INFLUÊNCIA DE SOFTWARES E SISTEMAS OPERACIONAIS NA SIMULAÇÃO DE MODELOS DINÂNICOS NÃO LINEARES

Felipe Lulli Milani*, Wilson Rocha Lacerda Junior*, Samir Angelo Milani Martins*, Erivelton Geraldo Nepomuceno*

*GCOM - Grupo de Controle e Modelagem, Departamento de Engenharia Elétrica
Universidade Federal de São João del-Rei, Praça Frei Orlando 170 - Centro, 36307-352
São João del-Rei, Minas Gerais, Brasil

Emails: felipe_lulli@hotmail.com, wilsonrljr@outlook.com, martins@ufsj.edu.br, nepomuceno@ufsj.edu.br

Abstract— This paper presents an analysis of the influence of softwares and operating systems and its respective hardware in the simulation of nonlinear dynamical systems. Furthermore, different mathematical equivalence of polynomial model of the Chua's circuit were chosen and simulated to highlight the inaccuracy of the results of each form of implementation of the analyzed model. The main results of the paper shows that equivalent mathematical models have different results when simulated in mathematical software, also with distinction when applied on different operating systems.

Keywords— Chua's circuit, nonlinear dynamic, chaos, IEEE 754

Resumo— Este artigo apresenta uma análise da influência de softwares e sistemas operacionais e seus respectivos hardwares na simulação de sistemas dinâmicos não lineares. Além disso, diferentes equivalências matemáticas do modelo polinomial do circuito de Chua foram escolhidas e simuladas para destacar a imprecisão dos resultados apresentados em cada forma de implementação do modelo analisado. Os principais resultados obtidos nesse artigo mostram que modelos matemáticos equivalentes, possuem resultados distintos quando simulados em softwares matemáticos, havendo também distinção quando aplicados em sistemas operacionais diferentes.

Palavras-chave— Circuito de chua, dinâmica não linear, caos, IEEE 754

1 Introdução

O advento da computação numérica permitiu um grande avanço para a análise de sistemas dinâmicos e, por meio de simulações computacionais, pode-se verificar o comportamento desse conjunto, principalmente na área de sistemas dinâmicos não lineares, onde existem pesquisas relacionadas a sistemas caóticos (Monteiro, 2006; Lozi and Valrose, 2012; Martins et al., 2013).

Cáculos realizados em computadores levaram pesquisadores a identificarem suspeitas sobre esses sistemas dinâmicos não lineares, passando pela descoberta de Lorenz (Lorenz, 1963) e pelo circuito de Chua (Chua et al., 1993), com a finalidade de contribuir para a compreensão de como esses sistemas são analisados pelos computadores.

O aumento no estudo de sistemas caóticos deve-se a Lorenz, graças a teoria do caos. A principal característica desses sistemas é a dificuldade em se fazer previsões a longo prazo, pelo simples motivo de que, havendo uma pequena alteração na condição inicial do sistema, as trajetórias divergem completamente ao longo do tempo.

Para a análise desses sistemas caóticos utiliza-se simulações de modelos matemáticos. Como essas simulações são feitas em computadores, deve-se verificar como os cálculos são processados pelo computador. Esses cálculos são feitos a partir da norma IEEE 754: Ponto Flutuante, criada para padronizar o cálculo numérico nos computadores. A norma propõe um método para computação com ponto flutuante que gera o mesmo resultado, independente se o processamento é feito em hardware, software ou uma combinação de ambos. Os resultados dos cálculos serão idênticos, independentes da implementação, dado os mesmos valores de entrada (*IEEE Standard for floating-point arithmetic*, 2008). Porém, como constatado por Rahman et al. (2014) e Lozi and Valrose (2012), esses cálculos podem conter erros inesperados, como erros de arredondamento e limitação dos números computacionais em relação aos números reais, fazendo com que haja divergência entre a trajetória calculada e a verdadeira.

