

Identificação de Sistemas Dinâmicos

Experimento 1 - Identificação de Processo Não-Linear de 4a Ordem

Rafael Ramos de Matos

Matrícula - 150145683

UnB - Universidade de Brasília

ramsunb@gmail.com

Abstract

This paper presents the results of identifying a fourth order liquid level system. The identification algorithms used in this for this article are: Estimate transfer function, ARX, ARMAX, Estimate Output-Error (EO), Estimate Box-Jenkins (BJ) and Estimate state-space.

1. Objetivos

O objetivo do experimento 1 é, usando a ferramenta MATLAB, fazer a identificação, usando simulações sobre um modelo não linear, de um sistema sistema de níveis de líquido de quarta ordem em torno de um ponto de operação que diminua o erro de identificação. Além disso, esse experimento visa aprimorar o conhecimento do aluno sobre quais variáveis devem ser consideradas durante o processo de identificação.

As figuras 1 e 2 mostram, respectivamente, o sistema de líquido real, presente no LARA, e um esquemático 3D do sistema.



Figura 1. Foto do sistema de níveis de líquidos (LARA)

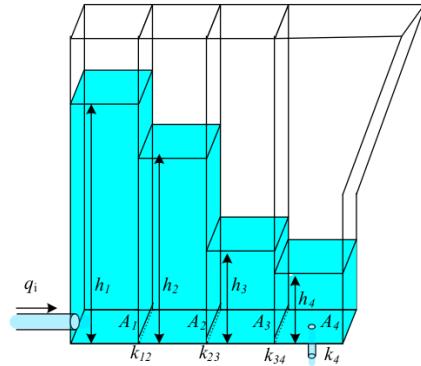


Figura 2. Esquemático do sistema de níveis de líquidos

2. Introdução

2.1. Contextualização

Na industria, o trabalho com líquidos é recorrente e, as vezes, bastante perigoso sua manipulação por trabalho humano. Armazenar, transferir e misturar esses líquidos em quantidades predefinidas com alta acurácia também não é tarefa fácil. Neste contexto, a identificação de sistema dinâmicos de níveis de líquido, para a realização de controle, se faz necessário para aumentar tanto a produção da industria quando a segurança durante as operações.

2.2. identificação de sistemas

Identificação de sistemas é um processo que faz uso de ferramentas matemáticas e computacionais, com intuito de realizar a construção de modelos a partir de dados de entrada e saída, que por sua vez, venham descrever o comportamento de um determinado sistema durante seu funcionamento.

Como foi mencionado no paragrafo a cima a identificação de sistema faz uso de de ferramentas matemática para a construção de modelos. De acordo com o modelo que será identificado, existe um algoritmo de

identificação que consegue modelar com maior precisão o sistema em questão.

Os algoritmos de identificação de sistemas que serão usados neste experimento serão discutidos nos tópicos posteriores.

3. Materiais

- Software MATLAB

4. Métodos

4.1. Modelo Função de transferência

Função de transferência descreve o comportamento dinâmico de um sistema observando as entradas e saída do sistema, ou seja, mostra como o efeito de uma determinada entrada é transferido para a saída.

Usando a função do matlab `tfest(data,np,nz)`, onde *data* é uma estrutura de dados que contem os dados de entrada e saída do experimento e, *np* e *nz*, são os números de polos e números de zeros, respectivamente.

O retorno desta função é uma função de transferência que mais se aproxima do modelo real.

4.2. Modelo ARX

ARX (*autoregressive with exogenous inputs*) é um modelo que pode ser levantado a partir do modelo geral:

$$A(q)y(k) = \frac{B(q)}{F(q)}u(k) + \frac{C(q)}{D(q)}\nu(k) \quad (1)$$

$$y(k) = \frac{B(q)}{A(q)F(q)}u(k) + \frac{C(q)}{A(q)D(q)}\nu(k)$$

$$y(k) = G(q)u(k) + C(q)H(q)\nu(k) \quad (2)$$

sendo q^{-1} o operador de atraso, de forma que $y(k)q^{-1} = y(k-1)$, $\nu(k)$ seja o ruido branco do sistema e os polinômios $A(q), B(q), C(q), D(q)$ e $F(q)$ os polinômios mostrados a seguir:

$$A(q) = 1 + a_1q^{-1} + \dots + a_{n_y}q^{-n_y};$$

$$B(q) = b_1q^{-1} + \dots + b_{n_u}q^{-n_u};$$

$$C(q) = 1 + c_1q^{-1} + \dots + c_{n_\xi}q^{-n_\xi};$$

$$D(q) = 1 + d_1q^{-1} + \dots + d_{n_d}q^{-n_d};$$

$$F(q) = 1 + f_1q^{-1} + \dots + f_{n_f}q^{-n_f}.$$

A figura 3, mostrada logo a seguir, mostrar a implementação do modelo geral por braços.

