

UNIVERSIDADE FEDERAL DE UBERLÂNDIA

FACULDADE DE ENGENHARIA ELÉTRICA

PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA



LUIS PAULO FAGUNDES

**ALGORITMOS GENÉTICOS APLICADOS NA ESCOLHA DA TAXA DE AMOS-
TRAGEM EM IDENTIFICAÇÃO DE SISTEMAS**

UBERLÂNDIA

2016

LUIS PAULO FAGUNDES

ALGORITMOS GENÉTICOS APLICADOS NA ESCOLHA DA TAXA DE AMOSTRA-
GEM EM IDENTIFICAÇÃO DE SISTEMAS

Dissertação de mestrado apresentada à Universidade
Federal de Uberlândia, como exigência parcial para
a obtenção do título de mestre em Ciências.

Área de concentração: Controle e Automação

Orientador: Dr. Fábio Vincenzi Romualdo da Silva
Coorientador: Dr. Márcio José da Cunha

Uberlândia

2016

Dados Internacionais de Catalogação na Publicação (CIP)
Sistema de Bibliotecas da UFU, MG, Brasil.

F156a
2016 Fagundes, Luis Paulo, 1984-
Algoritmos genéticos aplicados na escolha da taxa de amostragem
em identificação de sistemas / Luis Paulo Fagundes. - 2016.
87 f. : il.

Orientador: Fábio Vincenzi Romualdo da Silva.
Coorientador: Márcio José da Cunha.
Dissertação (mestrado) - Universidade Federal de Uberlândia,
Programa de Pós-Graduação em Engenharia Elétrica.
Inclui bibliografia.

1. Engenharia elétrica - Teses. 2. Sistemas não lineares - Teses. 3.
Algoritmos genéticos - Teses. 4. Células a combustível - Teses. I. Silva,
Fábio Vincenzi Romualdo da, 1974- II. Cunha, Márcio José da. III.
Universidade Federal de Uberlândia, Programa de Pós-Graduação em
Engenharia Elétrica. IV. Título.

CDU: 621.3

LUIS PAULO FAGUNDES

ALGORITMOS GENÉTICOS APLICADOS NA ESCOLHA DA TAXA DE AMOSTRA-
GEM EM IDENTIFICAÇÃO DE SISTEMAS

Dissertação de mestrado apresentada à Universidade
Federal de Uberlândia, como exigência parcial para
a obtenção do título de mestre em Ciências.

Data de aprovação:

_____, _____ de _____ de _____.

Banca examinadora:

Prof. Dr. Fábio Vincenzi Romualdo da Silva.

Universidade Federal de Uberlândia

Prof. Dr. Márcio José da Cunha.

Universidade Federal de Uberlândia

Prof. D.ra Aline Fernanda Bianco.

Centro Federal de Educação Tecnológica de Minas Gerais

Prof. Dr. Henrique José Avelar

Centro Federal de Educação Tecnológica de Minas Gerais

Uberlândia

2016

AGRADECIMENTOS

Agradeço primeiramente a Deus e aos meus pais que me conduziram até onde estou hoje.

Um muito obrigado a minha namorada Amanda por todo carinho e companheirismo, sua doçura tornou todos os fardos mais leves.

Aos meus orientadores Dr. Fábio Vincenzi e Dr. Márcio Cunha por me orientarem na execução deste trabalho e pelas lições ensinadas e todas as dúvidas sanadas.

Agradeço aos companheiros de estrada Frederico, Renato e Jean, com toda certeza vocês tornaram essas viagens muito menos longas.

Um agradecimento especial aos meus antigos professores Aline Bianco, Henrique Avelar e Kleber Fontoura que me inspiraram a trilhar o caminho acadêmico, a eles meu eterno agradecimento.

RESUMO

FAGUNDES, LUIS PAULO. **Algoritmos genéticos aplicados na escolha da taxa de amostragem em identificação de sistemas**. 2016. 85 p. Dissertação (Mestrado) – Programa de Pós-Graduação em Engenharia Elétrica, Faculdade de Engenharia Elétrica da Universidade Federal de Uberlândia. Uberlândia, 2016.

O presente trabalho tem como principal objetivo introduzir um novo método para seleção do tempo de amostragem dos sinais de entrada e saída utilizados no processo de identificação de sistemas utilizando representação NARMAX.

Para atingir tal objetivo é proposto um algoritmo genético que utiliza um sinal superamostrado, isto é, um sinal amostrado na mais alta frequência possível, e posteriormente taxas de decimação são usadas para criar diferentes indivíduos a partir do sinal superamostrado. A avaliação dos indivíduos utiliza um processo de identificação de sistemas com representação NARMAX. A avaliação do método proposto utilizou um algoritmo genético desenvolvido no software Matlab®.

O método proposto foi aplicado no processo de identificação do modelo de temperatura de uma célula a combustível de membrana polimérica e os resultados são apresentados.

Palavras-chave: Identificação de sistemas não lineares, Algoritmos genéticos, célula combustível, NARMAX.

ABSTRACT

FAGUNDES, Luis Paulo. Genetic Algorithm applied to the sample's time choice in system identification. 2016. 85 p. Dissertation (Master's Degree) – Programa de Pós-Graduação em Engenharia Elétrica, Faculdade de Engenharia Elétrica da Universidade Federal de Uberlândia. Uberlândia, 2016.

