Instituto Superior Técnico

Controlo por Computador

Identification and Computer Control of a Flexible Robot Arm Joint
Parte 1

Mestrado Integrado em Engenharia Aeroespacial

Grupo 14

Pedro Sarnadas, 86673
Renato Loureiro, 89708
Tiago Santos, 87290

2020/21

Índice

1 Introdução

Este relatório foi realizado no âmbito da cadeira Controlo por Computador com objectivo de aplicar num contexto "real"os conceitos teóricos leccionados sobre identificação de parâmetros de modelos e desenvolvimento de controladores de sistemas.

No caso particular em estudo, o sistema a modelar consistiu numa barra flexível cuja posição da ponta se pretende controlar através de inputs de tensão no motor DC.

O presente documento aborda a fase de identificação e modelação do modelo e os procedimentos, técnicas e critérios usados nesta fase inicial do projecto. Como foco de analise estiveram as 3 questões propostas no guia laboratorial - **Q1**, **Q2** e **Q3**, cujas suas respostas são apresentadas ao longo do relatório. De forma a facilitar a leitura cada resposta está devidamente assinalada no título da sua respectiva secção.

2 Controlo não trivial - Q1

O sistema que pretendemos modelar e controlar revela-se à priori não trivial e complexo - a relação entre o input e o output, tensão do motor DC e a posição da ponta da barra, respectivamente, não apresenta a propriedade de um sistema BIBO, 'Bounded input Bounded Output', dado que para um sistema com entrada (tensão) diferente de zero, o output do sistema nunca tenderá para um valor constante, mas sim para o infinito - neste caso a variável correspondente ao output é a posição angular da ponta da barra, o que significa que a anterior afirmação de o output do sistema tender para infinito corresponde, na verdade, na nossa planta de estudo à barra estar sempre a rodar.

A relação entre a tensão do motor DC e a posição angular do rotor do motor corresponde a uma derivada de primeira ordem: $tensão = \frac{d(posição)}{dt}$, ou seja coloca no nosso sistema um pólo na origem.

Para além das características complexas anteriormente apresentadas, a barra não corresponde a um corpo rígido, estando-lhe inerente uma propriedade elástica que implica oscilações no sistema em estudo e efeito de chicotada - que ocorre quando há inversão do sentido de rotação ou quando o sistema está estático e apresenta um novo input, e barra, incapaz de acompanhar na sua totalidade a variação, verifica uma velocidade no sentido oposto ao sentido desejado. Este fenómeno leva a concluir que o sistema é uma sistema de fase não mínima, o que corresponde à existência de zeros no semiplano complexo direito.

Das afirmações anteriores verifica-se imediatamente a não trivialidade do sistema em estudo e constata-se que a aplicação de um controlo proporcional no sistema não é capaz de o controlar, pois o sistema torna-se instável com o pólo na origem a deslocar-se para a direita devido à existência do zero no semiplano complexo direito (sistema ser de fase não mínima).

De modo a compreender o resultado final expectável da modelação do sistema, construiu-se uma estimativa da função de transferência do mesmo com as considerações anteriormente mencionadas: a existência de um zero no semiplano complexo direito (atribuindo-se como estimativa o valor 5), a presença do pólo na origem e o efeitos dos modos oscilatórios.

Para o comportamento oscilatório considerou-se a existência de dois modos com diferentes características correspondentes a dois pares de pólos complexos conjugados, um par com $\xi = 0.01$ e $\omega = 50 rad/s$ equivalente a um modo oscilatório pouco amortecido de maior frequência e um outro par com $\xi = 0.5$ e $\omega = 10$, correspondente a uma resposta oscilatória mais amortecida e com frequência menor.

A função de transferência da planta estimada considerando as características anteriormente mencionadas está presente na Equação [1]:

$$H(s) = \frac{s-5}{s(s^2+s+2500)(s^2+10s+100)} \tag{1}$$

Para compreender melhor o resultado esperado apresenta-se também o root locus do sistema e a resposta em frequência do mesmo, estes estão presentes nas Figuras [1] e [2]:

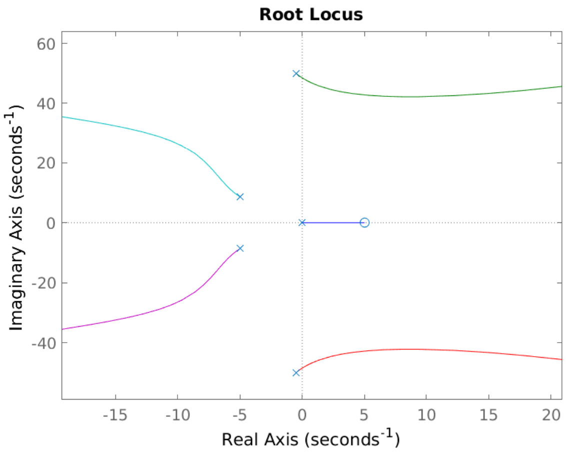

Figura 1

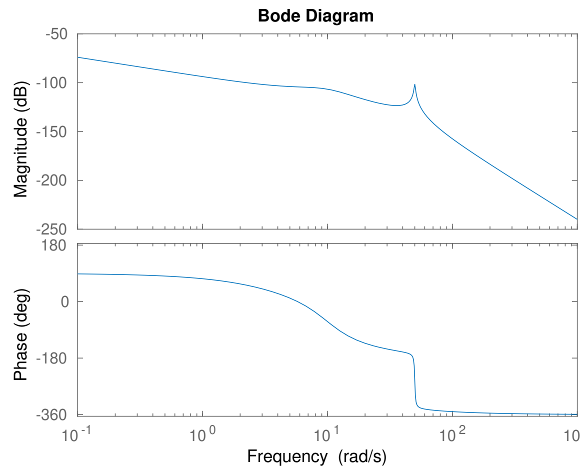

Figura 2

Por uma questão de curiosidade efetuou-se a conversão da função de transferencia anterior para a função de transferencia no dominio do discreto considerando um conversor D/A de ordem zero, um retentor A/D ideal e um periodo de amostragem de T=0.01s:

