1, W, v_var_y1_w);  % Tem que ser
v_var_y1 porque eh da saida 1 apenas
[Ko2_kalman_w, ~, ~] = lqe(A, B1, C_obs2, W, v_var_y2_w);  % Idem

% Entrada u
%%% Observador unico
[Kou_kalman_u, ~, ~] = lqe(A, B2, C, U, V_u);  % Agora eh V porque tem a
matriz de covariancia...
%%% Banco de Observadores
[Ko1_kalman_u, ~, ~] = lqe(A, B2, C_obs1, U, v_var_y1_u);  % Tem que ser
v_var_y1 porque eh da saida 1 apenas
[Ko2_kalman_u, ~, ~] = lqe(A, B2, C_obs2, U, v_var_y2_u);  % Idem
```

Projeto do filtro passa-baixas

```
fprintf('Projetando filtro butterworth...\n')
```

```
Projetando filtro butterworth...
```

Obtendo coeficientes do filtro e exemplificando uso

```
% Obtençao do filtro
[b_filter, a_filter] = butter(N_filter, wc_filter/(f_sr*pi), 'low');   % wc
= 25 + wn2 (padrao)
[H_filter, w_filter] = freqz(b_filter, a_filter, 1E6);
plot_filter_bode(w_filter, H_filter, f_sr, [1E-1, 1E4, -60, 0], [1E-1, 1E4,
-180, 0], ...
                 save_plots, './figs/bode/filter_mag_bode.png', './figs/bode/
filter_phase_bode.png');

% Exemplo de aplicacao (sintetico)
t_ex = 0:1E-3:10;
y_ex = 2*(t_ex-t_ex(1))/(t_ex(end)-t_ex(1)).*sin((20*pi/5).*t_ex) +
0.60*randn(1, length(t_ex));
plot_denoiser_ex(t_ex, y_ex, b_filter, a_filter, [t_ex(1) t_ex(end) -2.5
2.5], save_plots, ...
                 './figs/noise/filtering_ex1.png', ...
                 './figs/noise/filtering_ex2.png');
```

Motivação de ter realizado a otimização

```
% Exemplo de observador ruim (motivacao para otimizacao) --> Melhor
organizaçao dps
Kou_luenberger_w_ = Kou_luenberger_w;
Kou_luenberger_w = place(A', C', 6*eig(A))';
bs_ = 0.5*bs;
[A_sim_case, B1_sim_case, B2_sim_case, C_sim_case, D_sim_case] =
get_matrices(mv, ks, bs_, mr, kp, bp);
[t_luenberger_w, ~, y_luenberger_w, ~, ~, ~, y_hat_obsu_luenberger_w, ~, ~]
= run_sim('./simulations', 'luenberger_w');
plot_bad_obs_residue(t_luenberger_w, y_luenberger_w,
y_hat_obsu_luenberger_w, b_filter, a_filter, ...
                     true, save_plots, './figs/residue/bad_obs_ex.png');
Kou_luenberger_w = Kou_luenberger_w_;
```

Plotando influência da variação dos parâmetros nas saídas

```
fprintf('Gerando graficos de bode de acordo com variacao de parametros...\n')
```

```
Gerando graficos de bode de acordo com variacao de parametros...
```

```
percentual_range = linspace(0, 1, 11);
colors = ["#BC0000", "#FF595E", "#FF924C", "#FFCA3A", "#C5CA30", "#8AC926",
...
          "#36949D", "#1982C4", "#4267AC", "#565AA0", "#6A4C93"];
```