A simulação de sistemas não lineares vem sendo amplamente estudada na área e um grande número de pesquisadores acreditam precipitadamente nos resultados obtidos. Na engenharia, alguns dos softwares utilizados para realizar essas simulações são o Matlab e Scilab, que fazem cálculos no computador utilizando-se da aritmética de ponto flutuante, estabelecida pela norma IEEE 754. Porém, como apresentado em Lozi and Valrose (2012), "no simples caso de um sistema dinâmicos discreto (mapa de Hénon), há dúvidas quanto a natureza dos resultados computacionais: longas pseudo-órbitas instáveis ou atratores estranhos ?". Além disso, se tratando da simulação de sistemas não lineares é caracterísitco o acúmulo de erros, como destacado por Liao and Wang (2014), Bunimovich (2014), Galias (2013), Nepomuceno and Martins (2016).

O objetivo desse artigo é apresentar um es-

tudo sobre a influência do software e do sistema operacional utilizado, na simulação de modelos com dinâmica não-linear. Verificar o quão modelos matematicamente equivalentes são diferentes do ponto de vista computacional, quando simulados em diferentes softwares e sistemas operacionais.

O artigo está estruturado da seguinte forma: a presente seção apresentou uma breve introdução histórica sobre o tema a ser trabalhado no restante deste artigo. Na seção 2 são levantados os conceitos preliminares, suficientes para o entendimento do trabalho. Na seção 3 apresenta os métodos utilizados para se chegar aos resultados apresentados na seção 4. A seção 5 trata da conclusão do trabalho e propostas de possíveis trabalhos futuros.

2 Conceitos Preliminares

2.1 O circuito de Chua

O circuito de Chua é um dos sistemas mais conhecidos e estudados quando se trata de dinâmica não linear e caos (Aguirre, 2007). Esse circuito é constituído por uma única resistência não linear e quatro componentes de circuitos lineares: dois capacitores, um resistor e um indutor (Dedieu et al., 1993). Essa resistência negativa, conhecida como diodo de Chua, é o elemento que confere energia ao sistema, permitindo, assim, manter o circuito oscilando autonomamente (Aguirre, 2007).

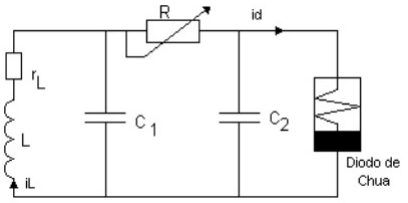

Figura 1: Circuito de Chua. Fonte: (Corrêa, 1997).

A curva característica desse elemento é apresentada na figura 2.

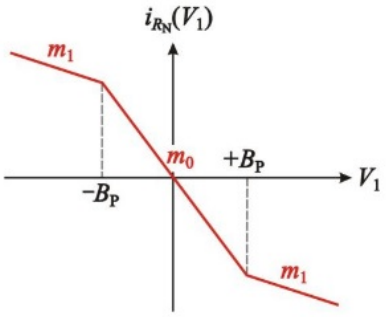

Figura 2: Curva característica do Diodo de Chua. Fonte: (Valerio, 2014).

O comportamento do circuito é descrito pelas equações apresentadas abaixo (Aguirre, 2007):

$$\begin{cases} C_1 \frac{dv_{C1}}{dt} & = \frac{v_{C2}-v_{C1}}{R} - i_d(v_{C1}) \\ C_2 \frac{dv_{C2}}{dt} & = \frac{v_{C1}-v_{C2}}{R} + i_L \\ L \frac{di_L}{dt} & = -v_{C2}, \end{cases} \quad (1)$$

sendo $v_C i$ a tensão sobre o capacitor C_i e i_L a corrente no indutor.

A corrente que circula pelo diodo de Chua é definida por:

$$i_{C_1} = \begin{cases} m_0 v_{C1} + B_p(m_0 - m_1), & |v_{C1}| < -B_p \\ m_1 v_{C1}, & |v_{C1}| \geq B_p \\ m_0 v_{C1} + B_p(m_1 - m_0), & v_{C1} > +B_p, \end{cases} \quad (2)$$

em que B_p,m_0 e m_1 o ponto de ruptura e as inclinações da função linear por partes,respectivamente. Em (Aguirre, 2007) foi apresentado um modelo NAR com dinâmica equivalente à do modelo 2. Este modelo será posteriormente apresentado e utilizado no contexto deste trabalho.

