Journal homepage http://revistas.unitru.edu.pe/index.php/SSMM

SELECCIONES MATEMÁTICAS
Universidad Nacional de Trujillo
ISSN: 2411-1783 (Online)
Vol. 05(01): 74 - 84 (2018)

Implementação de Identificação de Sistemas por Subespaços Usando C/C++

Implementation of Systems Identification by Subspace Using C/C++

Santos Miranda Borjas * and Guilherme Pillon de C. A. Pessoa †

Received, Dec. 31, 2017 *Accepted, Jun. 11, 2018*

DOI: http://dx.doi.org/10.17268/sel.mat.2018.01.09

Resumo

Hoje em dia o trabalho do engenheiro consiste cada vez mais em obter o modelo matemático do processo estudado. A maior parte da literatura referente à identificação de sistemas trata da obtenção de modelos polinomiais como o método de predição de erro (PEM) e método de variáveis instrumentais (IVM). No caso de sistemas complexos, o modelo em espaço de estados aparece como uma alternativa para os modelos PEM e IVM. Sistemas de grandes dimensões são comumente encontrados na indústria, e a aplicação dos algoritmos de identificação por subespaços neste campo é muito promissora. Na atualidade os modelos de identificação por subespaço, Multivariable Output Error State sPace (MOESP) e Numerical algorithms for Subspace State Space System IDentification (N4SID), são temas de estudo. O objetivo deste trabalho é implementar em C/C++ e no Matlab o algoritmo N4SID para identificar sistemas discretos invariantes no tempo operando em malha aberta, comparando estes, em relação ao desempenho e tempo de processamento.

Palavras chave. Identificação por Subespaços, Sistemas Deterministicos, C/C++.

Abstract

Nowadays, an engineer's work consists more and more of obtaining mathematical models of the studied processes. Great part of the literature referring to system identification deals with how to find polynomial models as Prediction Error Methods (PEM) and Instrumental Variable Methods (IVM). In case of complex systems, the state space model appears as an alternative to PEM and IVM models. For multivariable systems, these methods provide reliable state space models directly from input and output data. As systems of large dimensions are usually found in industry, the application of subspace identification algorithms in this field is very promising. Currently the subspace identification models Multivariable Output Error State sPace (MOESP) and Numerical algorithms for Subspace State Space System IDentification (N4SID), are topic of study. The objective of this work is to implement the N4SID algorithm in C/C ++ and in Matlab to identify discrete systems invariant in time operating in open loop, comparing these in relation to performance and processing time.

Keywords. Subspace Identification, Deterministic Systems, C/C++

1. Introdução.

A teoria da realização de sistemas para o caso determinístico pretende, em primeira instância,

* Universidade Federal do Rio Grande do Norte, Av. Senador Salgado Filho, Rio Grande do Norte- Brasil *santos.miranda@ccet.ufrn.br*.

† Universidade Federal do Rio Grande do Norte, Av. Senador Salgado Filho, Rio Grande do Norte- Brasil. *guilhermepillon@hotmail.com*

resolver o seguinte problema: dada a função de transferência, como encontrar um ou mais modelos no espaço de estados, que representem tal sistema. Existem técnicas que permitem solucionar tal problema, sobretudo para o caso SISO (Single Input,Single Output). Como a generalização para o sistema MIMO (Multiple Input Multiple Output) não é trivial, uma alternativa é trabalhar com as matrizes de resposta ao impulso. Existem muitas técnicas para solucionar este problema, porém a teoria da realização, utilizando a decomposição de valores singulares (SVD), tornou-se predominante por ser computacionalmente robusta [6]. Esta técnica é usada nos métodos de identificação por subespaços. Nos métodos de identificação de sistemas dinâmicos por subespaços são tratados modelos de sistemas lineares invariantes no tempo em espaço de estados operando em tempo discreto. Pelas restrições citadas, pode parecer uma classe altamente restrita de modelos (especialmente por serem lineares), no entanto é bastante surpreendente como muitos processos industriais podem ser descritos com precisão por este tipo de modelo [2], [4] e [5]. Por outro lado, existe um grande número de ferramentas disponíveis de projeto de controladores para tais sistemas e modelos. Para empregar tais métodos é necessário o uso de ferramentas de Teoria de Sistemas, Geometria e Álgebra Linear [3] e [4]. Os métodos lineares de identificação por subespaços estão relacionados com modelos da forma:

$$x_{k+1} = Ax_k + Bu_k + Ke_k \tag{1.1}$$

$$y_k = Cx_k + Du_k + e_k \tag{1.2}$$

onde $u_k \in \Re^m$ e $y_k \in \Re^l$ são, respectivamente, os valores medidos das entradas e das saídas no instante k dos processos com m entradas e l saídas, $x_k \in \Re^n$ é o vetor de estados com valor inicial x_0, e_k é um sinal do tipo ruído branco com média zero e matriz de covariância $E(e_k, e_k^T) = \Delta$. A, B, C, D e K são matrizes de dimensões apropriadas. Os algoritmos de identificação por subespaços calculam modelos em espaço de estados, a partir de dados de entrada e saída. É uma prática comum distinguir três casos distintos na identificação de sistemas por subespaços [6]: a) o caso puramente determinístico $e_k = 0$, b) o caso puramente estocástico $u_k = 0$ e c) o caso combinado determinístico e estocástico. Neste trabalho será tratado o caso puramente determinístico. O problema determinístico é: dada uma sequência de dados de entrada e saída determine a ordem n e as matrizes (A, B, C, D) do sistema desconhecido. Existem diversos métodos que solucionam esse problema. Entre os mais populares há o método MOESP [9] e N4SID [6]. O objetivo deste trabalho é implementar em C/C++ e no Matlab o algoritmo N4SID para identificar sistemas discretos invariantes no tempo operando em malha aberta, comparando estes, em relação ao desempenho e tempo de processamento. A principal vantagem de utilizar C/C++ é que permite programar funções próximas à linguagem de máquina, para melhorar o desempenho destes. Na atualidade os principais microcontroladores são facilmente programáveis usando a linguagem C/C++. Além disso, a programação nessa linguagem é aceita em diversas plataformas sendo utilizado em sistemas embarcados como celulares, calculadoras, etc.

1.1. Problema de Identificação.

Dado um conjunto de entradas U_k e saídas Y_k , determine a ordem n do sistema desconhecido e as matrizes A, B, C, D.

1.2. Solução Ideal.

As equações (1.1) e (1.2) podem ser expressas na forma matricial

$$\begin{bmatrix} X_{k+1} \\ Y_k \end{bmatrix} = \Theta \begin{bmatrix} X_k \\ U_k \end{bmatrix} \tag{1.3}$$

Onde

$$\Theta = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \tag{1.4}$$

corresponde aos parâmetros desconhecidos. Se na equação (1.3), as matrizes X_{k+1}, Y_k, X_k e U_k são dadas, então o parâmetro desconhecido Θ pode ser calculado pelo método dos mínimos quadrados,

$$\hat{\Theta} = \begin{bmatrix} \hat{A} & \hat{B} \\ \hat{C} & \hat{D} \end{bmatrix} = \mathrm{Min}_{A,B,C,D} \left\| \begin{bmatrix} X_{k+1} \\ Y_k \end{bmatrix} - \Theta \begin{bmatrix} X_k \\ U_k \end{bmatrix} \right\|_F^2 \tag{1.5}$$

onde $\hat{\Theta}$ denota a estimativa de Θ e $\| \ \|_F^2$ denota a norma de Frobenius de uma matriz. Da equação (1.5), resulta:

$$\hat{\Theta} = \begin{bmatrix} X_{k+1} \\ Y_k \end{bmatrix} \begin{bmatrix} X_k \\ U_k \end{bmatrix}^T \left(\begin{bmatrix} X_k \\ U_k \end{bmatrix} \begin{bmatrix} X_k \\ U_k \end{bmatrix}^T \right)^{-1} \tag{1.6}$$

Então em um caso ideal, quando se têm os dados de entrada, saída e a seqüência de estados para dois instantes de tempo sucessivos k e $k+1$, a identificação do parâmetro Θ na equação (1.6) é trivial. No entanto, na prática, X_{k+1} e X_k não são obtidos e têm que ser estimados dos dados de entrada e saída. Isto é um ponto importante nos métodos de identificação por subespaços. A diferença entre estes métodos reside na forma de como obter a seqüência de estados estimados.

1.3. Equações matriciais por subespaços.

Das equações (1.1) e (1.2) obtém-se:

$$Y_f = \Gamma_i X_f + H_i U_f \tag{1.7}$$

onde a matriz U_f é definida de forma similar à matriz Y_f. As matrizes Y_f e H_i são definidas por:

$$Y_f = \begin{bmatrix} y_i & y_{i+1} & \cdots & y_{i+j-1} \\ y_{i+1} & y_{i+2} & \cdots & y_{i+j} \\ \vdots & \vdots & \ddots & \vdots \\ y_{2i-1} & y_{2i} & \cdots & y_{2i+j-2} \end{bmatrix} \tag{1.8}$$

$$H_i = \begin{bmatrix} D_i & 0 & \cdots & 0 \\ CB & D & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ CA^{i-2}B & CA^{i-3} & \cdots & D \end{bmatrix} \tag{1.9}$$

O número de colunas em Y_f e U_f é $j = N - 2i + 1$, onde N representa a quantidade de dados e i é o número de linhas definido pelo usuário (por exemplo $i = 10$). $\Gamma_i \in \Re^{lixn}$ é a matriz de observabilidade estendida e é definida por:

$$\Gamma_i^T = \begin{bmatrix} (C)^T & (CA)^T \ldots (CA^{i-1})^T \end{bmatrix}^T \tag{1.10}$$

onde $(*)^T$ denota a transposta da matriz $(*)$. $X_p = X_0 = \begin{bmatrix} x_0, & \cdots & x_{j-1} \end{bmatrix}$ e $X_f = X_i = \begin{bmatrix} x_i, & \cdots & x_{i+j-1} \end{bmatrix}$ representam os estados passados e futuros, respectivamente, sendo que o símbolo p denota dados passados e f dados futuros.