```
% Variando bs
plot_sys_bode_var(mv, ks, bs, mr, kp, bp, 'bs', 0:0.1:1, k1*f0+eps, k2*f0 +
5, colors, ...
                  [0 60 -35 10], [0 60 15 60], [0 60 -80 -30], [0 60 -25
20], save_plots, ...
                  './figs/bode/bode_y1_bs_var_w.png', './figs/bode/
bode_y2_bs_var_w.png', ...
                  './figs/bode/bode_y1_bs_var_u.png', './figs/bode/
bode_y2_bs_var_u.png');
% Variando ks
plot_sys_bode_var(mv, ks, bs, mr, kp, bp, 'ks', percentual_range, k1*f0+eps,
k2*f0 + 5, colors, ...
                  [0 60 -35 -5], [0 60 0 50], [0 60 -80 -30], [0 60 -25 10],
save_plots, ...
                  './figs/bode/bode_y1_ks_var_w.png', './figs/bode/
bode_y2_ks_var_w.png', ...
                  './figs/bode/bode_y1_ks_var_u.png', './figs/bode/
bode_y2_ks_var_u.png');
```

Simulação para cada caso de dano

```
fprintf('Rodando as simulaçoes...\n')
```

```
Rodando as simulaçoes...
```

Situaçoes de analise

```
% Caso 0: Sem dano
[A_c0, B1_c0, B2_c0, C_c0, D_c0] = get_matrices(mv, ks, bs, mr, kp, bp);

% Caso 1: Perda do amortecimento da suspensao de 50%
bs_ = 0.5*bs;
[A_c1, B1_c1, B2_c1, C_c1, D_c1] = get_matrices(mv, ks, bs_, mr, kp, bp);

% Caso 2: Perda da rigidez da mola da suspensao de 50%
ks_ = 0.5*ks;
[A_c2, B1_c2, B2_c2, C_c2, D_c2] = get_matrices(mv, ks_, bs, mr, kp, bp);

% Caso 3: Perda de amortecimento da suspensao e da rigidez da mola de 50%
bs_ = 0.5*bs; ks_ = 0.5*ks;
[A_c3, B1_c3, B2_c3, C_c3, D_c3] = get_matrices(mv, ks_, bs_, mr, kp, bp);
```

Execução das simulações

```
for sim_case = ['1', '2', '3']  % Cada caso da simulacao
    A_sim_case = eval(sprintf('A_c%s', sim_case));
    B1_sim_case = eval(sprintf('B1_c%s', sim_case));
    B2_sim_case = eval(sprintf('B2_c%s', sim_case));
    C_sim_case = eval(sprintf('C_c%s', sim_case));
    D_sim_case = eval(sprintf('D_c%s', sim_case));
```

```
    % Simulaçao dos observadores
    %%% Entrada w
    [t_luenberger_w, ~, y_luenberger_w, ~, ~, ~,
y_hat_obsu_luenberger_w, y1_hat_obs1_luenberger_w, y2_hat_obs2_luenberger_w]
= run_sim('./simulations', 'luenberger_w');
    [t_kalman_w, ~, y_kalman_w, ~, ~, ~, y_hat_obsu_kalman_w,
y1_hat_obs1_kalman_w, y2_hat_obs2_kalman_w] = run_sim('./simulations',
'kalman_w');
    plot_residue(t_luenberger_w, y_luenberger_w, y_hat_obsu_luenberger_w,
y1_hat_obs1_luenberger_w, ...
                 y2_hat_obs2_luenberger_w, y_kalman_w, y_hat_obsu_kalman_w,
y1_hat_obs1_kalman_w, ...
                 y2_hat_obs2_kalman_w, sim_case, b_filter, a_filter, true,
save_plots, sprintf('./figs/residue/residue1_w_c%s.png', sim_case), ...
                 sprintf('./figs/residue/residue2_w_c%s.png', sim_case));
    print_residue_rmse(t_luenberger_w, y_luenberger_w,
y_hat_obsu_luenberger_w, y1_hat_obs1_luenberger_w, ...
                       y2_hat_obs2_luenberger_w, y_kalman_w,
y_hat_obsu_kalman_w, y1_hat_obs1_kalman_w, ...
                       y2_hat_obs2_kalman_w, sim_case, 'w', ...
                       b_filter, a_filter)