The present work has as the main goal to introduce a new method to select the sample time of input and output signals used in the identification process using NARMAX representation.

To achieve this goal is proposed a genetic algorithm wich uses a supersampled signal, i.e., a signal sampled in the most high frequency available, and later decimation rates are used to create different individuals from the high frequency sample signal. The individuals evaluation uses a system identification with NARMAX representation. The evaluation of the proposed method used a genetic algorithm developed in the software Matlab®.

The proposed method was applied in the process identification of a polimeric membrane fuel cell temperature model and the results are presented.

Key-words: Nonlinear system identification, genetic algorithm, fuel cell, NARMAX.

LISTA DE ILUSTRAÇÕES

Figura 1 – Sinal Superamostrado e sinal decimado.....	13
Figura 2 - Estrutura baseada em árvore para modelo polinomial não linear.....	15
Figura 3 - Estrutura de cromossomo utilizando codificação binária.	16
Figura 4 - Fluxograma de identificação de sistemas	21
Figura 5 - Resposta de um forno elétrico ao ruído quantizado.....	22
Figura 6 - Entrada de excitação do sistema ($u(k)$).....	29
Figura 7 - Resposta do sistema da equação 2.20 ao sinal de entrada da figura 5.....	29
Figura 8 - Comparação entre sistema da equação 2.20 e modelo da equação 2.21.....	30
Figura 9 - Comparação entre sistema identificado em um passo a frente e infinitos passos a frente.....	33
Figura 10 - Exemplo de recombinação por crossover	39
Figura 11 - Exemplo de recombinação por crossover com ponto de corte duplo.	39
Figura 12 - Mutação representada em vermelho.	40
Figura 13 - Função de autocovariância de $y(k)$	48
Figura 14 - Função de autocovariância de $y^2(k)$	48
Figura 15 - Validação de modelo encontrado com taxa de decimação de 136.	49
Figura 16 - Validação de modelo encontrado com taxa de decimação de 180.	49
Figura 17 - Fluxograma utilizado em algoritmos genéticos.....	52
Figura 18 – Tela inicial do software desenvolvido.	56
Figura 19 - Fluxograma do algoritmo genético desenvolvido.....	58
Figura 20 - Gráfico mostrando a variação da temperatura com a corrente drenada da célula.	59
Figura 21 - Célula a combustível de membrana polimérica.....	60
Figura 22 - Simulador de célula a combustível.	60
Figura 23 - Sinal de entrada gerado por ruído quantizado.	61
Figura 24 - Sinal de saída do simulador quando sujeito à entrada da figura 22.....	61
Figura 25 - Validação do modelo da equação 5.1.	62
Figura 26 - Validação do modelo da equação 5.2.	64
Figura 27 - Comparação entre os modelos da equação 5.1 e 5.2 em relação aos dados de validação.....	65

LISTA DE TABELAS

Tabela 1 – Valores dos Regressores recolhidos em experimentos aos regressores.....	26
Tabela 2 - Valores dos parâmetros dos regressores.....	30
Tabela 3 – Tabela mostrando valores para montagem do algoritmo da Roleta.	38
Tabela 4 – Representação dos indivíduos no algoritmo genético proposto.	53
Tabela 5 – Parâmetros de configuração do algoritmo genético.....	63
Tabela 6 – Resultados do algoritmo genético.....	63

SUMÁRIO

1 INTRODUÇÃO.....	10
1.1 Estado da arte.....	12
1.2. Estrutura do Trabalho	17
2 REVISÃO BIBLIOGRÁFICA	19
2.1 Identificação de Sistemas	19
2.1.1 Coleta dos dados e tratamento dos sinais	21
2.1.2 Seleção do modelo e determinação da estrutura.....	23
2.1.3 Critério para escolha dos regressores do modelo	24
2.1.4 Estimação dos parâmetros do modelo	31
2.1.5 Validação do modelo	33
2.2 ALGORITMOS GENÉTICOS.....	35
2.2.1 Representação de indivíduos em algoritmos genéticos	36
2.2.2 Inicialização da população e classificação	37
2.2.3 Seleção.....	37
2.2.4 Recombinação e mutação	38
2.2.5 Critérios de parada.....	40
3 ESCOLHA DO TEMPO DE AMOSTRAGEM EM IDENTIFICAÇÃO DE SISTEMAS NÃO LINEARES	42
3.1 Seleção da Estrutura e o tempo de amostragem	43
3.2 Estimação dos parâmetros do modelo e o tempo de amostragem	45
3.3 Determinação do tempo de amostragem por análise de correlação.....	46
4 METODOLOGIA.....	51
4.1 Algoritmo genético aplicado a determinação do tempo de amostragem.....	51
4.1.1 Representação do cromossomo	53
4.1.2 Função de avaliação.....	53
4.1.3 Seleção, recombinação e mutação	55
4.2 Software para determinação da taxa de decimação por meio de algoritmos genéticos..	55

4.3 Célula a combustível	58
5 RESULTADOS	62
6 Conclusão	66
REFERÊNCIAS	67
APÊNDICE A – Código para gerar o sinal de entrada que foi aplicado à célula a combustível.	71
APÊNDICE B – Termos candidatos para a identificação do modelo de temperatura da célula a combustível.....	72
APÊNDICE C- Software desenvolvido na plataforma MATLAB®.....	73

1 INTRODUÇÃO

A modelagem de sistemas dinâmicos tem sido de vital importância para o desenvolvimento de diversos campos, tais como: engenharia, economia, medicina e ecologia (CASSINI, 1999). Tal importância se justifica pelo fato de que a elaboração de um modelo para representar algum sistema pode ter diversas motivações, tais como: predição, simulação, controle, análise comportamental, otimização, detecção de falhas, e projeto de sensores virtuais que são capazes de substituir os sensores reais e fornecer o valor da variável medida (NELLES, 2001).