$$H(z) = \frac{4 \cdot 10^{-10} z^4 + 3.781 \cdot 10^{-9} z^3 - 3.469 \cdot 10^{-10} z^2 - 3.916 \cdot 10^{-9} z - 3.821 \cdot 10^{-10}}{z^5 - 4.642 z^4 + 8.847 z^3 - 8.662 z^2 + 4.353 z - 0.8958} \tag{2}$$

3 Resposta do sistema a um Step - Q2

Para verificar as características gerais do sistema estudou-se a resposta deste a um step de diferentes amplitudes - Figura [3]. Constatou-se que o sistema, como era esperado e explicitado na secção anterior, não estabiliza, devido à dinâmica que o motor confere ao sistema - colocação de um pólo na origem do sistema. Afere-se que o output do sistema (no infinito) tende a variar de forma proporcional ao valor da entrada, isto corresponde a dizer que a derivada do sistema é constante no infinito diferente de zero para um input diferente de zero.

Avaliou-se também a linearidade deste sistema, aplicando vários inputs e verificando os correspondentes outputs com o objetivo de certificar ou validar a veracidade da propriedade de linearidade presente na equação [3]. Após vários testes verificou-se que o sistema em estudo detém propriedades lineares e que não ocorre saturações do sistema.

$$f(u_1) = y_1, \quad f(u_2) = y_2 \quad \Rightarrow \quad f(u_1 + u_2) = y_1 + y_2 \tag{3}$$

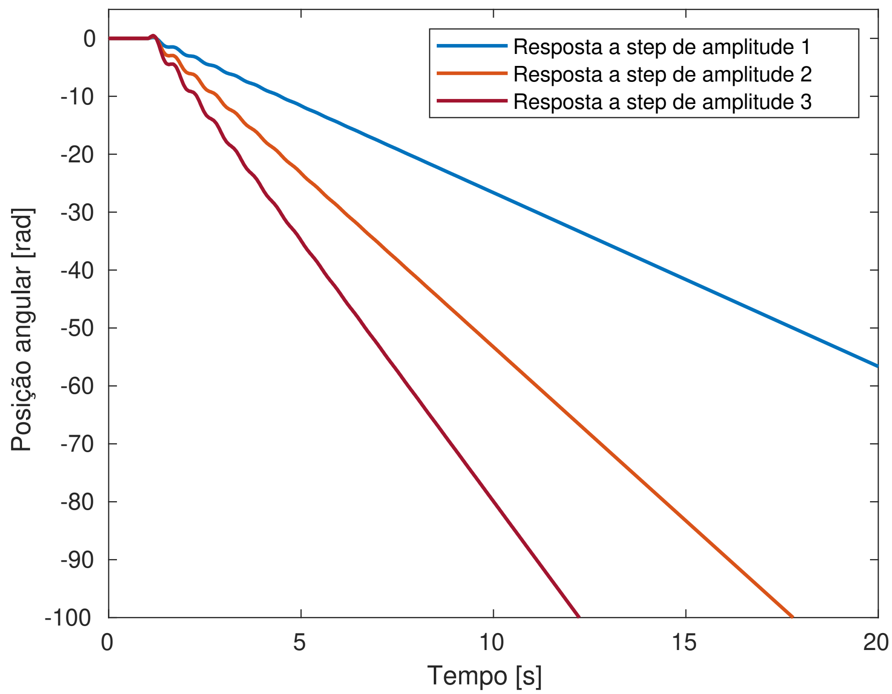

Figura 3: Resposta do sistema a Steps de várias amplitudes

4 Determinação do Modelo do sistema - Q3

4.1 Recolha de Dados

Na fase de aquisição de dados avaliou-se o comportamento do sistema à entrada de dois tipos sinais: quadrado e PRBS (pseudorandom binary sequence). A experimentação foi feita com sinais de mesma amplitude, mas com diferentes frequências para os sinais quadráticos e diferentes parâmetros B (equivalente à *switiching frequency*) para os PRBS's.

A natureza pseudo-aleatória do sinal PRBS permite prever que os resultados recolhidos para este tipo de sinal devem criar melhores modelos ARMAX. O factor cíclico constante dos sinais quadrados não permite avaliar tão bem as características do sistema.

Amplitude

Na aquisição de dados a amplitude usada foi constante já que a aplicação de diferentes amplitudes apenas modifica a rapidez de alteração do estado de saída, y_s. A amplitude do sinal de entrada afeta linearmente a derivada da deflexão (velocidade angular). Isto significa que um output, y resultante da soma de inputs - $\sum u_i$, é igual à soma dos outputs - $\sum y_i$, deles resultantes.

Esta conclusão é uma consequência do facto do modelo simulação fornecido ser linear invariante no tempo e não possuir saturações ou outro tipo de não linearidades.

Se se trabalhasse com o sistema real verificar-se-ia uma situação distinta. Os motores reais não teriam o comportamento idealizado devido à presença de resistências internas, fricção e efeito de *backlash* na caixa de velocidades (*gearbox clearence*) que para amplitudes muito baixas impediriam a barra de se deslocar (*deadzone*) e adicionariam adicionam efeitos não lineares para amplitudes elevadas.