1.4. Projeção ortogonal e projeção oblíqua.

A projeção ortogonal do espaço linha de A sobre o espaço linha de B é [6]:

$$A/B = AB(BB^T)^\dagger B \tag{1.11}$$

onde $(*)^\dagger$ denota a pseudo-inversa da matriz $(*)$.
A projeção oblíqua do espaço de linhas de G no espaço de linhas de H sobre o espaço de linhas de J é [6]:

$$G/_H J = [G/H^\perp].[J/H^\perp]^\dagger.J \tag{1.12}$$

onde $(*)^{\perp}$ denota o complemento ortogonal da matriz $(*)$. Propriedades da projeção ortogonal e projeção oblíqua:

$$A_x/A_x^{\perp} = 0 \tag{1.13}$$

$$A_x/_{A_x}C_x = 0 \tag{1.14}$$

Para a prova, ver [6].

2. Materiais e Métodos.

2.1. Método N4SID.

O método N4SID soluciona o problema de identificação determinística recuperando os estados passados e futuros do sistema desconhecido. A sequência de estados X_f pode ser expressa por $X_f = L_pW_p$ como combinação linear das entradas passadas e saídas passadas [6]. Logo, substituindo-se $X_f = L_pW_p$ na equação (1.3) tem-se:

$$Y_f = \Gamma_i L_p W_p + H_i U_f \tag{2.1}$$

na equação (2.1), aplicando a ambos os lados, a projeção ortogonal sobre o espaço linha de $U_f^{\perp}$ resulta:

$$Y_f/U_f^{\perp} = \Gamma_i L_p W_p/U_f^{\perp} \tag{2.2}$$

multiplicando a equação (2.2), a ambos os lados, por $[W_p/U_f^{\perp}]^{\dagger}.W_p$,

$$Y_f/U_f^{\perp}[W_p/U_f^{\perp}]^{\dagger}.W_p = \Gamma_i L_p W_p/U_f^{\perp}[W_p/U_f^{\perp}]^{\dagger}.W_p \tag{2.3}$$

é fácil verificar que $W_p = W_p/U_f^{\perp}[W_p/U_f^{\perp}]^{\dagger}$. Da equação (1.12) obtemos a projeção oblíqua Θ_i e Θ_{i-1} definida por [6]:

$$\Theta_i = Y_f/_{U_f^{\perp}}W_p \tag{2.4}$$

$$\Theta_{i-1} = Y_f^-/_{U_f^{-\perp}}W_p^+ \tag{2.5}$$

Θ_i dado na equação (2.4) pode ser computado da fatoração LQ a partir dos dados de entrada e saída, colocados na forma $[U_p^T U_f^T Y_p^T Y_f^T]^T$.

A equação (2.4) indica que o espaço coluna de Γ_i e Γ_{i-1} podem ser estimados pela SVD de Θ_i e Θ_{i-1}.

$$\Gamma_i = U_1 S_1^{1/2} \tag{2.6}$$

$$\Gamma_{i-1} = \underline{\Gamma_i} \tag{2.7}$$

usando a equação (2.5), computa-se:

$$X_i = \Gamma_i^{\dagger}\Theta_i \tag{2.8}$$

$$X_{i+1} = \Gamma_{i-1}^{\dagger}\Theta_{i-1} \tag{2.9}$$

Por último as matrizes do sistema são estimadas da equação (2.6) [6]:

$$\begin{bmatrix} X_{i+1} \\ Y_i|_i \end{bmatrix} = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} X_i \\ U_i|_i \end{bmatrix} \tag{2.10}$$

3. Processo Simulado.

3.1. Caso SISO.

Em [7] e [8] é citado um exemplo para fazer a identificação em malha fechada de uma planta e seu controlador. Esta planta representa um sistema SISO e é usada neste artigo para fazer a identificação determinística em malha aberta. As matrizes do sistema que simulam a planta são: a = [4.40 1 0 0 0; -8.09 0 1 0 0; 7.83 0 0 1 0; -4 0 0 0 1; 0.86 0 0 0 0]; b = [0.00098 ; 0.01299 ; 0.01859 ; 0.0033 ; -0.00002]; c = [1 0 0 0 0] e d=[0]. Dadas as matrizes do sistema, o passo seguinte é coletar os dados de entrada e saída. Isto é feito pelo seguinte algoritmo:
N = 1000; (número de dados a coletar);
ny = 1; (número de saídas);
nu = 1; (numero de entradas);
u = randn(N,nu); (dados das entradas coletadas);
y = dlsim(a,b,c,d,u); (dados das saidas coletadas)
Para este modelo com sinal de entrada u, foram coletados 1000 dados, dos quais 700 foram aplicados para identificação e o restante para validação. Os sinais usados na identificação são mostrados na figura 3.1.