Qualquer sistema, considerado como uma relação de causas e efeitos pode ser representado matematicamente. Para que os modelos matemáticos possam desempenhar suas diversas funções como supracitado, é necessário que o modelo seja fiel ao sistema (LJUNG, 1999).

Um modelo matemático é composto por equações diferenciais (para sistemas contínuos) ou equações a diferença (para sistemas em tempo discreto) e descrevem a variação no tempo das variáveis de interesse do sistema (RODRIGUES, 1996).

O modelamento de sistemas pode ser classificado em três categorias: modelagem caixa branca, modelagem caixa cinza e modelagem caixa preta (AGUIRRE, 2007b).

Na modelagem caixa branca é necessário um amplo conhecimento do sistema, visto que é feito o levantamento e formulação de todos os fenômenos que afetam o comportamento do sistema. Tais formulações são feitas por meio das leis da física, química ou termodinâmica e consiste em uma tarefa complexa.

A modelagem caixa preta é utilizada quando se tem pouco ou nenhum conhecimento a respeito da física do processo. Este tipo de modelagem utiliza os sinais de entrada, que excitam o sistema, e suas respostas para obter um modelo que represente de forma satisfatória a dinâmica do mesmo. A modelagem caixa preta está inserida no contexto de identificação de sistemas.

Para modelar um sistema utilizando modelagem caixa preta são utilizados apenas dados de entrada, isto é, os sinais que excitam o sistema, e sinais de saída, isto é, a resposta do sistema de interesse. Com os dados coletados, tanto a estrutura do modelo quanto os parâmetros são determinados. Um problema da modelagem caixa preta é que os parâmetros determinados podem não ter nenhuma relação direta com a física do processo (NELLES, 2001).

Já a modelagem caixa cinza é um processo intermediário entre a modelagem caixa branca e modelagem caixa preta, onde conhecimentos prévios do sistema são utilizados jun-

tamente com dados experimentais para se obter um modelo que represente a dinâmica do processo (JØRGENSEN e HANGOS, 1995).

Os modelos matemáticos que tentam explicar o comportamento dos sistemas são na verdade uma aproximação, podendo representar apenas alguns aspectos, a representação de todos os aspectos do sistema é dificilmente alcançada. Desta forma, deve-se adotar um compromisso entre a capacidade de representação de um sistema e a sua complexidade (CORREA, 1997).

Um modelo é apenas uma representação aproximada do sistema. Por isso, não há como representar todas as características do mesmo, compreendendo, sim, algumas características. Os modelos são desenvolvidos de forma aproximada e muitas simplificações são consideradas, ao final do processo de identificação é feita uma validação e, se o modelo for considerado inadequado, é feita uma reavaliação (AGUIRRE, 2007a). Ljung (LJUNG, 1999) explica bem esse conceito:

Um sistema real é um objeto de um tipo diferente dos nossos modelos matemáticos. De certa maneira há uma impenetrável, porém transparente, tela entre nosso mundo de descrições matemáticas e o mundo real. Nós podemos olhar através dessa janela e comparar certos aspectos do sistema físico com sua descrição matemática, mas nós nunca podemos estabelecer uma conexão exata entre eles. A questão de susceptibilidade natural para descrição matemática possui aspectos filosóficos profundos, e em termos práticos nós temos que ter uma visão mais pragmática dos modelos. Nossa aceitação dos modelos deve ser mais guiada pela “utilidade” do que pela “verdade”. Entretanto, nós podemos ocasionalmente usar um conceito de “sistema verdadeiro”, definido em termos de um modelo matemático. Essa ficção é útil para comparar métodos de identificação e entender suas propriedades. Em certos contextos nós assumimos que os dados observados foram gerados de acordo com regras matemáticas bem-definidas, o que com certeza é uma idealização (LJUNG, 1999), p. 6, tradução nossa).

A estrutura mais simples para modelos é a linear, apresenta boa aplicabilidade quando é possível considerar uma faixa de operação restrita a uma determinada região, e é de fácil obtenção, de forma que deve ser sempre considerado antes de se pensar na aplicação de uma estrutura não linear (NELLES, 2001). Quando o modelo linear não consegue representar satisfatoriamente o comportamento do sistema, então se deve considerar uma representação não linear.

No mundo real, a maioria dos sistemas de interesse apresentam não linearidades, de forma que uma estrutura linear não é suficiente para representar o comportamento de tais sistemas, neste caso deve-se optar por uma estrutura não linear (AGUIRRE, 2007a).

A representação escolhida neste trabalho foi a representação NARMAX (Nonlinear AutoRegressive Moving Average with eXogenous input). Esta representação tem como vantagens uma alta capacidade de representação de não linearidades, fácil simulação, linearidade nos parâmetros, possibilidades de obter modelos estáticos e a facilidade de se combinar a seleção da estrutura com a estimação dos parâmetros (MORAIS, 2013).