    %%% Entrada u
    [t_luenberger_u, ~, y_luenberger_u, ~, ~, ~,
y_hat_obsu_luenberger_u, y1_hat_obs1_luenberger_u, y2_hat_obs2_luenberger_u]
= run_sim('./simulations', 'luenberger_u');
    [t_kalman_u, ~, y_kalman_u, ~, ~, ~, y_hat_obsu_kalman_u,
y1_hat_obs1_kalman_u, y2_hat_obs2_kalman_u] = run_sim('./simulations',
'kalman_u');
    plot_residue(t_luenberger_u, y_luenberger_u, y_hat_obsu_luenberger_u,
y1_hat_obs1_luenberger_u, ...
                 y2_hat_obs2_luenberger_u, y_kalman_u, y_hat_obsu_kalman_u,
y1_hat_obs1_kalman_u, ...
                 y2_hat_obs2_kalman_u, sim_case, b_filter, a_filter, true,
save_plots, sprintf('./figs/residue/residue1_u_c%s.png', sim_case), ...
                 sprintf('./figs/residue/residue2_u_c%s.png', sim_case));
    print_residue_rmse(t_luenberger_u, y_luenberger_u,
y_hat_obsu_luenberger_u, y1_hat_obs1_luenberger_u, ...
                       y2_hat_obs2_luenberger_u, y_kalman_u,
y_hat_obsu_kalman_u, y1_hat_obs1_kalman_u, ...
                       y2_hat_obs2_kalman_u, sim_case, 'u', ...
                       b_filter, a_filter)
end
```

```
-----------  Caso 1 | Entrada w ----------
================= Residuo 1 =================
---------------- Luenberger -----------------
- OBSU: 2.12e-04    1.69e-03
- OBS1: 1.52e-04    1.85e-04
-----------------------------------------------
------------------ Kalman --------------------
- OBSU: 6.78e-05    4.15e-04
```

```
- OBS1: 1.85e-04    2.09e-04
------------------------------------------------
================================================
================== Residuo 2 ==================
---------------- Luenberger ------------------
- OBSU: 2.20e-01    1.11e+00
- OBS2: 6.17e-02    6.98e-02
------------------------------------------------
------------------ Kalman --------------------
- OBSU: 6.46e-02    6.70e-02
- OBS2: 6.89e-02    7.52e-02
------------------------------------------------
================================================
================================================
=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
=-=-=-=-=-=-  Caso 1 | Entrada u -=-=-=-=-=-=-=
================== Residuo 1 ==================
---------------- Luenberger ------------------
- OBSU: 4.17e-05    1.80e-03
- OBS1: 4.65e-06    1.42e-04
------------------------------------------------
------------------ Kalman --------------------
- OBSU: 9.12e-06    1.15e-03
- OBS1: 9.12e-06    1.26e-03
------------------------------------------------
================================================
================== Residuo 2 ==================
---------------- Luenberger ------------------
- OBSU: 2.94e-02    1.03e+00
- OBS2: 1.64e-03    5.18e-02
------------------------------------------------
------------------ Kalman --------------------
- OBSU: 3.21e-03    4.22e-01
- OBS2: 3.23e-03    4.77e-01
------------------------------------------------
================================================
=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
=-=-=-=-=-=-  Caso 2 | Entrada w -=-=-=-=-=-=-=
================== Residuo 1 ==================
---------------- Luenberger ------------------
- OBSU: 2.12e-04    1.33e-02
- OBS1: 1.52e-04    1.16e-04
------------------------------------------------
------------------ Kalman --------------------
- OBSU: 6.78e-05    2.28e-03
- OBS1: 1.85e-04    3.05e-04
------------------------------------------------
================================================
================== Residuo 2 ==================
---------------- Luenberger ------------------
- OBSU: 2.20e-01    9.09e+00
- OBS2: 6.17e-02    2.91e-02
------------------------------------------------
------------------ Kalman --------------------
- OBSU: 6.46e-02    3.33e-01
- OBS2: 6.89e-02    5.97e-02
------------------------------------------------
================================================
=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
=-=-=-=-=-=-  Caso 2 | Entrada u -=-=-=-=-=-=-=
================== Residuo 1 ==================
---------------- Luenberger ------------------
- OBSU: 4.17e-05    1.85e-02
- OBS1: 4.65e-06    3.98e-04
------------------------------------------------
```