Duração

A duração dos testes em sistema Simulink é pouco importante, mas se se trabalhasse com o sistema real ter-se-ia de fazer uma experiência longa o suficiente para se poder fazer *burn-in* dos momentos experimentais iniciais (5 a 10 segundos) já que estes são contaminados pelo transiente mecânico. Com isto dito a duração dos testes escolhida foi de 100 segundos.

Frequência

A escolha da frequência e parâmetro B tem de ter em consideração que variações constantes muito rápidas mantem o sistema sempre em regime transiente e não permitem avaliar o sistema corretamente. O *switch rate* usado nos sinais de entrada tem de ser tal que permita analisar o regime transiente e inicio do comportamento *steady-state*.

4.2 Frequência de Amostragem

Na discretização dos sinais é necessário não perder porções importantes do comportamento do sistema. Por isso a frequência de amostragem tem ser rápida o suficiente para capturar o transiente e a dinâmica do sistema.

Simultaneamente a frequência também não deve ser demasiado rápida devido ao ruído (de alta frequência) que uma vez detectado contamina os resultados obtidos. Este problema não se põe, porem, já que se está a trabalhar com um sistema simulado sem ruído. Assim a frequência de amostragem seleccionada foi a recomendada de 100Hz. Recorda-se que a frequência de amostragem afecta a localização dos pólos do sistema discreto e por isso a dinâmica do sistema. Mas que a estabilidade ou instabilidade não é afectada.

4.3 Filtragem - remoção do efeito integral

O controlo do motor é feito em termos da velocidade angular através da medição do ângulo da barra, isto é, o motor possui um efeito integrador (pólo na origem). A existência deste pólo é problemática significando que entradas constantes resultam numa variação (aumento/diminuição) constante de θ. Esta característica confere ao sistema uma estabilidade criticamente estável (quasi-instavel) – bounded input resulta em unbounded output.

A aplicação das ferramentas de identificação de sistemas deve ser feita para casos estáveis. Logo na construção do modelo tem de se fazer uma abordagem do tipo:

- Remoção do efeito integral nos dados,
- Identificação do Modelo (relação do ângulo da barra em função da excitação eletrica),
- Adição de um Integrador (pólo z=1, em domínio discreto).

A remoção do efeito integrador é feita por derivação dos dados, com um termo: $(1 - z^{-1})x(z)$.

No entanto esta operação poderia ter consequências no ruído de alta frequência pelo que se adiciona também um filtro Passa-Baixo.

Estas duas operações - derivação e filtragem – podem ser aplicadas simultaneamente usando a função filter do Matlab com o filtro sugerido no enunciado:

$$H(z) = \frac{(1-\lambda)(1-z^{-1})}{1-\lambda z^{-1}} \tag{4}$$

Este processo de filtragem requer a definição do parâmetro lambda, λ. O seu valor de varia entre 0 e a unidade e define a extensão da filtragem feita aos dados. A sua escolha deve ter em consideração o *noise* e a resposta temporal do sinal filtrado. A eliminação dos efeitos do ruido não pode resultar num sinal filtrado que tenha perdido as qualidades características do sistema "real". Neste caso particular, isto significa, não perder o efeito de chicotada (*whiplash*) ou a dinâmica oscilatória principal.

Para determinar o valor de λ, a resposta do sinal filtrado a uma mesma onda quadrática de 0.06Hz foi comparada com a "derivada aproximada" (função "diff") do sinal de saída y_s (deflexão total). Nesta análise registou-se o sinal filtrado para diferentes valores de lambda - sinal filtrado a azul e sinal derivado a laranja. Os resultados são apresentados nos seguintes gráficos:

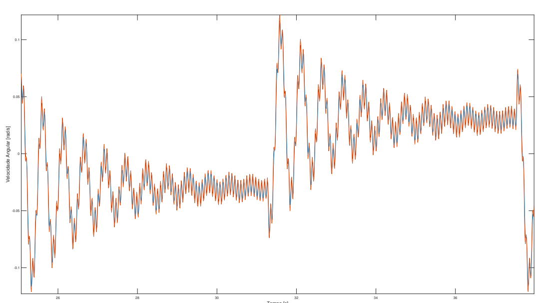

Figura 4: $\lambda = 0.4$

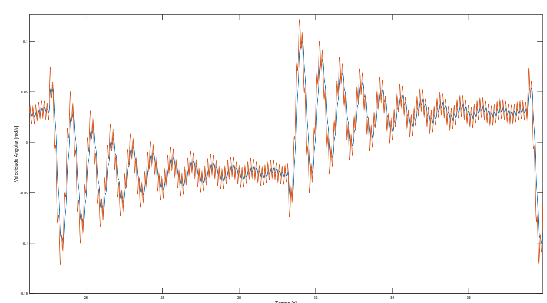

Figura 5: $\lambda = 0.8$

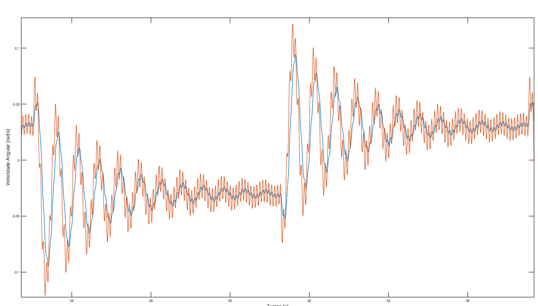

Figura 6: $\lambda = 0.85$

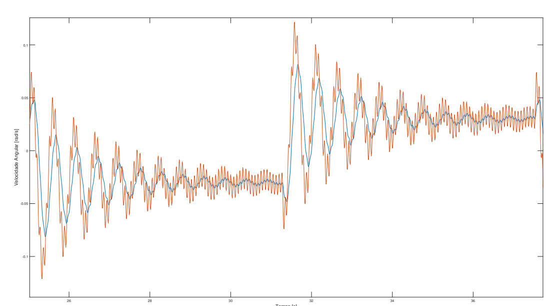

Figura 7: $\lambda = 0.9$

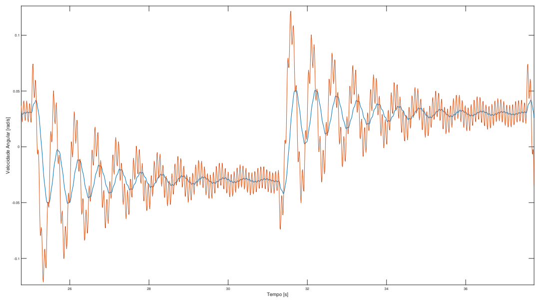

Figura 8: $\lambda = 0.95$

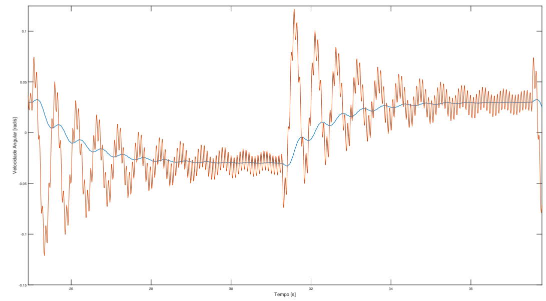

Figura 9: $\lambda = 0.99$

Os resultados para os valores 0.4 e 0.6 são pouco filtrados, não existindo grande alteração da amplitude ou forma do sinal. Para lambdas mais elevados a partir de 0.90 verifica-se o contrário: a onda filtrada começa a perder o comportamento característico - o efeito “whiplash” e eventualmente a dinâmica oscilatória do sistema.

Estas conclusões levaram a escolha de um lambda intermédio, sendo o valor selecionado - 0.8.

4.4 Análise de dados de simulação e modelação

No processo de geração do modelo, escolheu-se entre um sinal do tipo PRBS ou SQUARE e a respetiva frequência (parâmetro B no caso de PRBS) e também escolheu-se os parâmetros do modelo ARMAX - $\mathbf{n_a}$, $\mathbf{n_b}$, $\mathbf{n_c}$ e $\mathbf{n_k}$. Para facilitar a análise de dados, gerou-se para cada sinal de input, o melhor modelo gerado pelo ARMAX para diferentes valores de $\mathbf{n_a}$ - Na Tabela [1], [2], [4] e [3] está presente os casos correspondentes a PRBS B=0.02, PRBS B=0.05, PRBS B=0.005 e Square com $f = 0.1Hz$.

O critério de decisão entre cada modelo criado consiste na comparação da resposta do modelo real e do modelo proposto ao mesmo input - no caso em questão seleccionou-se um sinal de teste/validação do tipo PRBS com B=0.01 - seleccionou-se este sinal pois tem características rápidas, o que permite validar a dinâmica do modelo proposto, mas também permite às vezes observar o sistema a estabilizar num valor constante, mesmo que com oscilações.

Sinal	**na**	**nb**	**FIT %**
PRBS B=0.02	2	2	32.53
	3	2	23.21
	4	1	35.52
	5	5	98.17
	6	5	99.61
	7	6	99.68
	8	5	99.69

Tabela 1: Melhores resultados para PRBS B=0.02

Sinal	**na**	**nb**	**FIT %**
PRBS B=0.05	2	2	10.14
	3	2	14.76
	4	1	7.49
	5	5	99.64
	6	5	99.70
	7	6	99.70
	8	8	99.70

Tabela 2: Melhores resultados para PRBS B=0.05

Sinal	na	nb	FIT %
SQUARE f=0.08Hz	2	2	33.67
	3	4	25.83
	4	1	31.31
	5	5	99.68
	6	6	99.70
	7	4	99.70
	8	7	99.70

Tabela 3: Melhores resultados para SQUARE $f = 0.1$

Sinal	na	nb	FIT %
PRBS B=0.005	2	2	35.62
	3	3	30.67
	4	1	37.57
	5	5	91.55
	6	6	98.88
	7	7	99.61
	8	5	99.62

Tabela 4: Melhores resultados para PRBS B=0.005

Verifica-se pelos dados anteriormente apresentados que se no modelo ARMAX o parâmetro $\mathbf{n_a}$ tem o valor de 5 ou superior, o parâmetro **FIT** desloca-se logo para valores que rondam os 90% - assim sendo todos os resultados anteriores que correspondem a $\mathbf{n_a} \geq 5$ são plausíveis de serem o modelo que deveremos escolher. Por questões de controlabilidade dever-se-á optar pelo resultado menos complexo, ou seja o que apresenta menos pólos e zeros no sistema.

Assim sendo, optou-se por escolher o resultado proveniente da aplicação do sinal PRBS com B=0.02 e o ARMAX com os seguintes parâmetros: $\mathbf{n_a} = 5$, $\mathbf{n_b} = 5$, $\mathbf{n_c} = 5$, $\mathbf{n_k} = 1$.

Para confirmar as carateristicas básicas e essenciais que o modelo que se está a desenvolver tem que ter, apresenta-se na Figura [10], a resposta no modelo proposto a um sinal SQUARE, onde se pode verificar o fenómeno de chicoteada.