1.1 Amostragem de sinais e decimação

Ao se recolher o sinal que será utilizado no processo de identificação de sistemas, muitas vezes este sinal não se encontra de forma adequada para ser utilizado nos algoritmos de identificação de sistemas (LJUNG, 1999). Estes sinais podem conter distúrbios em frequências acima daquelas que são de interesse do sistema a ser identificado, bem como falta de dados ou ainda muito mais amostras do que o necessário. Assim um pré-processamento deve ser aplicado aos sinais a fim de que estes se tornem adequados aos algoritmos de identificação.

Nos processos de identificação de sistemas usualmente é necessário obter a partir de um sinal contínuo no tempo, um sinal discreto no tempo, o qual é composto por amostras igualmente espaçadas no tempo do sinal contínuo. A esse procedimento dá-se o nome de amostragem do sinal.

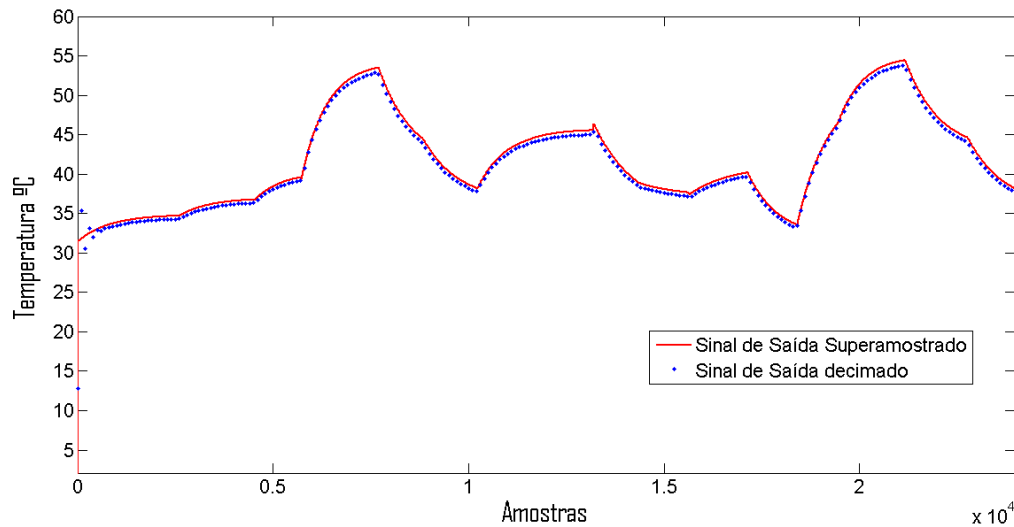
Na maioria das aplicações a taxa de amostragem dos sinais de entrada e saída são definidos pelo hardware, e quando a taxa de amostragem é muito maior do que a necessária para se realizar a identificação do sistema é utilizada a técnica de decimação.

A decimação consiste em a partir de um sinal amostrado em uma frequência determinada, amostrar o mesmo em uma frequência menor do que a original.

Assim um sinal que tivesse 1000 amostras, e fosse decimado a uma taxa de 10, teria ao fim apenas 100 amostras, isto é, o novo sinal decimado seria um subconjunto de 100 amostras do sinal original de 1000 amostras.

Um sinal superamostrado pode ser definido como um sinal amostrado em uma alta frequência, e é este o sinal que usualmente é decimado a fim de se obter um sinal com uma taxa de amostragem menor do que a original. A figura 1 apresenta o sinal de saída de um sistema superamostrado e o sinal obtido a partir da decimação do sinal de saída superamostrado.

Figura 1 – Sinal Superamostrado e sinal decimado.



Fonte: Elaboração própria.

Nos processos de identificação de sistemas técnicas de filtragens são também utilizadas para garantir uma melhor adequação dos sinais, contudo quando se trata de técnicas de identificação não lineares o uso de filtro se torna extremamente restrito, devido ao fato de que sinais não lineares apresentam comportamentos dinâmicos que se alteram ao serem processados por um filtro (LJUNG, 1999). Assim, para o tratamento dos sinais proposto nesse trabalho será considerado apenas a decimação dos sinais de entrada e saída.

O uso de um sinal superamostrado na identificação de sistemas não lineares utilizando representação NARMAX não permite a obtenção de bons modelos, isto ocorre principalmente devido ao alto nível de correlação dos sinais superamostrados, fazendo com que as matrizes utilizadas na detecção da estrutura e estimação dos parâmetros se tornem mal condicionadas devido à dependência linear imposta em suas colunas pelo alto nível de correlação entre as amostras. Outros problemas surgem da superamostragem dos sinais e são apresentados no capítulo 3.

Desta forma o primeiro desafio que o processo de identificação de sistemas não lineares enfrenta é a seleção adequada do tempo de amostragem dos sinais de entrada e saída.

A próxima sessão aborda o estado da arte apresentando pesquisas e pontos que foram propostos para resolver o problema da seleção do tempo de amostragem dos sinais para uso em identificação de sistemas.