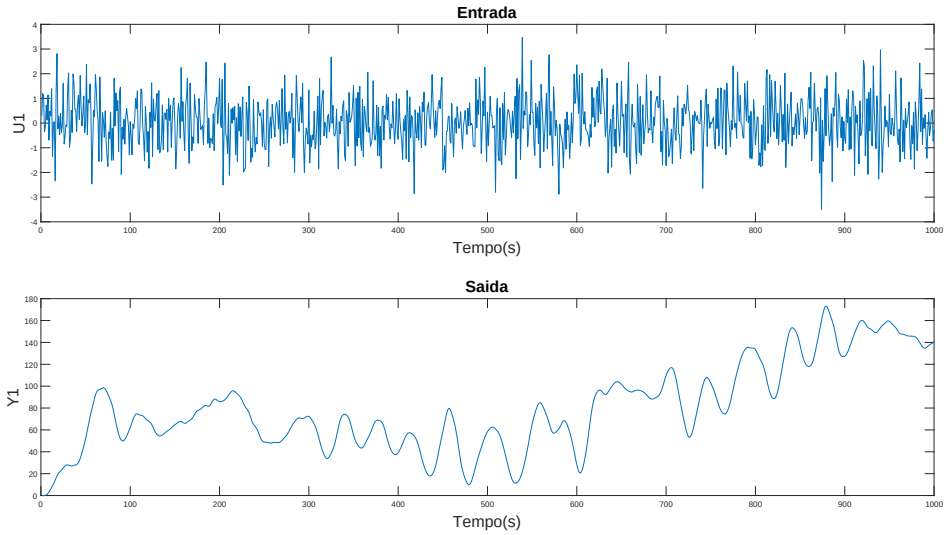

Figura 3.1. *Sinais de entrada e saída usadas na identificação*

Agora deve-se encontrar o melhor modelo que simule o processo. Comparar a simulação do modelo obtido com dados medidos é provavelmente a forma mais usual de se validar um modelo. Nesse caso, deseja-se saber se o modelo reproduz ao longo do tempo os dados observados. Uns dos indicadores de desempenho mais usados são média da variância relativa (MVAF) e o FIT, os quais são definidos como

$$MVAF(\%) = \frac{1}{l}\sum_{i=1}^{l}(1 - \frac{\text{Var}(y-\hat{y})}{\text{Var}(y)}).100 \tag{3.1}$$

$$FIT(\%) = \left(1 - \frac{||y-\hat{y}||_2}{||y - \text{mean}(y)||_2}\right).100 \tag{3.2}$$

onde y é a saída real e $\hat{y}$ é a saída estimada pelo modelo obtido. O índice MVAF é usado pelo SI (System Identification) Toolbox do Matlab. O índice FIT é usado pela função "compare"no Matlab. Estes índices de desempenho são empregados para se avaliar a qualidade do modelo produzido por cada algoritmo, como mostra a tabela 3.1. A ordem $n = 5$ do sistema é dada pelos valores singulares mais significativos, como mostra a figura 3.2.

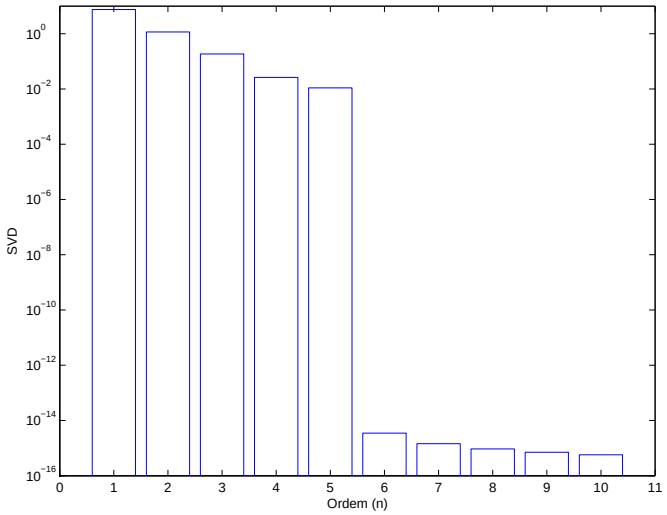

FIGURA 3.2. *Valores Singulares*

TABELA 3.1
Resultados numéricos do desempenho dos algoritmos.