2.2 Identificação de Sistemas e o Modelo Polinomial do circuito de Chua

A Identificação de Sistemas é a área do conhecimento que estuda maneiras de modelar e analisar sistemas, na tentativa de encontrar algum padrão nas observações (Billings, 2013; Ljung, 1987). Em muitos casos é preferível usar técnicas de identificação para se obter modelos que descrevem o comportamento de um sistema (Aguirre, 2007).

As principais etapas do processo de identificação de sistemas consiste em (Aguirre, 2007): *i*) testes dinâmicos e coleta de dados; *ii*) escolha da representação matemática a ser usada; *iii*) determinação da estrutura do modelo; *iv*) estimação de parâmetros e *v*) validação do modelo.

Um modelo NARX (nonlinear autoregressive model with exogenous variables) polinomial pode ser definido por: (Martins and Aguirre, 2014):

$$\begin{aligned} y(k) \quad = \quad & F^{\ell}[y(k-1), ..., y(k-n_y) \\ & u(k-d), ..., u(k-n_u)] + e(k), \end{aligned} \quad (3)$$

em que n_y, n_u, d, $u(k)$ e $y(k)$ são os atrasos em y, em u, o tempo morto, o sinal de entrada e o sinal de saída no instante k, respectivamente e $e(k)$ representa ruído e possíveis incertezas.

2.3 Norma IEEE 754: Ponto Flutuante

O padrão IEEE 754 foi definido pelo Instituto de Engenheiros Eletricistas e Eletrônicos, com a finalidade de padronizar as operações e representações de números binários com aritmética de ponto flutuante.

Essa norma foi criada pela necessidade da unificação dos cálculos em computadores, pois antes desse padrão utilizavam-se maneiras de calcular números com pontos flutuantes desenvolvidos pelas suas fabricantes e isso criava uma incompatibilidade entre essas máquinas.

Para a representação dos números com ponto flutuante a norma criou padrões para computadores que utilizam 32 bits e 64 bits. Para o formato simples de 32 bits, utiliza-se o primeiro bit para bit de sinal, 8 bits para o expoente e 23 bits para a mantissa. O computador que possui 64 bits, tem o primeiro bit para bit de sinal, 10 bits para o expoente e 53 bits para a mantissa. Consequentemente por possuir maior número de bits os computadores que utilizam 64 bits tem maior precisão utilizando cálculos com ponto flutuante.

3 Metodologia

Para execução da metologia a seguir proposta, foram utilizados dois notebooks: o primeiro, Hewlett-Packard(HP) - Pavilion G42, equipado com Intel core I5 450M, 2GB de memória ram DDR3 e o segundo sendo um notebook Samsung Chronos, equipado com processador Intel core I5 2450M e 8GB de memória ram DDR3. Os softwares utilizados foram Scilab, versão 5.5.2 (tanto para Windows, quanto para Linux) e Matlab, versão R2012b (para Windows) e R2012a (para Linux). Os sistemas operacionais utilizados foram Windows 7 Home Premium para ambos os computadores - 64bits, Linux Fedora 22 no primeiro computador apresentado e Linux Ubuntu 15.10 para o segundo.

As simulações foram feitas com intervalos de 0 a 1000 amostras, porém para melhor visualização, as Figuras foram inseridos entre 500 a 1000 iterações, que equivalem ao intervalo de tempo de 3,6μ a 7,2μ do modelo simulado.

O modelo NAR (Nonlinear Autoregressive model) polinomial para o circuito de Chua utilizado nas análises é o mesmo modelo apresentado em (Aguirre, 2007):

$$\begin{aligned} y_1(k) &= 3.523y_1(k-1) - 4.2897y_1(k-2) \\ &\quad -0.2588y_1(k-4)\underline{-1.7784y_1(k-1)^3} \\ &\quad +2.0652y_1(k-3)\underline{+6.1761y_1(k-1)^2} \\ &\quad \times y_1(k-2) + 0.1623y_1(k-1)y_1(k-2) \\ &\quad \times y_1(k-4) - 2.7381y_1(k-1)^2y_1(k-3) \\ &\quad -5.5369y_1(k-1)(k-2)^2 \\ &\quad +0.1031y_1(k-2)^3 + \underbrace{0.4623y_1(k-4)^3} \\ &\quad -0.5247y_1(k-2)^2y_1(k-4) \\ &\quad -1.8965y_1(k-1)y_1(k-3)^2 \\ &\quad +5.4255y_1(k-1)y_1(k-2)y_1(k-3) \\ &\quad + \underbrace{0.7258y_1(k-2)y_1(k-4)^2} \\ &\quad -1.7684y_1(k-3)y_1(k-4)^2 \\ &\quad + \underbrace{1.18y_1(k-3)^2y_1(k-4)}. \end{aligned} \tag{4}$$