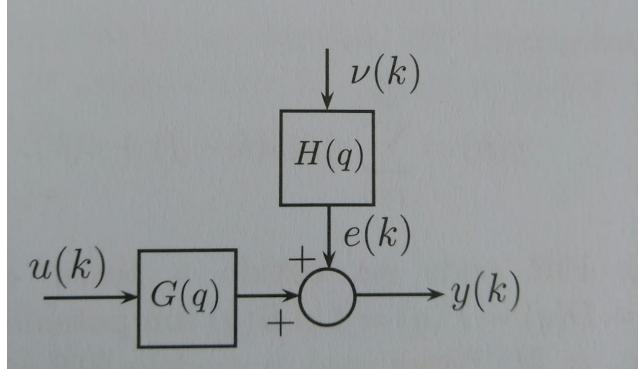


Figura 3. modelo geral

A partir do modelo geral, podemos implementar o modelo ARX fazendo os polinômios $C(q) = D(q) = F(q) = 1$ e $A(q)$ e $B(q)$ polinômios quaisquer, fazendo a equação 1 resultar em:

$$A(q)y(k) = B(q)u(k) + \nu(k) \quad (3)$$

Isolando a saída $y(k)$ da equação 3, temos o modelo ARX como modelo de erro na equação.

$$y(k) = \frac{B(q)}{A(q)}u(k) + \frac{1}{A(q)}\nu(k) \quad (4)$$

A figura 4 mostrada logo abaixo mostra a representação esquemática do modelo ARX.

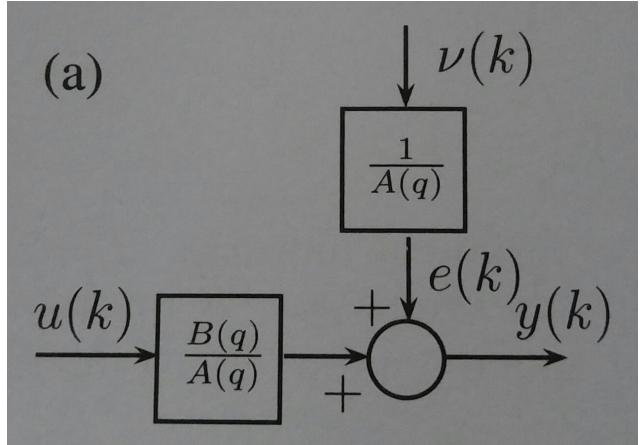


Figura 4. Modelo ARX

Com o auxilio da ferramenta MATLAB, usando o comando `arx(data, orders)`, onde *data* é uma estrutura de dados que contem os dados de entrada e saída do experimento e *orders* é a ordem do sistema a ser identificado.

A função `arx()` retorna os polinômios $A(q)$ e $B(q)$ com as ordens especificadas, previamente, na função.

4.3. Modelo ARMAX

O modelo ARMAX (*autoregressive moving average with exogenous inputs*) é construído tomando como base o modelo geral descrito na equação 1. Para isso, fazemos $D(q) = F(q) = 1$ e $A(q), B(q)$ e $C(q)$ polinômios quaisquer, resultando na seguinte equação:

$$A(q)y(k) = B(q)u(k) + C(q)\nu(k) \quad (5)$$

Outra forma de escrevermos a equação 5 pode ser vista logo abaixo.

$$y(k) = \frac{B(q)}{A(q)}u(k) + \frac{C(q)}{A(q)}\nu(k) \quad (6)$$

A figura 5, mostrada logo abaixo, apresenta o esquemático do modelo ARX.