Algoritmo	Tempo (s)	FIT (%)	MVAF (%)
N4SID(MATLAB)	0,060	100	100
N4SID(C/C++)	0,006	100	100

Analisando-se os valores da tabela 3.1, todos os algoritmos tiveram um bom desempenho em termos de validação. Verifica-se que o tempo de processamento para obtensão do modelo é menor para N4SID (C/C++). Com o objetivo de visualizar o desempenho da implementação proposta, optou-se pelo algoritmo N4SID (C/C++) para identificar o processo. A figura 3.3 mostra as saídas do processo real (linha contínua) e aquelas geradas pelo modelo determinístico identificado (linha tracejada). Podese observar que o modelo identificado reproduz muito bem as principais características do processo. Foram consideradas condições iniciais nulas.

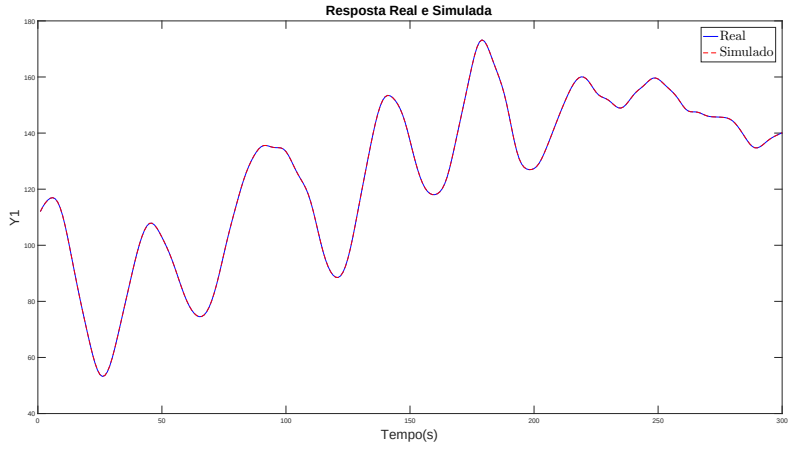

FIGURA 3.3. *Comparação das respostas do processo real (linha contínua) versus modelo (linha tracejada).*

3.2. Caso MIMO.

Para avaliar o desempenho do algoritmo N4SID (C/C++) para o caso MIMO, é usado um modelo em espaço de estados de uma aplicação industrial típica, o tanque para mistura, mostrado na figura 3.4

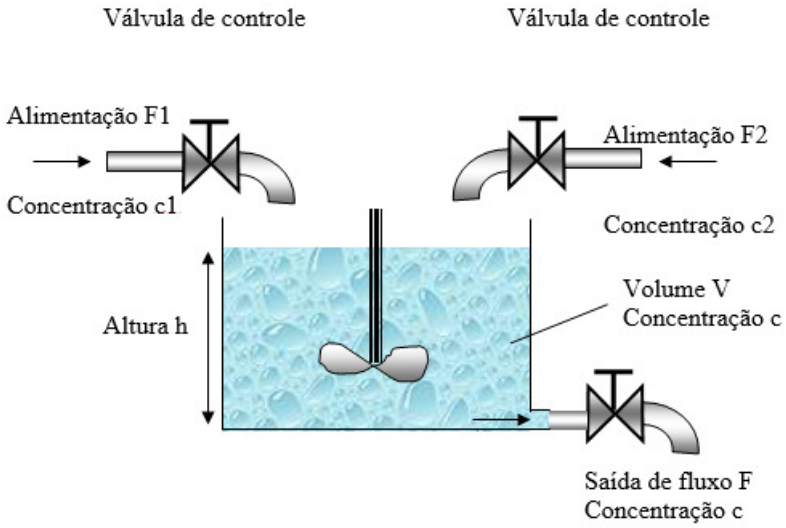

FIGURA 3.4. *Tanque para Mistura*

O tanque é alimentado mediante dois fluxos de entrada $F_1(t)$ e $F_2(t)$. Ambas as alimentações contêm material diluído com concentrações c_1 e c_2 , respectivamente. A saída de fluxo é representada por $F(t)$. Assume-se que o tanque seja bem agitado, tal que a concentração de saída seja igual à concentração $c(t)$ no tanque.

3.2.1. Modelo Dinâmico do Tanque.

As equações de balanço de massa e de concentração são:

$$\frac{dV(t)}{dt} = F_1(t) + F_2(t) - F(t) \tag{3.3}$$

$$\frac{d[c(t)V(t)]}{dt} = c_1F_1(t) + c_2F_2(t) - c(t)F(t) \tag{3.4}$$

onde $V(t)$ é o volume do fluido no tanque. A vazão do fluido na saída depende da altura $h(t)$:

$$F(t) = k\sqrt{h(t)} \tag{3.5}$$

onde k é uma constante experimental. Supondo que o tanque tem área de base S, resulta:

$$V(t) = h(t)S \tag{3.6}$$

$$F(t) = k\sqrt{\frac{V(t)}{S}} \tag{3.7}$$

substituindo-se a equação (3.6) , nas equações (3.3) e (3.4),

$$\frac{dV(t)}{dt} = F_1(t) + F_2(t) - k\sqrt{\frac{V(t)}{S}} \tag{3.8}$$

$$\frac{d[c(t)V(t)]}{dt} = c_1F_1(t) + c_2F_2(t) - c(t)k\sqrt{\frac{V(t)}{S}} \tag{3.9}$$