1.2 Estado da arte

O estudo da correta taxa de amostragem dos sinais de entrada e saída para identificação recebem diversas perspectivas na literatura, seja simplesmente garantir que a identificação do sistema seja possível, ou visando minimizar o erro quadrático, ou ainda garantir uma maior representatividade do modelo identificado. Nesse contexto podem-se citar alguns trabalhos.

Em 1985 Sinha e Puthenpura, (SINHA e PUTHENPURA, 1985) propuseram um método para seleção da taxa de amostragem aplicada na identificação de sistemas contínuos. Nesta abordagem os autores apresentam uma transformação bilinear entre os planos z e w , e é proposto um critério baseado na posição dos autovalores no plano w . O método permitiu o aumento da precisão e da estabilidade no sistema identificado, contudo o método proposto só é aplicável em sistemas lineares.

(XIN *et al.*, 1995) apresenta um método para se estimar a taxa de decimação utilizando uma minimização do erro quadrático médio. É feito a estimação da resposta ao impulso e então a resposta estimada é interpolada a fim de recuperar o sinal original. É utilizado o estimador de mínimos quadrados e o erro médio quadrático é expresso em função da taxa de decimação e interpolação. Esta abordagem permite a minimização do erro quadrático por meio da adequada seleção da taxa de decimação, contudo a técnica foi aplicada apenas a resposta ao sinal de impulso, sinal este que não atende os requisitos de variabilidade do sinal de entrada necessário para se realizar a identificação de processos não lineares.

(HORÁČEK *et al.*, 2005) mostrou que o uso de dados decimados na identificação de processos caixa cinza pode levar a erros no modelo e até a modelos sem interpretação física. É apresentada como possível explicação para os erros a mudança de fase gerada pela decimação dos sinais.

Como citado por Aguirre e Billings (BILLINGS e AGUIRRE, 1995b) a grande maioria dos artigos disponíveis na literatura trata do processo de seleção dos dados de entrada e saída na identificação de sistemas lineares enquanto pouca atenção tem sido dada para os efeitos causados pela escolha dos sinais de entrada no processo de identificação não linear.

No mesmo artigo supracitado, Aguirre e Billings apresentam os efeitos da amostragem no processo de identificação de sistemas não lineares, tais como a influência negativa que uma taxa de amostragem alta pode ocasionar na seleção da estrutura, e também os efeitos negativos que uma baixa taxa de amostragem pode ocasionar na estimação dos parâmetros. É apresentado também um método para a seleção da taxa de amostragem utilizando uma análise

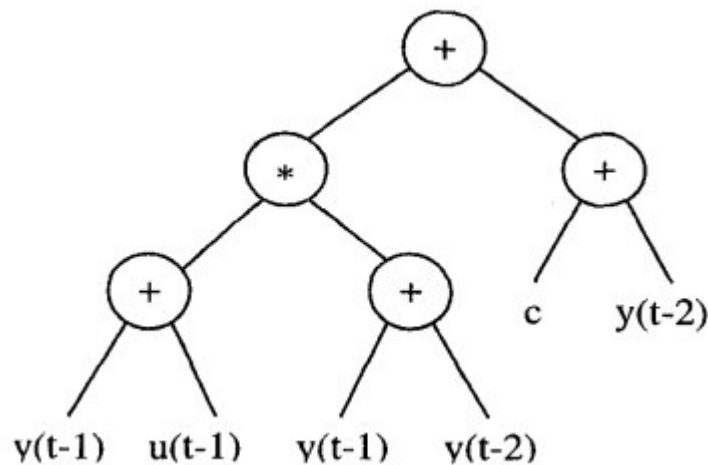
de correlação do sinal de saída que fornece uma estimativa da taxa de decimação a ser utilizada em um sinal superamostrado.

Neste sentido, o presente trabalho tem por objetivo contribuir com o processo de escolha da taxa de amostragem para identificação de sistemas não lineares, e para isso fará uso de algoritmos genéticos.

Diversos trabalhos abordam o uso de algoritmos genéticos e representação NARMAX no processo de identificação de sistemas não lineares, sempre com o objetivo de melhorar a representatividade do sistema, bem como minimizar o esforço computacional.

Em 1997 (RODRIGUEZ-VAZQUEZ e FLEMING, 1997) apresentaram uma solução utilizando algoritmos evolutivos baseado em árvore, para selecionar a estrutura de um modelo não linear com representação NARMAX. A estrutura era dada por uma expressão baseada em árvore, que era evoluída pelo próprio algoritmo genético. A estrutura da solução proposta pode ser vista na figura 2.

Figura 2 - Estrutura baseada em árvore para modelo polinomial não linear.



Fonte: Rodriguez-Vazquez e Fleming (1997).


Com base nestas expressões, uma estrutura era validada determinando os parâmetros do modelo, e em seguida comparada com o modelo real. O critério da informação de Akaike foi utilizado para garantir um compromisso entre representatividade e complexidade do modelo.

Ainda utilizando representação baseada em árvore, em (PU *et al.*, 2006) é proposto um algoritmo genético juntamente com uma abordagem multi-objetivo, onde a complexidade e desempenho dos modelos de sistemas caóticos puderam ser avaliados.

Em Camasca *et. al.* (2006) algoritmos genéticos foram também aplicados no processo de seleção da estrutura de modelos NARMAX. O cromossomo foi estruturado como na figura 3, onde cada gene representava um possível regressor a ser utilizado na estrutura, e o valor de 0 ou 1, indicava a ausência ou presença de um determinado termo.