Escolheram-se duas equivalências matemáticas do modelo 4, compondo um total de três modelos matematicamente equivalentes. Assim, pretende-se verificar como se dá o resultado da simulação destas equivalências matemáticas em diferentes softwares e sistemas operacionais, bem como verificar situações em que a simulação dos modelos não é mais confiável. As alterações realizadas para se obter as equivalências matemáticas estão destacadas, para evidenciar o quão simples foram essas alterações.

$$\begin{aligned} y_2(k) &= 3.523y_2(k-1) - 4.2897y_2(k-2) \\ &\quad -0.2588y_2(k-4)\underline{-1.7784y_2(k-1)} \\ &\quad \underline{\times y_2(k-1)y_2(k-1) + 2.0652y_2(k-3)} \\ &\quad \underline{+6.1761y_2(k-1)y_2(k-1)y_2(k-2)} \\ &\quad +0.1623y_2(k-1)y_2(k-2)y_2(k-4) \\ &\quad -2.7381y_2(k-1)^2y_2(k-3) \\ &\quad -5.5369y_2(k-1)(k-2)^2 \\ &\quad +0.1031y_2(k-2)^3 + 0.4623y_2(k-4)^3 \\ &\quad -0.5247y_2(k-2)^2y_2(k-4) \\ &\quad -1.8965y_2(k-1)y_2(k-3)^2 \\ &\quad +5.4255y_2(k-1)y_2(k-2)y_2(k-3) \\ &\quad +0.7258y_2(k-2)y_2(k-4)^2 \\ &\quad -1.7684y_2(k-3)y_2(k-4)^2 \\ &\quad +1.18y_2(k-3)^2y_2(k-4). \end{aligned} \tag{5}$$

$$
\begin{aligned}
y_3(k) \quad = \quad & 3.523y_3(k-1) - 4.2897y_3(k-2) \\
& -0.2588y_3(k-4) - 1.7784y_3(k-1)^3 \\
& +2.0652y_3(k-3) + 6.1761y_3(k-1)^2 \\
& \times y_3(k-2) + 0.1623y_3(k-1)y_3(k-2) \\
& \times y_3(k-4) - 2.7381y_3(k-1)^2 \\
& \times y_3(k-3) - 5.5369y_3(k-1) \\
& \times (k-2)^2 + 0.1031y_3(k-2)^3 \\
& \underbrace{+0.4623y_3(k-4)y_3(k-4)y_3(k-4)} \\
& -0.5247y_3(k-2)^2y_3(k-4) \\
& -1.8965y_3(k-1)y_3(k-3)^2 \\
& +5.4255y_3(k-1)y_3(k-2)y_3(k-3) \\
& \underbrace{+0.7258y_3(k-2)y_3(k-4)y_3(k-4)} \\
& -1.7684y_3(k-3)y_3(k-4)^2 \\
& \underbrace{+1.18y_3(k-3)y_3(k-3)y_3(k-4)}. \quad (6)
\end{aligned}
$$

Como pode ser observado, as Equações (4), (5) e (6) são matematicamente equivalentes. Neste trabalho será verificada a possível equivalência computacional destes modelos, quando simulados em diferentes softwares e sistemas operacionais. Ademais, espera-se comparar, dadas as possíveis diferenças, quais delas estão mais próximas do valor esperado, de acordo com o que será proposto a seguir.

A fim de estabalecer um critério de confiabilidade dos dados obtidos nos resultados, foi proposto um índice de dispersão baseado no desvio padrão. Considerando a média aritimética dos resultados como a esperança matemática, calculou-se o desvio padrão dos resultados e estabeleceu-se uma condição de parada que define o máximo de iterações a serem realizadas na simulação de forma que a resposta do sistema não apresente um erro superior a 10% em relação ao valor esperado.