Considere agora uma situação estável, onde todas as quantidades sejam constantes, é dizer $\overline{F_1}$, $\overline{F_2}$ e $\overline{F}$ para os fluxos, $\overline{V}$ para o volume e $\overline{c}$ para a concentração no tanque. Neste caso, as equações (3.6), (3.7) e (3.8) podem ser reescritas na forma,

$$0 = \overline{F_1} + \overline{F_2} - \overline{F} \tag{3.10}$$

$$0 = c_1\overline{F_1} + c_2\overline{F_2} - c\overline{F} \tag{3.11}$$

$$\overline{F} = k\sqrt{\frac{\overline{V}}{S}} \tag{3.12}$$

3.2.2. Modelo do sistema em espaço de estados.

Caso se perturbe o sistema, é dizer, se as quantidades não são mais constantes, tem-se as seguintes equações:

$$F_1(t) = \overline{F_1} + \mu_1(t) \tag{3.13}$$

$$F_2(t) = \overline{F_2} + \mu_2(t) \tag{3.14}$$

$$V(t) = \overline{c} + \xi_1 \tag{3.15}$$

$$c(t) = \overline{c} + \xi_2 \tag{3.16}$$

Neste caso, μ_1 e μ_2 representa as variáveis de entrada e $\xi_1(t)$ e $\xi_2(t)$ representa as variáveis de estado. Assumindo-se que as quatro quantidades sejam pequenas, as equações (3.7) e (3.8) podem ser representadas,

$$\dot{\xi_1}(t) = \mu_1(t) + \mu_2(t) - \frac{k}{2\overline{V}}\sqrt{\frac{\overline{V}}{S}}\xi_1(t) \tag{3.17}$$

$$\dot{\xi_2}(t)V_0 + \overline{c}\dot{\xi_1}(t) = c_1\mu_1(t) + c_2\mu_2(t) - \overline{c}\frac{k}{2\overline{V}}\sqrt{\frac{\overline{V}}{S}}\xi_1(t) - k\sqrt{\frac{\overline{V}}{S}}\xi_2(t) \tag{3.18}$$

Substituindo-se a equação (3.9) nas equações (3.11) e (3.12) temos,

$$\dot{\xi_1}(t) = \mu_1(t) + \mu_2(t) - \frac{1}{2}\frac{\overline{F}}{\overline{V}}\xi_1(t) \tag{3.19}$$

$$\dot{\xi_2}(t)\overline{V} + \overline{c}\dot{\xi_1}(t) = c_1\mu_1(t) + c_2\mu_2(t) - \frac{1}{2}\overline{c}\frac{\overline{F}}{\overline{V}}\xi_1(t) - \overline{F}\xi_2(t) \tag{3.20}$$

Define-se:

$$\frac{\overline{V}}{\overline{F}} = \theta \tag{3.21}$$

onde Θ corresponde à constante de tempo do tanque. Substituindo-se a equação (3.15) na equação (3.13) resulta,

$$\dot{\xi_1}(t) = \mu_1(t) + \mu_2(t) - \frac{1}{2}\frac{1}{\theta}\xi_1(t) \tag{3.22}$$

Substituindo-se as equações (3.16) e (3.15) na equação (3.14) tem-se,

$$\dot{\xi_2}(t) = \frac{(c_1 - \overline{c})}{\overline{V}}\mu_1(t) + \frac{(c_2 - \overline{c})}{\overline{V}}\mu_2(t) - \frac{1}{\theta}\xi_2(t) \tag{3.23}$$

As equações (3.16) e (3.17) podem ser expressadas na forma,

$$\dot{\xi_1}(t) = -\frac{1}{2}\frac{1}{\theta}\xi_1(t) + 0\xi_2(t) + \mu_1(t) + \mu_2(t) \tag{3.24}$$

$$\dot{\xi_2}(t) = 0\xi_1(t) - \frac{1}{\theta}\xi_2(t) + \frac{(c_1 - \overline{c})}{\overline{V}}\mu_2(t) \tag{3.25}$$

Representando a equação (3.18) na forma matricial,

$$\dot{x}(t) = \begin{bmatrix} \frac{-1}{2\theta} & 0 \\ 0 & \frac{-1}{\theta} \end{bmatrix} x(t) + \begin{bmatrix} 1 & 1 \\ \frac{c_1 - \overline{c}}{\overline{V}} & \frac{c_2 - \overline{c}}{\overline{V}} \end{bmatrix} \mu(t)$$

Onde os estados e entradas são:

$$x(t) = \begin{bmatrix} \xi_1(t) \\ \xi_2(t) \end{bmatrix}$$

$$u(t) = \begin{bmatrix} \mu_1(t) \\ \mu_2(t) \end{bmatrix}$$

Definem-se agora as variáveis de saída:

$$\eta_1(t) = F(t) - \overline{F} \approx \frac{1}{2}\frac{\overline{F}}{\overline{V}}\xi_1(t) = \frac{1}{2\theta}\xi_1(t)$$
$$\eta_2(t) = c(t) - \overline{c} = \xi_2(t)$$