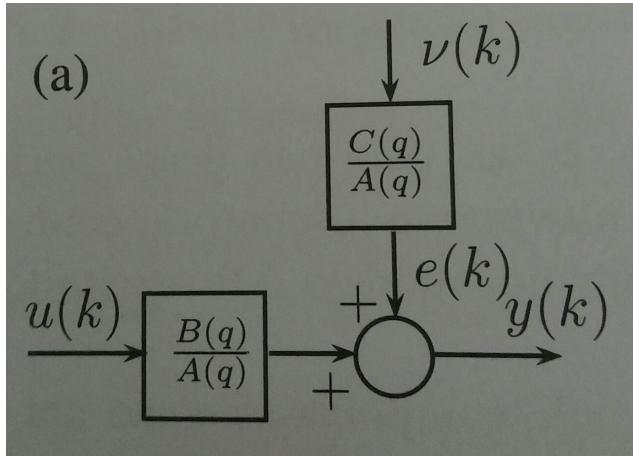


Figura 5. Modelo ARMAX

Usando a função do matlab *armax(data, ordens)*, onde *data* é uma estrutura de dados que contém os dados de entrada e saída do experimento e *ordens* é a ordem dos polinômios $A(q), B(q)$ e $C(q)$.

O retorno dessa função são os polinômios $A(q), B(q)$ e $C(q)$.

4.4. Modelo OE

O modelo de erro na saída também pode ser derivado da equação 2 fazendo o polinômio $A(q) = 1$. Seguindo com formulação, temos $A(q) = C(q) = D(q) = 1$, $B(q)$ e $F(q)$ polinômios quaisquer, temos que:

$$y(k) = \frac{B(q)}{F(q)}u(k) + \nu(k) \quad (7)$$

fazendo $\frac{B(q)}{F(q)}u(k) = W(k)$ temos que a nova formulação fica como pode ser visto na equação 8.

$$y(k) = W(k) + \nu(k) \quad (8)$$

Como pode ser visto, como esse modelo, o ruído branco será levado diretamente para a saída do sistema, sem nenhuma intervenção dinâmica. Essa afirmação pode ser conferida na representação esquemática do modelo de erro na saída, que é mostrado na figura 6.

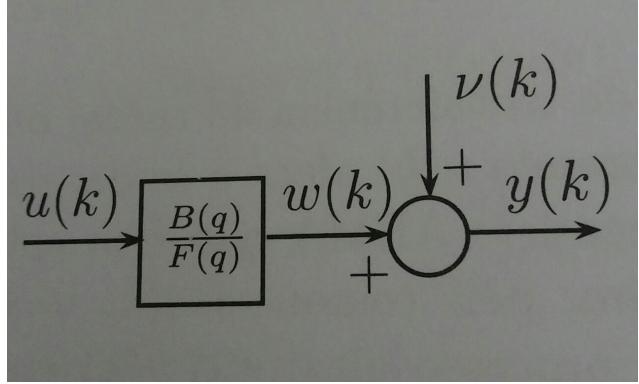


Figura 6. modelo de erro na saída (eo)

Usando a função do matlab *oe(data, ordens)*, onde *data* é uma estrutura de dados que contém os dados de entrada e saída do experimento e *ordens* é a ordem dos polinômios $B(q)$ e $F(q)$.

O retorno dessa função são os polinômios $B(q)$ e $F(q)$.

4.5. Modelo Box-Jenkins (BJ)

O modelo de Box-Jenkins pode ser obtido a partir de considerações sobre o modelo geral representado na equação 1. A consideração que será acatada para a construção do modelo de Box-Jenkins é, usando a equação 1, tomando-se $A(q) = 1$ e considerando os polinômios restantes arbitrários.

$$y(k) = \frac{B(q)}{F(q)}u(k) + \frac{C(q)}{D(q)}\nu(k) \quad (9)$$

A equação 9 mostra que, o modelo de Box-Jenkins (BJ), as funções de transferência do sistema, $\frac{B(q)}{F(q)}$ e $\frac{C(q)}{D(q)}$, não têm parâmetros em comum, ou seja, a parametrização por esse modelo trada o ruído branco de forma independente.

A figura 7 mostrada logo abaixo mostra a representação esquemática do modelo Box-Jenkins.