```
------------------- Kalman --------------------
- OBSU: 9.12e-06    6.03e-03
- OBS1: 9.12e-06    6.47e-03
-----------------------------------------------
===============================================
================== Residuo 2 ==================
---------------- Luenberger ------------------
- OBSU: 2.94e-02    1.33e+01
- OBS2: 1.64e-03    8.32e-02
-----------------------------------------------
------------------- Kalman --------------------
- OBSU: 3.21e-03    1.14e+00
- OBS2: 3.23e-03    1.37e+00
-----------------------------------------------
===============================================
=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
=-=-=-=-=-=-  Caso 3 | Entrada w -=-=-=-=-=-=
================== Residuo 1 ==================
---------------- Luenberger ------------------
- OBSU: 2.12e-04    1.81e-02
- OBS1: 1.52e-04    1.30e-04
-----------------------------------------------
------------------- Kalman --------------------
- OBSU: 6.78e-05    3.10e-03
- OBS1: 1.85e-04    3.73e-04
-----------------------------------------------
===============================================
================== Residuo 2 ==================
---------------- Luenberger ------------------
- OBSU: 2.20e-01    1.24e+01
- OBS2: 6.17e-02    2.66e-02
-----------------------------------------------
------------------- Kalman --------------------
- OBSU: 6.46e-02    4.52e-01
- OBS2: 6.89e-02    6.95e-02
-----------------------------------------------
===============================================
=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
=-=-=-=-=-=-  Caso 3 | Entrada u -=-=-=-=-=-=
================== Residuo 1 ==================
---------------- Luenberger ------------------
- OBSU: 4.17e-05    2.34e-02
- OBS1: 4.65e-06    5.41e-04
-----------------------------------------------
------------------- Kalman --------------------
- OBSU: 9.12e-06    8.34e-03
- OBS1: 9.12e-06    8.94e-03
-----------------------------------------------
===============================================
================== Residuo 2 ==================
---------------- Luenberger ------------------
- OBSU: 2.94e-02    1.66e+01
- OBS2: 1.64e-03    1.12e-01
-----------------------------------------------
------------------- Kalman --------------------
- OBSU: 3.21e-03    1.54e+00
- OBS2: 3.23e-03    1.83e+00
-----------------------------------------------
===============================================
=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
```

Melhor estimador e forma de excitação (após análise)

```
fprintf('Gerando residuos com limiares para o melhor sistema...\n')
```

```
Gerando residuos com limiares para o melhor sistema...
```

```
sim_case = '0';
A_sim_case = eval(sprintf('A_c%s', sim_case));
B1_sim_case = eval(sprintf('B1_c%s', sim_case));
B2_sim_case = eval(sprintf('B2_c%s', sim_case));
C_sim_case = eval(sprintf('C_c%s', sim_case));
D_sim_case = eval(sprintf('D_c%s', sim_case));
[t_best, ~, y_best, ~, ~, ~, y_hat_obsu_best, y1_hat_obs1_best,
y2_hat_obs2_best] = run_sim('./simulations', best_sim);

if best_exct_pos == "u"
    y_hat_best = y_hat_obsu_best;
else
    y_hat_best = struct;
    y_hat_best.y1_hat = y1_hat_obs1_best.y1_hat;
    y_hat_best.y2_hat = y2_hat_obs2_best.y2_hat;
end