Portanto, as saídas deste sistema linearizado são:

$$y(t) = \begin{bmatrix} \frac{1}{2\theta} & 0 \\ 0 & 1 \end{bmatrix} x(t) \quad \text{onde} \quad y(t) = \begin{bmatrix} \eta_1 \\ \eta_2 \end{bmatrix}$$

Considerando os valores nominais das variáveis da planta: $\overline{F_1} = 0,015m^3/s$, $\overline{F_2} = 0,005m^3/s$, $\overline{F} = 0,02m^3/s$, $c_1 = 1kmol/m^3$, $c_2 = 2kmol/m^3$, $\overline{c} = 1.25kmol/m^3$, $\overline{V} = 1m^3$, $\theta = 50s.$, tem-se o seguinte sistema de estados linearizados em tempo contínuo,

$$\dot{x}(t) = \begin{bmatrix} -0.01 & 0 \\ 0 & -0.02 \end{bmatrix} x(t) + \begin{bmatrix} 1 & 1 \\ -0.25 & 0.75 \end{bmatrix} \mu(t)$$

$$y(t) = \begin{bmatrix} 0.01 & 0 \\ 0 & 1 \end{bmatrix} x(t)$$

com matrizes A ,B ,C ,D

$$A = \begin{bmatrix} -0.01 & 0 \\ 0 & -0.02 \end{bmatrix} ; \; B = \begin{bmatrix} 1 & 1 \\ -0.25 & 0.75 \end{bmatrix} ; \; C = \begin{bmatrix} 0.01 & 0 \\ 0 & 1 \end{bmatrix} ; \; D = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

Em tempo discreto, com tempo de amostragem $T\alpha = 10$ [11], as matrizes A ,B ,C ,D são mostradas a seguir

$$A = \begin{bmatrix} 0.90 & 0 \\ 0 & 0.81 \end{bmatrix} ; \; B = \begin{bmatrix} 9.51 & 9.51 \\ -2.26 & 6.79 \end{bmatrix} ; \; C = \begin{bmatrix} 0.01 & 0 \\ 0 & 1 \end{bmatrix} ; \; D = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

Obtidas as matrizes do sistema, o passo seguinte é coletar os dados de entrada e saída do sistema determinístico. Os dados de entrada e saída foram coletados usando um sinal de entrada binario pseudo-aleatórios (PRBS: pseudo-random binary signal) dado pelo seguinte algoritmo
ny = 2;(numero de saidas)
nu = 2;(numero de entradas)
N = 3000;(quantidade de amostras)
u = randn(N,nu); (dados das entradas coletados)
y = dlsim(a,b,c,d,u); (dados das saidas coletados).
Para este modelo com sinal de entrada u, foram coletados 3000 dados, dos quais 2000 foram usados na identificação e o restante foram usados na validação. A ordem do sistema identificado é $n = 2$. Os resultado da simulação são mostrados na tabela 3.2.

Tabela 3.2
Resultados numéricos do desempenho dos algoritmos.

Algoritmo	Tempo (s)	FIT (%)	MVAF (%)
N4SID(MATLAB)	0,053	100	100
N4SID(C/C++)	0,035	100	100

Analisando-se os valores da tabela 3.2, todos os algoritmos tiveram um bom desempenho em termos de validação. Verifica-se que o tempo de processamento para a obtenção do modelo é menor para N4SID desenvolvido em C/C++. Com o objetivo de visualizar o desempenho da implementação proposta, optou-se pelo algoritmo N4SID (C/C++) para identificar o processo. A figura 3.5 mostra as saídas do processo real (linha contínua) e aquelas geradas pelo modelo determinístico identificado (linha tracejada). Pode-se observar que o modelo identificado reproduz muito bem as principais características do processo. Foram consideradas condições iniciais nulas.