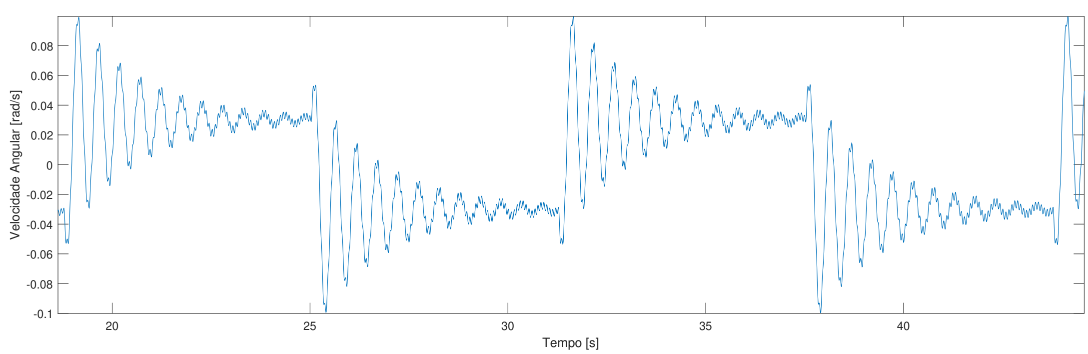

Figura 10: Reação do Modelo proposto a um input SQUARE onde se verifica a chicoteada

4.5 Conversão para Modelo em Espaço de Estados

O modelo em espaço de estados e a função de transferência correspondente está apresentada em baixo:

$$A = \begin{bmatrix} 5.120 & -11.426 & 14.347 & -10.699 & 4.459 & -0.801 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \qquad B = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$C = 10^{-3} \cdot \begin{bmatrix} -0.0648 & 0.4139 & -0.5277 & 0.5512 & -0.4267 & 0 \end{bmatrix} \qquad D = 0$$

$$H(z) = \frac{-6.483 \cdot 10^{-5} z^5 + 4.139 \cdot 10^{-4} z^4 - 5.277 \cdot 10^{-4} z^3 + 5.512 \cdot 10^{-4} z^2 - 4.267 \cdot 10^{-4} z}{z^6 - 5.12 z^5 + 11.43 z^4 - 14.35 z^3 + 10.7 z^2 - 4.459 z + 0.8014}$$

Apresenta-se na Figura [11] a localização no espaço complexo a localização dos zeros e dos pólos do modelo proposto que propõe simular o modelo real fornecido e na Tabela [5], está identificado de forma numérica os pólos e os zeros.

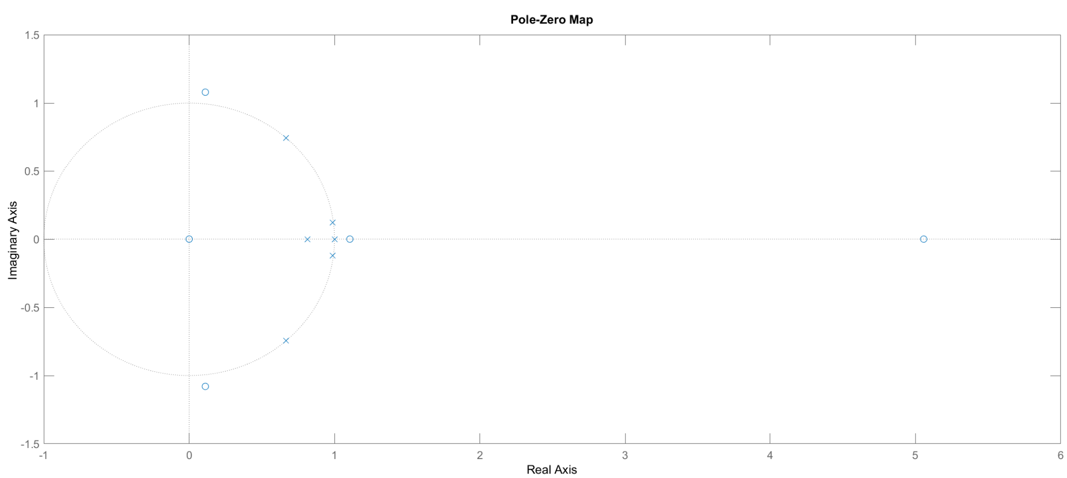

Figura 11: Representação no espaço complexo da localização dos pólos e zeros do modelo proposto.

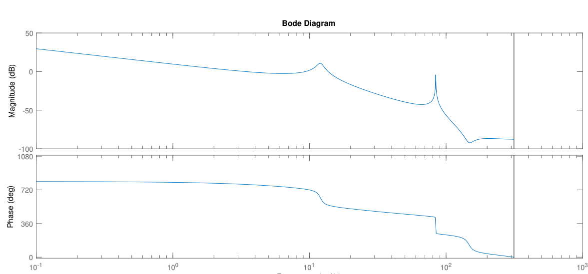

Figura 12: Resposta em frequência do modelo escolhido

Pólos	Zeros
$0.6669 + 0.7440i$	0
$0.6669 - 0.7440$	$5.0564 + 0.0000i$
$0.9869 + 0.1192i$	$1.1053 + 0.0000i$
$0.9869 - 0.1192i$	$0.1114 + 1.0795i$
$1.0000 + 0.0000i$	$0.1114 - 1.0795i$
$0.8122 + 0.0000i$	-

Tabela 5: Valor do pólos e zeros do sistema proposto

Para validar o modelo proposto, comparou-se a resposta que o modelo proposto e o modelo real têm a um mesmo sinal de input. Nas Figuras abaixo estão apresentadas várias respostas a diferentes sinais - uns muito rápidos onde o sistema não tem tempo de estabilizar num valor constante.