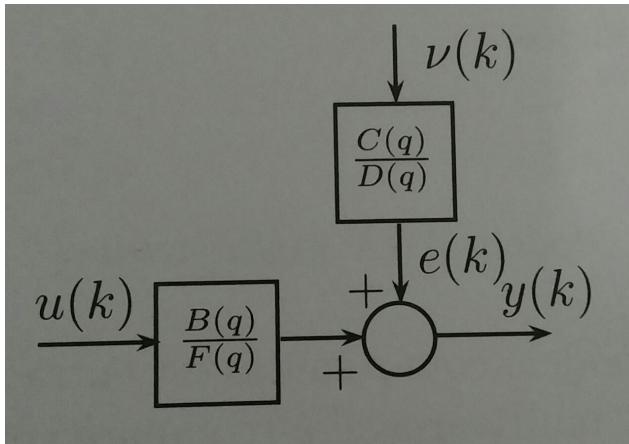


Figura 7. Modelo Box-Jenkins

Usando a função do matlab `bj(data, ordens)`, onde *data* é uma estrutura de dados que contém os dados de entrada e saída do experimento e *ordens* é a ordem dos polinômios $B(q)$, $F(q)$, $C(q)$ e $D(q)$. o retorno dessa função são os polinômios $B(q)$, $F(q)$, $C(q)$ e $D(q)$.

O retorno dessa função são as matrizes A , B , C , D e a matriz de atraso, K .

4.6. Modelo Espaço-de-Estados (Forma Canônica Controlável)

O modelo de espaço de estados é uma representação que pode ser usada para modelar relações de entrada e saída de sistemas e, também considera, na modelagem o estado de suas variáveis interna do sistema. Esse tipo de representação modela sistemas no domínio do tempo e é mais adequado para representar sistemas não lineares e que possuem muitas variáveis. Os modelos característico em espaço de estados possui o seguinte formato:

$$\begin{aligned}\dot{\mathbf{x}} &= A\mathbf{x} + B\mathbf{u} \\ \mathbf{y} &= C\mathbf{x} + D\mathbf{u}\end{aligned}\quad (10)$$

Na equação 10, $\mathbf{x} \in \mathbb{R}^n$ é um vetor de estado de n -dimensões, o ponto indica indica a derivada temporal; $\mathbf{u}(t) \in \mathbb{R}^r$ é o vetor m -dimensional de saídas medidas e A , B , C e D são matrizes contantes.

Usando a função do matlab `ssest(data, nx)`, onde *data* é uma estrutura de dados que contém os dados de entrada e saída do experimento e *nx* é a ordem do sistema.

O retorno dessa função são as matrizes A , B , C , D e a matriz de atraso, K .

5. Procedimentos

Os procedimentos para a realização deste experimento, foram, basicamente, entendimento dos algoritmos fornecidos pelo professor, implementação do algoritmo que gerasse tanto o de sinal de entrada do sistema quanto fizesse

a identificação dos modelos. Além disso, a amplitude do sinal de entrada, frequência do sinal de entrada, ponto de operação do sistema e taxa de amostragem eram variáveis que teriam que ser calibrada para que o experimento corresse dentro dos parâmetros especificados pelo professor. Esses três procedimentos serão melhor discutido nos seus respectivos tópicos.

5.1. Entendimento Dos Algoritmos Fornecido Pelo Professor

A primeira tarefa feita neste experimento foi ler o roteiro para compreender qual seria objetivo do experimento e quais suas etapas durante o processo de realização. Para isso, foi feita a leitura do roteiro em conjunto com a análise do código `Lin_Analitica.m` e do bloco simulink `lia4MA.slx`, para entender quais eram as entradas e saídas do sistema, de forma que o experimento fosse feito corretamente.

5.2. Escolhas da Variáveis: Ponto de Operação, Amplitude e taxa de Amostragem

Como a identificação do sistema é para pequenos sinais, foi escolhido a amplitude de 1, pois com essa amplitude, a entrada provocava uma variação significativa na saída, de tal modo, que permitia sua identificação.

O dois ponto de operação foram escolhidos arbitrariamente como os valores de 10 e 12.