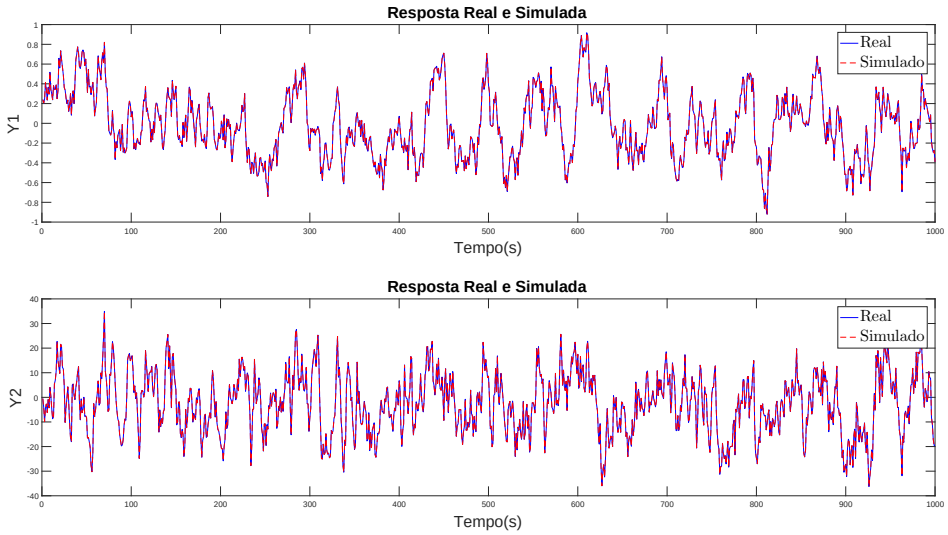

Figura 3.5. *Comparação das respostas do processo real (linha contínua) versus modelo (linha tracejada)*

As matrizes do modelo obtido são dadas por:

$$A = \begin{bmatrix} 0.8187 & -0.0001 \\ -0.0045 & 0.9048 \end{bmatrix} \; ; \; B = \begin{bmatrix} 0.8343 & -2.5046 \\ 0.3105 & 0.1404 \end{bmatrix}$$
$$C = \begin{bmatrix} -0.0180 & 0.3550 \\ -2.7142 & -0.0037 \end{bmatrix} \; ; \; D = 10^{-9} \begin{bmatrix} 0.09 & 0.026 \\ -1.1 & 4.6 \end{bmatrix}$$

4. Conclusões.

Neste trabalho, é implementado o algoritmo N4SID na linguagem de programação C/C++ e no Matlab para identificação deterministica de sistemas discretos invariantes no tempo operando em malha aberta. Dois sistmas em estudo (um SISO e um MIMO) foram usados para avaliar o desempenho destes e seus resultados foram comparados via validação cruzada, usando dois criterios de validação. Para este caso o algoritmo N4SID implementado no Matlab foi o mais lento nos dois casos de identificação. A principal vantagem de utilizar C/C++ é que permite programar funções próximas à linguagem de máquina. Além disso, a programação nessa linguagem é aceita em diversas plataformas sendo utilizado em sistemas embarcados.
A ordem das matrices do sistema do modelo identificado podem diferir da ordem das matrizes do sistema do modelo real.

Agradecimentos. Os autores agradecem ao programa institucional de bolsas de iniciação científica da UFRN, pelo aporte financeiro deste trabalho, PIBITI UFRN.

Referências

[1] Borjas,S.D., Garcia, C. *Subspce identification using the integration of MOESP and N4SID methods applied to the Shell benchmark of a distillation* TEMA-Tend.Mat.Apl.Comput., Vol. 12, (2011), 183-194.

[2] Borjas, S.D.M. Garcia, C. *Modelagem de FCC usando métodos de identificação por predição de erro e por subespaços.* IEEE América Latina, Revista virtual - na Internet,**2**, No. 2, (2004), 108-113.

[3] De Moor, B., Van Overschee P. and Favoreel, W. *Algorithms for subspace state space system identification - an overview.* In B. Datta (Ed.), Applied and computational control, signal and circuits, Vol. 1, pp. 247-311. Birkhauser: Boston (Chapter 6), 1999.

[4] Favoreel, W., De Moor, B. and Van Overschee, P. *Subspace state space system identification for industrial processes.* Journal of Process Control,**10**, No.3, (2000), 149-155.

[5] Roberto, P.; Kurka, G.; Cambraia, H. *Application of a multivariable input-output subspace identification technique in structural analysis*, Journal of Sound and Vibration,**312**, No. 3, (2008), 461-47.

[6] Van Overschee, P.; De Moor, B.*Subspace Identification for Linear Systems: Theory, Implementation, Applications*, Dordrecht: Kluwer Academic Publishers, 1996.

[7] Van Overschee, P.; De Moor, B. *Closed loop subspace systems identification,em "Proc. 36th IEEE Conference on Decision and Control*, pp. 1848-1853, San Diego, 1997.

[8] Verhaegen, M. *Application of a subspace model identification technique to identify LTI systems operating in closed loop*, Automatica,**29**, No 4, (1993), 1027-1040.

[9] Verhaegen, M.; Dewilde, P. *Subspace model identification. part i: the output-error state-space model identification class of algorithms*, International Journal of Control,**56**, No. 1, (1992), 1187-1210.

[10] Akaike, H. *Information theory and an extension of the maximum likelihood principle.* In: Second International Symposium on Information Theory, Budapest, Hungary. Petrov, B.N.; Csaki, F.; (Eds.), pp. 267-281, 1973.

[11] Borjas, S.D.M. *Estudo da identificação por subespaços em malha aberta e fechada e proposta de novos algoritmos*, Tese de Doutorado, POLI, USP, São Paulo, SP, 2009.