plot_best_obs_residue(t_best, y_best, y_hat_best, sim_case, r1_limit, ...
    r2_limit, [t_best(1) t_best(ceil(length(t_best)/2)), -r1_limit*1.1, ...
    r1_limit*1.1], [t_best(1) t_best(ceil(length(t_best)/2)), -r2_limit*1.1,
...
    r2_limit*1.1], b_filter, a_filter, false, save_plots, ...
    './figs/residue/residue1_limiar_ex.png', ...
    './figs/residue/residue2_limiar_ex.png');

for sim_case = ['1', '2', '3']
    A_sim_case = eval(sprintf('A_c%s', sim_case));
    B1_sim_case = eval(sprintf('B1_c%s', sim_case));
    B2_sim_case = eval(sprintf('B2_c%s', sim_case));
    C_sim_case = eval(sprintf('C_c%s', sim_case));
    D_sim_case = eval(sprintf('D_c%s', sim_case));

    [t_best, ~, y_best, ~, ~, ~, y_hat_obsu_best, y1_hat_obs1_best,
y2_hat_obs2_best] = run_sim('./simulations', best_sim);

    if best_exct_pos == "u"
        y_hat_best = y_hat_obsu_best;
    else
        y_hat_best = struct;
        y_hat_best.y1_hat = y1_hat_obs1_best.y1_hat;
        y_hat_best.y2_hat = y2_hat_obs2_best.y2_hat;
    end

    plot_best_obs_residue(t_best, y_best, y_hat_best, sim_case, r1_limit, ...
                          r2_limit, "auto", "auto", b_filter, a_filter,
true, save_plots, ...
```

```
                            sprintf('./figs/residue/residue1_limiar_c%s.png',
sim_case), ...
                            sprintf('./figs/residue/residue2_limiar_c%s.png',
sim_case));
end
```

Validando para situações menos críticas (variação de 10%)

```
% Caso 4: Perda do amortecimento da suspensao de 10%
bs_ = 0.9*bs;
[A_c4, B1_c4, B2_c4, C_c4, D_c4] = get_matrices(mv, ks, bs_, mr, kp, bp);

% Caso 5: Perda da rigidez da mola da suspensao de 10%
ks_ = 0.9*ks;
[A_c5, B1_c5, B2_c5, C_c5, D_c5] = get_matrices(mv, ks_, bs, mr, kp, bp);

% Caso 6: Perda de amortecimento da suspensao e da rigidez da mola de 10%
bs_ = 0.9*bs; ks_ = 0.9*ks;
[A_c6, B1_c6, B2_c6, C_c6, D_c6] = get_matrices(mv, ks_, bs_, mr, kp, bp);

for sim_case = ['4', '5', '6']
    A_sim_case = eval(sprintf('A_c%s', sim_case));
    B1_sim_case = eval(sprintf('B1_c%s', sim_case));
    B2_sim_case = eval(sprintf('B2_c%s', sim_case));
    C_sim_case = eval(sprintf('C_c%s', sim_case));
    D_sim_case = eval(sprintf('D_c%s', sim_case));

    [t_best, ~, y_best, ~, ~, ~, y_hat_obsu_best, y1_hat_obs1_best,
y2_hat_obs2_best] = run_sim('./simulations', best_sim);

    if best_exct_pos == "u"
        y_hat_best = y_hat_obsu_best;
    else
        y_hat_best = struct;
        y_hat_best.y1_hat = y1_hat_obs1_best.y1_hat;
        y_hat_best.y2_hat = y2_hat_obs2_best.y2_hat;
    end

    plot_best_obs_residue(t_best, y_best, y_hat_best, sim_case, r1_limit, ...
                            r2_limit, "auto", "auto", b_filter, a_filter,
true, save_plots, ...
                            sprintf('./figs/residue/residue1_limiar_c%s.png',
sim_case), ...
                            sprintf('./figs/residue/residue2_limiar_c%s.png',
sim_case));
end
```