Com os modelos equivalentes citados, foram realizadas simulações de acordo com os métodos a seguir: simulações realizadas no mesmo Sistema Operacional; simulações realizadas em diferentes Sistemas Operacionais; simulações realizadas em diferentes Hardwares; em diferentes Softwares (Matlab e Scilab);

4 Resultados

Foram feitas simulações do modelo $y_1(k)$, utilizando softwares e sistemas operacionais diferentes. Pode-se ver o resultado na figura 3. Foi realizada as simulações das três equivalências (y_1,y_2,y_3) em todos os softwares e sistemas operacionais. Os resultados podem ser analisados na Figura 4. Os atratores dupla volta do modelo 4 $y_1(k)$, foram simulados no notebook HP, utilizando todos os softwares e sistemas operacionais disponíveis podendo ser analisado na Figura 5. Observa-se que, apesar das séries temporais divergirem rapidamente quando calculadas em diferentes softwares e sistemas operacionais, os atrator permanecem com sua imagem. Ressalta-se que em alguns casos, atratores são utilizados como ferramentas para validação de modelos matemáticos Pereira et al. (2014). Aqui é mostrado que isto pode ser um problema, uma vez que não se sabe, a priori, quais das séries geradas mais se aproxima da órbita verdadeira.

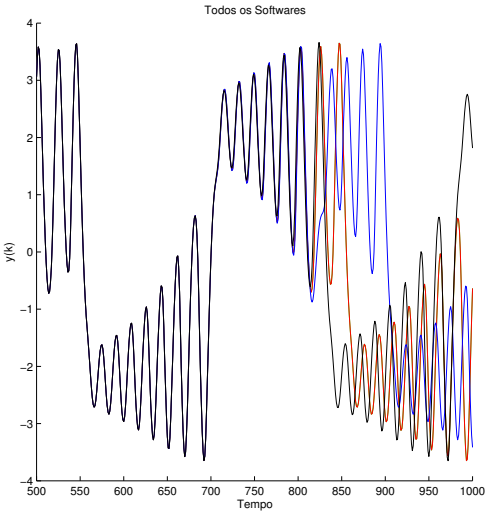

Figura 3: Simulação do modelo $y_1(k)$ nos diferentes softwares e sistemas operacionais. Cor Verde - Simulação do modelo $y_1(k)$ no Matlab e no Linux, Cor Azul - Simulação do modelo $y_1(k)$ no Matlab e no Windows, Cor Vermelha - Simulação do modelo $y_1(k)$ no Scilab e no Linux, Cor Preta - Simulação do modelo $y_1(k)$ no Scilab e no Windows

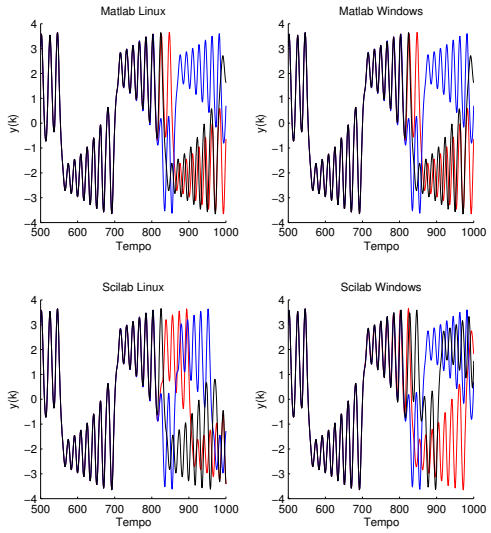

Figura 4: Simulação das três equivalências matemáticas em todos os softwares e sistemas operacionais. Cor Vermelha - Primeira equivalência, modelo $y_1(k)$, Cor Azul - Segunda equivalência, modelo $y_2(k)$, Cor Preta - Terceira equivalência, modelo $y_3(k)$.