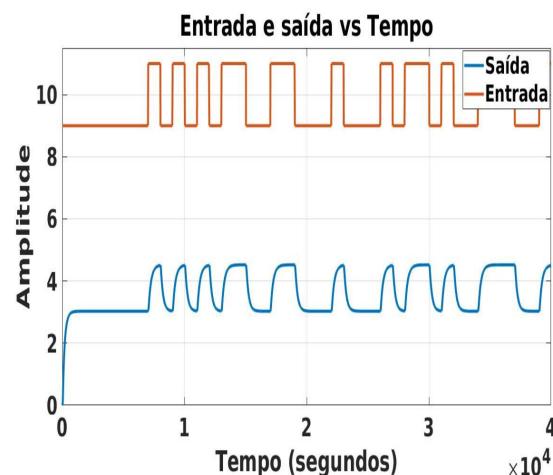


Figura 8. Sinais de entrada e saída com amplitude 10

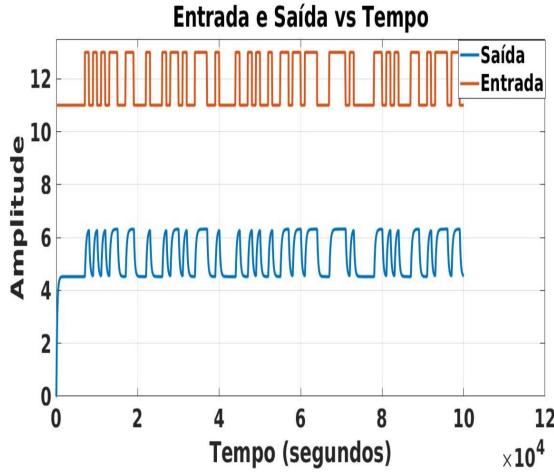


Figura 9. Sinais de entrada e saída com amplitude 12

A taxa de amostragem foi escolhida de forma que fornecesse o melhor FIT possível. Uma discussão mais aprofundada sobre a taxa de amostragem e nível de ruído se dará nas análises dos resultados.

5.3. Implementação Do Algoritmo: Entrada De Dados E Identificação

Com os valores para as variáveis de amplitude, ponto de operação e taxa de amostragem do sinal de entrada já foram pré-definidos na subseção anterior, partimos para próxima etapa. Para os sinais de entrada, foi gerado um vetor de tempo e um sinal PRBS (*Pseudorandom binary signal*) com a função *idinput()*.

Na função *idinput()* implementada, foram feitos alguns experimentos variando-se o argumento de banda do sinal até que o sinal de saída do sistema atingisse o regime permanente.

Com o sinal de entrada calibrado, partiu-se para a identificação do sistema usando os modelos e as funções do MATLAB citados na seção 4. O algoritmo de identificação foi rodado diversas vezes, aumentando-se o tamanho da amostra até que essa variável não influenciasse no FIT dos modelos.

6. Resultados

6.1. Identificação sem ruído: Ponto de operação 10

A figura 10, gerada usando a função *compare()* do MATLAB, mostra os FIT em porcentagem de todos os modelos abordado nesse experimento.

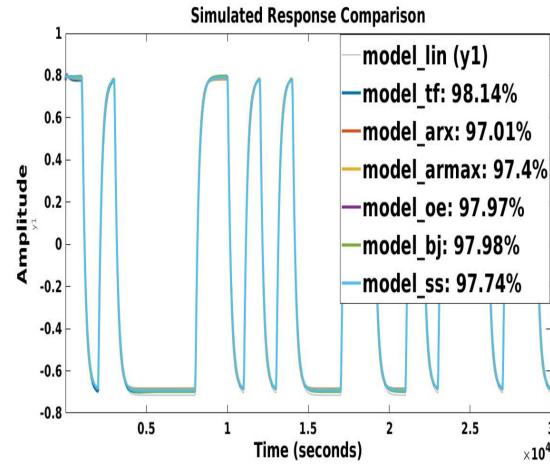


Figura 10. Comparação dos modelos com ponto de operação 10 em relação ao Ground Truth

Funções de transferência dos modelos:

Modelo continuo gerado pelo estimador Função de transferência.

$$TF(S) = \frac{0.0054562(S + 0.0003893)(S^2 - 0.1525S + 0.08751)}{(s + 2.563)(S + 0.04847)(S + 0.005151(S + 0.0003908))} \quad (11)$$

Modelo continuo gerado pelo estimador ARX.