Figura 3 - Estrutura de cromossomo utilizando codificação binária.

DC Term	$y(k-1)$	$y(k-2)$	$u(k-1)$	$y(k-1)^2$	$y(k-2) \times y(k-1)$	$u(k-1) \times y(k-1)$	$y(k-2)^2$	$u(k-1) \times y(k-2)$	$u(k-1)^2$
0	1	1	1	0	0	1	0	0	1


 $Termos = [y(k-1), y(k-2), u(k-1), u(k-1)y(k-1), u(k-1)^2]$

Fonte: Camasca *et al.*, 2006.

Uma penalidade na função do erro médio quadrático garantiu que termos espúrios fossem eliminados e as simulações mostraram que os algoritmos genéticos podem ser uma alternativa para a seleção da estrutura de modelos NARMAX.

Já em Bai *et. al.* (2007) observa-se uma nova abordagem de programação evolutiva denominada programação de expressão de gene (*Gene expression Programming – GEP*). Nesta abordagem o gene em um cromossomo é usado para representar um regressor e então uma estrutura NARMAX é representada por um cromossomo multigene.

O tamanho dos cromossomos não é mais fixo, mas permanece entre 1 e o número máximo de genes. Essa abordagem permite que estruturas de diferentes tamanhos tenham a chance de serem avaliadas.

Uma das vantagens dessa abordagem é que não foi necessário especificar o atraso máximo dos sinais de entrada e saída. A função *fitness* considerava a capacidade de representação do modelo bem como sua complexidade, o que permitiu que o modelo obtido fosse simples e adequado ao sistema proposto.

Diferentemente dos trabalhos já citados, Mohamed Vall e M'Hiri (2008) apresentam um algoritmo genético para determinar o sinal de entrada para ser utilizado na identificação de um sistema não linear em malha fechada, que utiliza controle de estrutura variável.

O algoritmo genético se concentra em encontrar uma função de chaveamento que tenha suas propriedades próximas a de um ruído branco e então esta função foi utilizada para gerar o sinal de excitação do sistema.

As simulações mostraram que foi possível identificar o sistema em malha fechada utilizando uma representação NARMAX.

Um algoritmo genético com competição paralela e competição justa foi desenvolvido em Yuan e Bai (2010). A população original foi dividida em várias faixas de acordo com o *fitness* de cada indivíduo, de forma que indivíduos com baixo *fitness* competem apenas entre si, e não com indivíduos com alto *fitness*. Tal método apresentou um bom desempenho em evitar a convergência prematura da população.

Os cromossomos foram representados baseados em árvore e este método se mostrou capaz de identificar os sistemas propostos, selecionando a estrutura e estimando os parâmetros.

Com base nos artigos mencionados, observa-se que os algoritmos genéticos tem sido amplamente utilizados para selecionar a estrutura dos modelos NARMAX.

O presente trabalho traz uma abordagem diferente. Neste trabalho os algoritmos genéticos são aplicados na escolha da taxa de amostragem dos sinais de entrada e saída que são utilizados no processo de identificação de sistemas não lineares, utilizando representação NARMAX.

Os objetivos deste trabalho são os seguintes:

1. Aplicação de uma nova metodologia para obtenção da taxa de amostragem dos sinais de entrada e saída utilizados no processo de identificação de sistemas NARMAX.
2. Obtenção de um modelo de temperatura não linear de uma célula a combustível de membrana polimérica.

1.2. Estrutura do Trabalho

O presente trabalho se organiza em seis partes, Capítulo 1 traz uma introdução ao assunto de modelamento e identificação de sistemas e a estrutura organizacional.

No Capítulo 2 é apresentada uma revisão bibliográfica dos principais assuntos que foram utilizados como base para desenvolvimento do presente trabalho, sendo revisados conceitos importantes sobre identificação de sistemas e algoritmos genéticos.

O problema da escolha do tempo de amostragem é apresentado no Capítulo 3 mostrando suas influências no processo de identificação de sistemas e também apresentando um método para seleção da taxa de amostragem utilizando análise de correlação.

O Capítulo 4 traz a metodologia utilizada para desenvolver a solução proposta, bem como uma descrição da célula a combustível, sistema o qual foi utilizado para validação da proposta. Os resultados da solução são apresentados no Capítulo 5.

O trabalho é finalizado no Capítulo 6 com conclusões sobre os resultados alcançados e sugestão para trabalhos futuros.

2 REVISÃO BIBLIOGRÁFICA

Este capítulo tem como objetivo introduzir os assuntos que serviram como base para o desenvolvimento do trabalho, trazendo conceitos importantes sobre a área de identificação de sistemas e algoritmos genéticos.