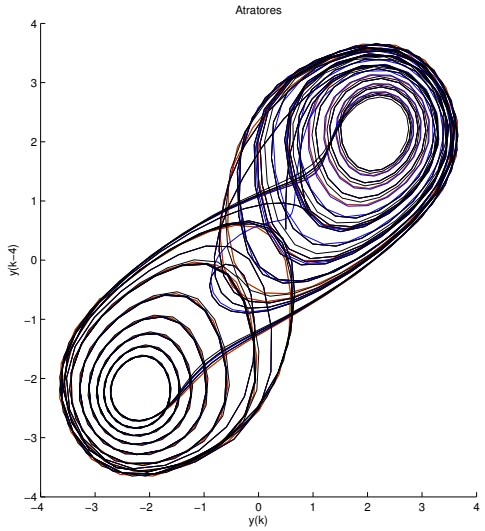

Figura 5: Simulação do Atrator utilizando o modelo $y_1(k)$ usando todos os softwares e sistemas operacionais. Cor verde - $y_1(k)$ no Matlab e no Linux, Cor Vermelha - $y_1(k)$ no Matlab e no Windows, Cor Azul - $y_1(k)$ no Scilab e no Linux, Cor Preta - $y_1(k)$ no Scilab e no Windows.

O gráfico da Figura 6 mostra a relação do desvio padrão pelo tempo dos resultados obtidos nos quatro softwares utilizados.

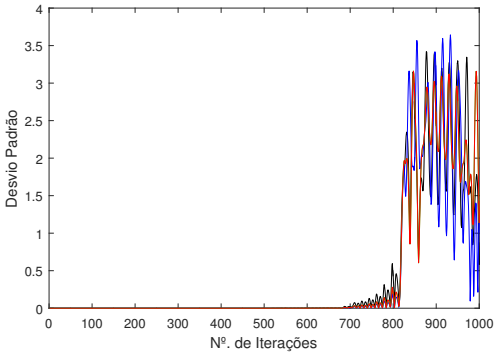

Figura 6: Simulação do desvio padrão de todas as equivalências em todos os softwares. Cor Verde - Desvio padrão no Matlab e no Linux, Cor Vermelha - Desvio padrão no Matlab e no Windows, Cor Azul - Desvio padrão no Scilab e no Linux, Cor Preta - Desvio padrão no Scilab e no Windows

Como se trata do cálculo do desvio padrão, espera-se que, em simulações com resultados computacionalmente equivalentes, o desvio padrão seja nulo. Contudo, como pode ser observado na Figura 6, os resuldados obtidos não são computacionalmente equivalentes. Para o Matlab executado em Windows e Linux, erros maiores que 10% em relação ao maior valor apresentado nos resultados acontecem a partir da 816 iterações, que equivalem a $9,79\mu$ segundos. Já no Scilab, executado em Linux Fedora, o erro acima de 10% acontece a partir de 817 iterações ($9,80\mu$ segundos) e no Scilab Windows a partir de 795 iterações, equivalente a $9,54\mu$ segundos).

É importante ressaltar que as simulações não apresentaram diferenças quando realizadas em hardwares diferentes. Além disso, seriam necessárias simulações do modelo com todas as equivalências matemáticas possíveis para uma melhor aproximação do valor esperado para, consequentemente, estabelecer qual delas seria a mais confiável.

5 Conclusões

Este trabalho apresentou uma análise da influência de softwares, hardwares e sistemas operacionais na simulação de sistemas dinâmicos não lineares. Para tal análise, foram utilizados os softwares Scilab e Matlab, executados em dois hardwares diferentes e dois sistemas operacionais diferentes para simular diferentes equivalências matemáticas de um modelo NAR do circuito de Chua. Além disto, propôs-se um índice para verificar a confiabilidade das simulações dado um erro máximo pré estabelecido.

Os resultados obtidos mostraram que a mesma equivalência matemática de um modelo pode apresentar diferentes resultados, alterando-se apenas o software ou o sistema operacional uti-

lizado para executar a simulação. Foi constatado que o software Scilab apresenta diferentes valores para uma mesma equivalência matemática de um modelo, se este é simulado em sistemas operacionais diferentes.