$$TF(s) = \frac{5.6317e - 05(S^2 + 2.63S + 2.095)(S^2 + 1.32S + 7.177)}{(s + 0.6418)(S + 0.03173)(S + 0.005723)(S^2 + 0.1924S + 9.879)} \quad (12)$$

Modelo continuo gerado pelo estimador ARXMAX.

$$TF(s) = \frac{4.7558e - 05(S^2 - 2.37S + 2.346)(S^2 - 0.6876S + 6.466)}{(s + 0.4355)(S + 0.04132)(S + 0.005495)(S^2 + 0.2381S + 9.884)} \quad (13)$$

Modelo continuo gerado pelo estimador Output-Error.

$$TF(S) = \frac{0.0013656(S + 0.007954)(S^2 - 0.0468S + 0.02239)}{(S + 0.007393)(S + 0.005291)(S^2 + 0.1383S + 0.008349)} \quad (14)$$

Modelo continuo gerado pelo estimador Box-Jenkins.

$$TF(s) = \frac{2.7984e - 06(S + 1.521)(S^2 + 2.428S + 3.021)}{(s + 0.2264)(S + 0.00496)(S^2 + 0.2337S + 0.01531)} \quad (15)$$

Modelo continuo gerado pelo estimador Espaço-de-Estados (Forma Canônica Controlável).

$$TF(s) = \frac{0.00065596(S + 0.4698)(S^2 - 1.957S + 1.272)}{(S + 15)(S + 0.005179)(S^2 + 0.1456S + 0.006805)} \quad (16)$$

6.2. Identificação sem ruído: Ponto de operação 12

A figura 11, gerada usando a função *compare()* do MATLAB, mostra os FIT em porcentagem de todos os modelos abordado nesse experimento.

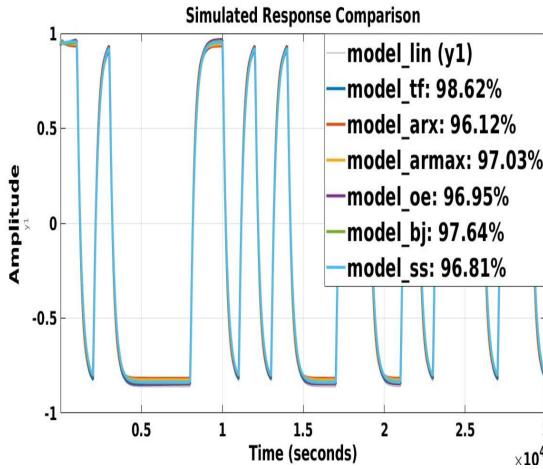


Figura 11. Comparação dos modelos com ponto de operação 12 em relação ao Ground Truth

Funções de transferência dos modelos:

Modelo continuo gerado pelo estimador Função de transferência.

$$TF(S) = \frac{-0.0081228(S - 0.4101)(S + 0.2717)(S + 1.729e - 06)}{(S + 6.386)(S + 0.03707)(S + 0.004286)(S + 2.757e - 06)} \quad (17)$$

Modelo continuo gerado pelo estimador ARX.

$$TF(S) = \frac{4.861e - 05(S^2 + 2.662S + 2.133)(S^2 + 1.344S + 7.202)}{(S + 0.6972)(S + 0.02507)(S + 0.004939)(S^2 + 0.2358S + 9.883)} \quad (18)$$

Modelo continuo gerado pelo estimador ARXMAX.

$$TF(s) = \frac{-8.3472e - 06(S - 1.614)(S + 1.481)(S^2 - 1.966S + 10.02)}{(S + 0.0044)(S^2 + 0.135S + 0.00515)(S^2 + 0.4172S + 9.92)} \quad (19)$$

Modelo continuo gerado pelo estimador Output-Error.

$$TF(s) = \frac{0.00887(S + 0.00416)(S^2 + 0.03494S + 0.0052)}{(s + 0.1642)(s + 0.0748)(s + 0.004495)(s + 0.003804)} \quad (20)$$

Modelo continuo gerado pelo estimador Box-Jenkins.

$$TF(s) = \frac{1.3e - 05(S^2 + 2.312S + 1.724)(S^2 + 1.064S + 7.115)}{(S + 0.0041)(S^2 + 0.096S + 0.0044)(S^2 + 0.035S + 9.87)} \quad (21)$$