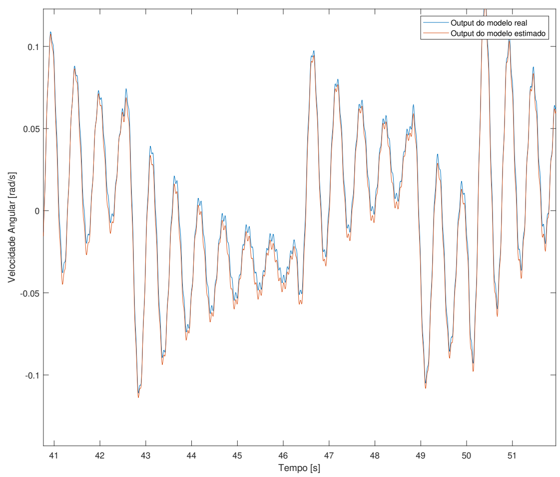

Figura 13: PRBS B=0.008

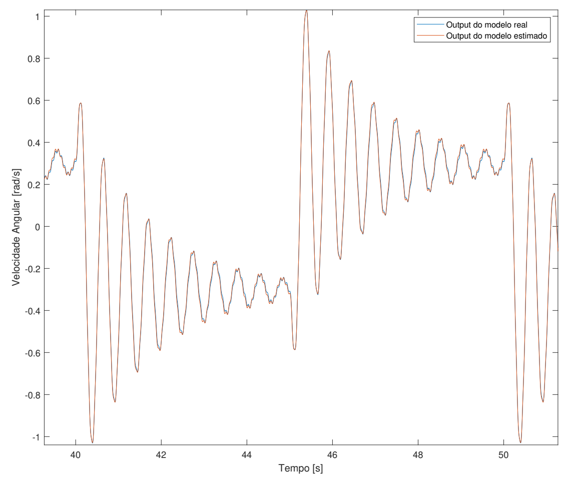

Figura 14: Square f=0.1Hz

Por observação das respostas no domínio do tempo, conclui-se que o modelo proposto tem uma resposta igual ao modelo real e assim sendo, é possível afirmar que o modelo proposto é um bom modelo para modelar a dinâmica do modelo real.

Referências

[1] Roberto Diversi, Roberto Guidorzi, and Umberto Soverini. Identification of armax models with noisy input. *International Federation of Autometic Control (IFAC)*, 2011.

5 Anexos

5.1 ARMAX [1]

A modelação do sistema "Motor+Barra" pode ser feito através do modelo ARMAX - *auto-regressive moving-average with exogenous inputs.* Este modelo descreve processos estocásticos estacionários na forma polinomial como soma de três termos: parcela autoregressão (AR), componente média móvel (MA) e um termo independente (exógeno). O modelo resulta da consideração de inputs anteriores, outputs e amostras de "white noise". A equação da abordagem ARMAX é dada por:

$$y(t) + \sum_{i=1}^{n_a} a_i y(t-i) = \sum_{j=0}^{n_b-1} b_j u(t-n_k-j) + e(t) + \sum_{k=1}^{n_c} c_k e(t-k) \tag{5}$$

E esquema equivalente ao modelo ARMAX é representado por:

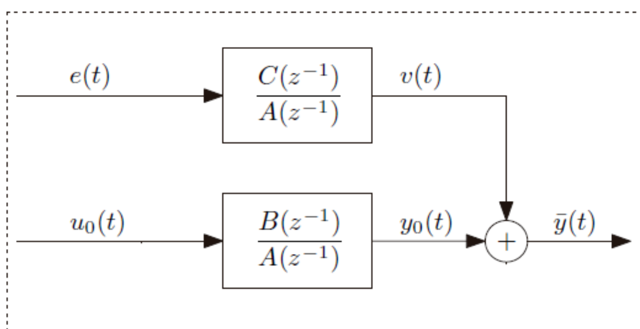

Figura 15: Esquema do Modelo ARMAX

Interpretando a imagem anterior reconhece-se uma componente determinística ligada ao input $u_0(t)$ que é caracterizada pela função transferência $\frac{B(z-1)}{A(z-1)}$ e uma parte estocástica comandada pelo "white noise", e(t).

O ramo estocástico caracterizado pela expressão $\frac{C(z-1)}{A(z-1)}$ converte o AWGN (additive white gaussian noise) num sinal v(t) de "colored noise" que representa possíveis efeitos de perturbação não considerados na componente determinística. Ou seja, o ARMAX modela o erro como uma combinação linear de ruído branco. A sua saída é $y^{-}(t) = y_0(t) + v(t)$.

O modelo ARMAX foi realizado com recurso ao MATLAB e à função de mesmo nome da Control Systems Toolbox. À função foi fornecido duas series temporais - par input/output - de um estimulo aplicado ao sistema a modelar e as previsões dos parâmetros:

- n_A - ordem do polinómio A(z),
- n_B - ordem do polinómio B(z)+1,
- n_C - ordem polinómio C(z) e modelo do erro,
- e n_K - input-output delay do sistema.

Uma vez que os parâmetros anteriores são desconhecidos foi necessário fazer uma estimativa realista recordando o significado destes na ordem de zeros e pólos e a condição-restrição para sistemas causais - ordem do numerador < ordem do denominador.