2.1 Identificação de Sistemas

A identificação de sistemas estuda técnicas alternativas de modelagem matemática. O modelo é obtido com a medição das entradas e saídas do processo, por isso pouco ou nenhum conhecimento prévio do sistema é necessário. São técnicas aplicáveis a qualquer sistema e são relativamente rápidas (LJUNG, 1999; NELLES, 2001; AGUIRRE, 2007a)

As principais etapas no processo de identificação são (LJUNG, 1999):

1. **Testes dinâmicos e coleta de dados:** dados são necessários para gerar o modelo e muitas vezes não estão disponíveis, sendo alguns testes necessários para extrair informação dinâmica do sistema. Problemas comuns nesta etapa ocorrem na escolha dos sinais de excitação, na execução do teste e na escolha do tempo de amostragem, sendo esse último o ponto de interesse desse trabalho. Posteriormente será mostrada a influência do tempo de amostragem no processo de identificação de sistemas.
2. **Representação matemática:** escolha do tipo de modelo a ser utilizado, sua estrutura, estimação dos parâmetros, etc. Essas escolhas são feitas com base nas características do sistema e no conhecimento e experiência de quem projeta o modelo. No presente trabalho a representação escolhida foi a NARMAX.
3. **Validação do modelo:** com o modelo obtido, ou o conjunto deles, é feita uma validação para verificar se o(s) modelo(s) incorpora(m) as características desejadas. A obtenção de modelos diferentes é útil para comparação, visto que algum pode ter desempenho melhor, em alguns aspectos, que outros.

O processo de identificação de sistemas é um processo iterativo, isto é, ao se chegar a um modelo final, deve-se testar se o mesmo atende aos critérios requeridos, se apresenta uma boa capacidade de representação quando comparado à sua complexidade e, principalmente, deve-se compará-lo a um conjunto de dados diferentes dos dados de identificação. Caso o

modelo obtido não se adeque, o processo deve ser reiniciado, revisando detalhes como tipo de representação, estrutura do modelo, algoritmos de estimação dos parâmetros, etc.

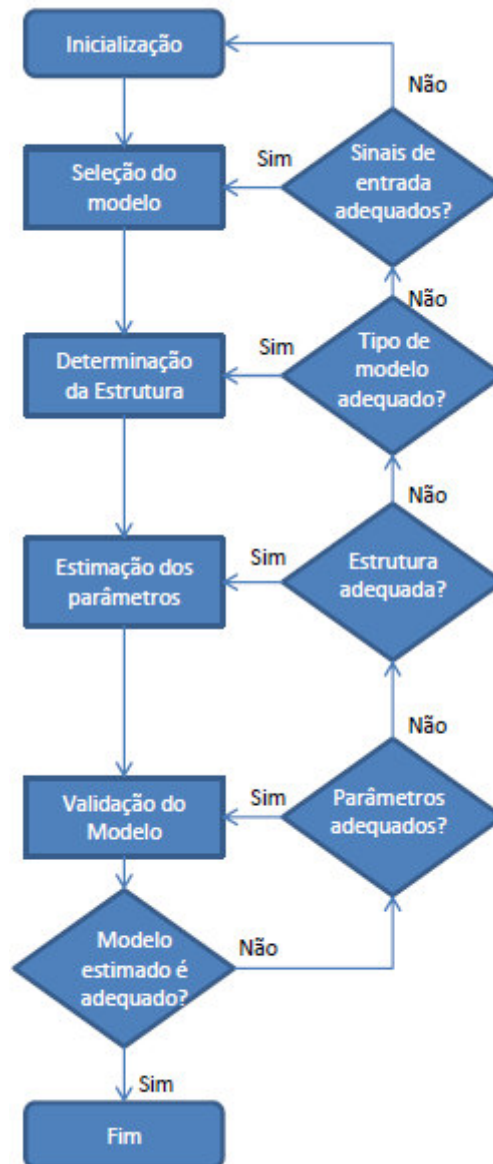
Os possíveis motivos que podem levar um modelo a ser considerado inadequado, na terceira etapa, são (LJUNG, 1999):

- O procedimento numérico falho em achar o melhor modelo para o critério estabelecido;
- O critério não foi bem estabelecido;
- Escolha inapropriada de modelos, pois nenhum foi considerado satisfatório;
- Os dados não foram informativos o suficiente para a seleção dos modelos.

A maior parte da aplicação da identificação consiste, de fato, em encontrar esses problemas. Isso é feito de forma iterativa, guiada pela informação prévia e os problemas encontrados (LJUNG, 1999).

O fluxograma na figura 4 mostra o processo de identificação de sistemas de forma generalizada.

Figura 4 - Fluxograma de identificação de sistemas



Fonte: Adaptado de (MORAIS, 2013).

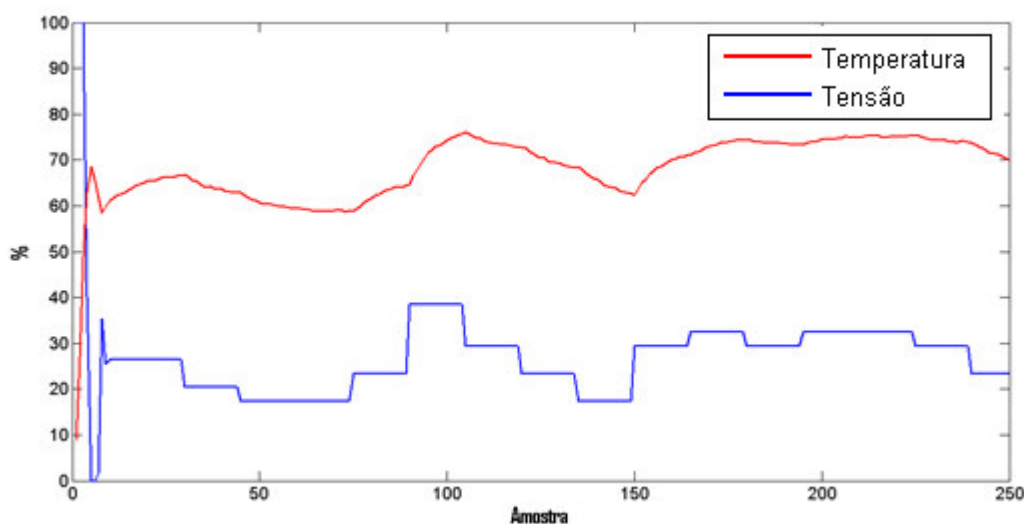
2.1.1 Coleta dos dados e tratamento dos sinais