Modelo continuo gerado pelo estimador Espaço-de-Estados (Forma Canônica Controlável).

$$TF(s) = \frac{0.00024(S + 0.8937)(S^2 - 0.9986S + 0.9271)}{(S + 11.11)(S + 0.003989)(S^2 + 0.1149S + 0.005003)} \quad (22)$$

6.3. Identificação com ruído crescente: Ponto de operação 10

Nível de Ríodo	FT	ARX	ARMAX	OE	BJ	SS
0	98.1	97.1	97.3	97.9	97.9	97.7
0.2	97.8	93.2	93.2	97.8	97.6	75.8
0.4	97.8	94.3	93.3	97.9	97.7	89.1
0.6	98.3	95.7	93.0	97.9	97.7	83.0
0.8	97.6	95.8	93.2	96.1	97.7	92.9
1.0	97.2	95.8	92.6	97.9	97.8	94.7
1.2	97.9	95.3	92.7	97.9	97.8	82.1
1.4	97.9	94.5	92.9	97.2	97.8	82.1
1.6	96.9	93.8	91.4	97.1	97.8	92.3
1.8	0.7	92.7	91.7	97.2	97.8	34.3
2.0	97.8	91.46	92.3	97.2	97.8	91.8

Tabela 1. FITs, em porcentagem, dos modelos em função de ruído crescente no ponto de operação igual a 10

6.4. Identificação com ruído crescente: Ponto de operação 12

Nível de Ríodo	FT	ARX	ARMAX	OE	BJ	SS
0	98.6	96.1	97.0	96.9	97.6	96.8
0.2	97.9	92.2	92.3	96.4	96.6	95.3
0.4	97.4	93.2	92.6	96.4	96.7	95.6
0.6	97.4	93.94	92.6	97.3	96.8	94.2
0.8	98.5	94.6	92.6	96.5	96.8	96.3
1.0	97.4	94.8	92.7	96.4	96.9	95.2
1.2	1.7	95.3	91.9	96.3	97.0	92.9
1.4	0.7	92.7	91.7	97.2	97.8	34.3
1.6	98.2	95.0	91.5	97.5	97.1	89.2
1.8	97.5	94.1	92.4	97.5	97.1	76.0
2.0	21.3	93.3	92.7	61.7	97.1	49.1

Tabela 2. FITs, em porcentagem, dos modelos em função de ruído crescente no ponto de operação igual a 12

7. Conclusões

Os resultados desse experimento, no tocante os dois pontos de operação, indica que, todos o métodos de identificação têm melhores resultado, exceto o método arx, no ponto de operação igual a 10. Essa primeira comparação se baseia no experimento com ruido zero.

Quando adicionamos ruído, os resultados se mostram diferentes. As variações na taxa de acerto, que podem ser vistas nas tabelas 1 e 2, mudam a medida que o nível de ruído varia de zero até 2. Esses resultados são bastante interessante, pois a taxa de acerto, em função do nível de ruído dos modelos, não é inversamente proporcional, como era de se imaginar.

Contudo, podemos concluir que existem estimadores de modelos mais sensíveis a ruído, no sentido de resultar numa boa identificação. Se considerarmos nosso nível de ruído máximo como 1, podemos perceber que o melhor estimador de modelo, dos abordados na seção 4, é o estimador por função de transferência 4.1. O estimador de modelo *output error* 4.4 também tem bons resultados. No quesito variação do ponto de operação, ele tem um resultado bem estável na sua taxa de acerto com o *Ground Truth*.

Podemos afirmar também, que o estimador de modelo que mais é prejudicado com o aumento do ruído é estimador por espaço de estados. Dentre todos os modelos, ele foi o que apresentou o pior resultado na região de contorno de nível de ruído de 0 a 1. Logo, para um predição *inf*, com ou sem ruído, o melhor modelo, sem dúvidas, é o de função de transferência.

8. Questão

O que podemos fazer, é colocar um compensador proporcional K grande, de forma que diminua a constante de tempo do sistema.

Referências

- [1] Notas de aulas: *Identificação Sistemas Dinâmicos - 167851, Turma A.*
- [2] Luis Antonio Aguirret (2015) *Introdução à identificação de sistemas*. Editora da UFMG, quarta edição.
- [3] <https://www.mathworks.com/help/>