A condição anterior aplicada ao ARMAX toma a forma $n_k + n_B - 1 < n_A$. Como para o atraso puro se considerou o valor unitário a condição ficou $n_B < n_A$. No modelo do erro do sistema assumiu-se $n_C = n_A$.

A necessidade de um modelo robusto com resposta próxima do comportamento do sistema "real" não é a única consideração a fazer. A futura tarefa de design do controlador requer que nesta fase do projecto se faça uma modelação o mais simples possível sem comprometer a acurácia do modelo.

Com estas considerações em mente fez-se experimentação com diferentes possíveis combinações.

A resposta temporal do modelo tem de ser tal que este mantenha as qualidades características do sistema "real". No caso do "Motor+Barra", isto significa, não perder o efeito de chicotada e a principal dinâmica oscilatória.

5.2 Código no MATLAB

5.2.1 *BasicSim.m*

```
% Flexible robot link
%----------------------------------------------------------------------
% Loads the true flexible link model
load('barrassmodel.mat')
% Defines experiment parameters
fs=1000;
Ts=1/fs;     % Sampling interval
tfinal=100;

flag.input=1; % 1 - sinal quadrado
              % 0 - PRBS

% PRBS input
    PRBS.dfreq=10000;  % Create the discrete time vector for this signal
    PRBS.time=0:tfinal/PRBS.dfreq:tfinal;

    PRBS.ref=idinput(length(PRBS.time),'PRBS',[0 0.008]);
    PRBS.T=AveragePeriod(PRBS.time,PRBS.ref); % Average signal Period
    PRBS.input=[transpose(PRBS.time), PRBS.ref];
% Square input
    Square.dfreq=10000;  % create the discrete time vector for this signal
    Square.time=0:tfinal/Square.dfreq:tfinal;
    Square.freq=0.1;  %Hertz
    Square.ref=10*square(2*pi*Square.freq*Square.time);
    Square.input=[transpose(Square.time), transpose(Square.ref)];


% Simulates the true model to generate data for identification
```

```
% Input: Ts, State-Space(Atrue,Btrue,Ctrue,Dtrue),
% Output: t, ts, us, y, ys
sim('barra1');

% Filtering the output with differentiation
% The filter added has the property of being a low pass filter and
% atteniate the high frequency noise that may exist

    lambda.value=0.8;
    lambda.num=[1-lambda.value, -(1-lambda.value)];
    lambda.den=[1 ,-lambda.value];
    dy=filter(lambda.num,lambda.den,ys);

% Plot of the filter output signal with the labmda parameter specified
% earlier

    figure(2)
    gg=plot(ts,dy);
    set(gg,'LineWidth',1.5);
    gg=xlabel('Tempo [s]');
    set(gg,'FontSize',14);
    gg=ylabel('Velocidade Angular [rad/s]');
    set(gg,'Fontsize',14);
    %xlim([10 35]);
    hold on
    %plot(ts(1:length(ts)-1),diff(ys));
    plot(ts,filter([1,-1],[1,0],ys));
    hold off
    u = dtrend(us);
%% Plots the continuous and the discrete time outputs

    figure(2)
    gg=plot(t,y);                  % Plot the continous time output position y
    set(gg,'LineWidth',1.5);
    gg=xlabel('t (s)');
    set(gg,'FontSize',14);
    gg=ylabel('y (volt)');
    set(gg,'Fontsize',14);
    %xlim([40 50]);
    hold on
    gg=plot(ts,ys,'r');            % Plot the discrete time output position y
    set(gg,'LineWidth',1.5);
    hold off

% Saves data for identification
% us and ys contain i/o data with a sampling interval of Ts
    save('iodata1.mat','us','ys','ts')

%% Identification algorithm


    z = [dy u];          % Z has the input and output data
```

```
    na = 5;             % AR part - order of the poles
    nb = 5;             % X part - order of zeros + 1
    nc = na;            % MA part - order of C
    nk = 1;             % Atraso puro âĂŞ pure delay
    nn = [na nb nc nk];
    th = armax(z,nn);     % th is a structure in identification toolbox
                        % format

    [den1,num1] = polydata(th);
    dysim = filter(num1,den1,u); % gives the dy time domain using the
                                 % estimated plant
    [num,den] = eqtflength(num1,conv(den1,[1 -1])); % plant from u to y
    ysim = filter(num,den,u);    % estimated response of y with the
                                 % identified plant parameters
    [A,B,C,D] = tf2ss(num,den);  % State Space configuration



    %compare(z,th)               % Use this function to get a fit value
                                 % for a test set


%% Write into .txt file

    fileID=fopen('data_identification.txt','a');
    fprintf(fileID,'--------------------------------------------- \n');
    fprintf(fileID,'Ts = %f \n',Ts);
    fprintf(fileID,'Square wave input with 1 Mhz\n');
    fprintf(fileID,' na = %d \n nb = %d \n nc = %d \n nk = %d \n',na,nb,nc,nk);
    fprintf(fileID,' den=');
    fprintf(fileID,' %f ',den);
    fprintf(fileID,'\n');
    fprintf(fileID,' num=');
    fprintf(fileID,' %f ',num);
    fprintf(fileID,'\n');
    %writematrix(round(A,2),'data_identification.txt','WriteMode','append');
    fclose(fileID);




%----------------------------------------------------------------------
% End of file
```