Além disso, foi identificado que diferentes representações matemáticas, mas equivalente entre si, também influenciam na resposta obtida computacionalmente. Tanto o software Scilab quanto o Matlab se mostraram passíveis desta questão, apresentando diferenças nos resultados em todos os sistemas operacionais e hardwares testados. Contudo, uma mesma equivalência matemática executada em um mesmo software e sistema operacional, alterando-se apenas o hardware, apresentou o mesmo resultado.

Trabalhos futuros devem incluir mais softwares utilizados para simulações de sistemas não lineares, bem como outros sistemas operacionais e hardwares. Pretende-se, também, investigar a causa matemática das divergências apresentadas neste trabalho, à luz da norma IEEE 754 que versa sobre aritmética de pontos flutuantes.

Agradecimentos

Os autores agradecem à UFSJ e às agências CAPES, CNPq e FAPEMIG pelo apoio.

Referências

Aguirre, L. A. (2007). *Introdução à identificação de sistemas – Técnicas lineares e não lineares aplicadas a sistemas reais*, Editora UFMG.

Billings, S. A. (2013). *Nonlinear system identification: NARMAX methods in the time, frequency, and spatio-temporal domains*, Wiley.

Bunimovich, L. A. (2014). Short- and long-term forecast for chaotic and random systems (50 years after lorenz's paper), **27**(9): R51–R60.

Chua, L. O., Wu, C. W., Huang, A. and Zhong, G.-Q. (1993). A universal circuit for studying and generating chaos-part 1: Strange attractors, *IEEE Transactions on Circuits and Systems-I–Fundamental Theory and Applications* **40**(10).

Corrêa, M. V. (1997). *Identificação de sistemas dinâmicos não-lineares utilizando modelos narmax racionais – aplicações a sistemas reais*, Master's thesis, Universidade Federal de Minas Gerais.

Dedieu, H., Kennedy, M. P. and Hasler, M. (1993). Chaos shift keying - Modulation and demodulation of a chaotic carrier using self-synchronizing chua circuits, *Ieee Transactions on Circuits and Systems Ii-Analog and Digital Signal Processing* **40**(10): 634–642.

Galias, Z. (2013). The dangers of rounding error for simulations and analysis of nonlinear circuits and systems - and how to avoid them, *IEEE Circuits and Magazine* .

IEEE Standard for floating-point arithmetic (2008). IEEE Std 754-2008.

Liao, S. and Wang, P. (2014). On the mathematically reliable long-term simulation of chaotic solutions of Lorenz equation in the interval [0,10000], *Science China Physics, Mechanics and Astronomy* **57**(2): 330–335.

Ljung, L. (1987). *System identification: theory for the user*, Prentice-Hall information and system sciences series, Prentice-Hall.

Lorenz, E. (1963). Deterministic nonperiodic flow, *J Atmospheric Science* .

Lozi, R. and Valrose, P. (2012). Can we trust in numerical computations of chaotic solutions of dynamical systems ?, *Proceedings in Chaos* pp. 1–41.

Martins, S. A. M. and Aguirre, L. A. (2014). Narx modelling of the bouc-wen model, *Anais do XX Congresso Brasileiro de Automática* pp. 2051–2057.

Martins, S. A. M., Nepomuceno, E. G. and Barroso, M. F. S. (2013). Improved structure detection for polynomial narx models using a multiobjective error reduction ratio, *Journal of Control, Automation and Electrical Systems* **24**(6): 764–772.

Monteiro, L. H. A. (2006). *Sistemas dinâmicos*, Editora Livraria da Física.

Nepomuceno, E. G. and Martins, S. A. M. (2016). A lower bound error for free-run simulation of the polynomial narmax, *Systems Science and Control Engineering* .

Pereira, L. O., Sansão, J. P. H., Cardoso, A. S. V. and Mozelli, L. A. (2014). Plataforma didática e experimental para investigações do circuito de chua, *Anais do XX Congresso Brasileiro de Automática* .

Rahman, A., Islam, A. T. M. S., Khalid, M., Abdullah-Al-Kafi and Rahman, M. (2014). Optimized hardware architecture for implementing IEEE 754 standard double precision floating point adder / subtractor, pp. 147–152.

Valerio, L. R. (2014). Dinâmica não-linear e caos: o circuito de chua, Trabalho de Conclusão de Curso. Universidade Federal de Alfenas.