5.2.2 IdentificationAlgorithm_best.m

```
% ----------------------------------------------------------------------------
% File: IdentificationAlgorithm_best.m
% Description: Determine a set of best model estimates considering a
%              certain test set choosen
% ----------------------------------------------------------------------------

load('barrassmodel.mat')

% Set sampling frequency
    fs=100;
    Ts=1/fs;
    tfinal=100;

    lambda.value=0.8;
    lambda.num=[1-lambda.value, -(1-lambda.value)];
    lambda.den=[1 ,-lambda.value];



% Create test set for fitting
    flag.input=0;
    % PRBS test set configuration
    PRBS.dfreq=10000;  % Create the discrete time vector for this signal
    PRBS.time=0:tfinal/PRBS.dfreq:tfinal;
    PRBS.ref=idinput(length(PRBS.time),'PRBS',[0 0.01]);
    PRBS.T=AveragePeriod(PRBS.time,PRBS.ref); % Average signal Period
    PRBS.Avfreq=1/PRBS.T;
    PRBS.input=[transpose(PRBS.time), PRBS.ref];
    % Square Signal test set configuration
    Square.dfreq=10000;  % create the discrete time vector for this signal
    Square.time=0:tfinal/Square.dfreq:tfinal;
    Square.freq=0.04;  %Hertz
    Square.ref=square(2*pi*Square.freq*Square.time);
    Square.input=[transpose(Square.time), transpose(Square.ref)];
    sim('barra1');  % output: aux, us, y, ys, ts, t
    % Create struct with test set configuartions
    testset.u=us;
    testset.t=ts;
    testset.ys=ys;
    testset.dy=filter(lambda.num,lambda.den,ys);
    fprintf('test set PRBS with average Frequency: %f [Hz]\n', PRBS.Avfreq);
    clear PRBS Square aux us y ys ts t


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

% Flag for choosing type of input signal
    flag.input=1;   % 0 - PRBS
                    % 1 - sinal quadrado
```

```

% PRBS input
     PRBS.dfreq=10000;  % Create the discrete time vector for this signal
     PRBS.time=0:tfinal/PRBS.dfreq:tfinal;
     PRBS.ref=idinput(length(PRBS.time),'PRBS',[0 0.02]);
     PRBS.T=AveragePeriod(PRBS.time,PRBS.ref); % Average signal Period
     PRBS.Avfreq=1/PRBS.T;
     PRBS.input=[transpose(PRBS.time), PRBS.ref];
% Square input
     Square.dfreq=10000;  % create the discrete time vector for this signal
     Square.time=0:tfinal/Square.dfreq:tfinal;
     Square.freq=0.08;  %Hertz
     Square.ref=square(2*pi*Square.freq*Square.time);
     Square.input=[transpose(Square.time), transpose(Square.ref)];

sim('barra1');  % output: aux, us, y, ys, ts, t
dy=filter(lambda.num,lambda.den,ys);
u=dtrend(us);
z = [dy u];          % Z has the real input and output data
Data(1,:)=["Type of Signal", "Freq [Hz]" ,"na", "nb", "nc", "nk", "FIT"];
for na=1:8
    for nb=1:na
    nc = na;             % MA part - order of C
    nk = 1;              % Atraso puro âĂŞ pure delay
    nn = [na nb nc nk];
    th = armax(z,nn);     % th is a structure in identification toolbox
                          % format

    [den1,num1] = polydata(th);
    %dysim = filter(num1,den1,u); % gives the dy time domain using the
                                  % estimated plant
    [num,den] = eqtflength(num1,conv(den1,[1 -1])); % plant from u to y
    %ysim = filter(num,den,u);     % estimated response of y with the
                                  % identified plant parameters
    %[A,B,C,D] = tf2ss(num,den);  % State Space configuration

    [¬,FIT]=compare([testset.dy testset.u],th);      % get a test set for ...
       this, maybe a PRBS

    if (exist('best'))==0
        best.FIT=FIT;   best.na=na;     best.nb=nb;     best.nc=nc;
        best.nk=nk;     best.num=num;   best.den=den;
    else
       if best.FIT<FIT
            best.FIT=FIT;   best.na=na;     best.nb=nb;     best.nc=nc;
            best.nk=nk;     best.num=num;   best.den=den;
       end
    end

    % Save data from each simulation
        fileID=fopen('data_identification.txt','a');
        fprintf(fileID,'---------------------------------------------- \n');
```

```
        fprintf(fileID,'Ts = %f \n',Ts);
        %fprintf(fileID,'Square wave input with %f Hz\n', square.f);
        fprintf(fileID,' na = %d \n nb = %d \n nc = %d \n nk = %d \n'...
            ,na,nb,nc,nk);
        fprintf(fileID,' den=');    fprintf(fileID,' %f ',den);
        fprintf(fileID,'\n');       fprintf(fileID,' num=');
        fprintf(fileID,' %f ',num); fprintf(fileID,'\n');
        fprintf(fileID,'FIT=%f \n',FIT);
        %writematrix(round(A,2),'data_identification.txt','WriteMode','append');
        fclose(fileID);
    end
    % Write the best estimation for each na choosen
    % The criteria is the FIT value resulted from the test set
        fileID=fopen('best_cases.txt','a');
        fprintf(fileID,'--------------------- \n');
        fprintf(fileID,' na = %d \n nb = %d \n',best.na,best.nb);
        fprintf(fileID,' den=');
        fprintf(fileID,' %f ',best.den);
        fprintf(fileID,'\n');
        fprintf(fileID,' num=');
        fprintf(fileID,' %f ',best.num);
        fprintf(fileID,'\n');
        fprintf(fileID,' FIT=%f \n',best.FIT);
        fclose(fileID);

        Data(na+1,:)=["PRBS", PRBS.Avfreq ,best.na, best.nb, best.nc, ...
            best.nk, best.FIT];


    clear best
end
```