No processo de identificação de sistemas caixa preta, o primeiro passo é a coleta dos dados que serão utilizados no processo. Os sinais de entrada que irão excitar o sistema e a resposta a essa excitação são coletados com o objetivo de serem utilizados no processo. A escolha desses sinais deve ser condicionada ao tipo de sistema, linear ou não linear. Em caso de sistemas lineares a escolha do sinal de entrada parece estar bem consolidada e pouca atenção tem sido dada a este assunto (BILLINGS e AGUIRRE, 1995b). Para o caso não linear foi demonstrado

(LEONTARITIS e BILLINGS, 1987) que o uso do sinal binário pseudo aleatório o (*PRBS*) como entrada pode causar perda de informações relevantes para o processo de identificação.

Para o caso de sistemas não lineares, a frequência e a amplitude dos sinais de entrada são de grande importância (AGUIRRE e BILLINGS, 1995a). O sinal mais comumente utilizado na identificação de processos não lineares é o “ruído quantizado” (RODRIGUES, 1996). Neste tipo de sinal, um algoritmo sorteia um valor dentro de uma faixa de interesse e mantém esse nível por um tempo pré-determinado, ao fim deste intervalo um novo valor é sorteado e novamente é mantido pelo mesmo período de tempo. Esse sinal tem sido muito utilizado por conseguir excitar a planta em uma maior faixa de frequência. A figura 5 apresenta o sinal de resposta de um forno elétrico a um sinal de entrada utilizando “ruído quantizado”.

Figura 5 - Resposta de um forno elétrico ao ruído quantizado



Fonte: Adaptado de Rodrigues (1996).

Embora o ruído quantizado apresente boa aplicabilidade, no presente trabalho foi utilizado um sinal de entrada semelhante ao ruído quantizado, contudo com o tempo em que o sinal permanece em determinado nível sendo variável de forma aleatória. Tal abordagem foi feita para melhorar o condicionamento das matrizes dos regressores, tendo em vista que quanto mais um sinal permanece em um determinado nível maior é a chance da matriz dos regressores possuir colunas linearmente dependentes e mais mal condicionada será a inversa da matriz dos regressores (AGUIRRE, 2007a). O algoritmo para o sinal de entrada utilizado nesse trabalho pode ser visto no apêndice A.

Outro ponto importante no processo de coleta é a seleção da taxa de amostragem, contudo tal assunto será tratado no capítulo 3, por se tratar de ponto fundamental na pesquisa desenvolvida nesse trabalho.

2.1.2 Seleção do modelo e determinação da estrutura

No presente trabalho optou-se por utilizar uma estrutura "não-linear auto-regressivo com média móvel e entrada exógena" NARMAX (*nonlinear autoregressive moving average with exogenous inputs*). Esta representação foi escolhida devido à simplicidade e o número reduzido de parâmetros a se determinar, quando comparado com representações como as séries de Wiener e de Volterra, o que resulta em um menor esforço computacional (LEONTARITIS e BILLINGS, 1984).

Um modelo polinomial NARMAX pode ser representado por

$$y(k) = F^l[y(k-1), \dots, y(k-n_y), u(k-\tau_d), \dots, u(k-n_u), e(k-1), \dots, e(k-n_e)] + e(k) \quad (2.1)$$

Sendo $e(k)$ os efeitos que não podem ser bem representados por F^l .

Matematicamente essa função pode ser expandida por:

$$y(k) = \sum_{m=0}^l \sum_{p=0}^m \sum_{n_1, n_m}^{n_y, n_u} C_{p, m-p}(n_1, \dots, n_m) \prod_{i=1}^p y(k-n_i) \prod_{i=p+1}^m u(k-n_i) \quad (2.2)$$

Sendo que

$$\sum_{n_1, n_m}^{n_y, n_u} = \sum_{n_1=1}^{n_y} \dots \sum_{n_m=1}^{n_u}, \quad (2.3)$$

E o limite superior é n_y se o somatório se refere a fatores do tipo $y(k-n_i)$ ou n_u para fatores do tipo $u(k-n_i)$.

Na expansão de um modelo NARMAX é necessário descobrir quais termos serão candidatos a fazer parte da identificação, obviamente alguns critérios devem ser seguidos para selecionar dentre os termos candidatos, quais farão efetivamente parte do modelo final.

Deve-se atentar durante o processo de expansão da equação 2.2, que alguns termos podem vir a se repetir, assim um termo que seja multiplicado por uma constante $C_{0,3}$, poderia ser agrupado com outro termo idêntico, que fosse multiplicado por outra constante $C_{3,1}$, gerando um termo de constante $(C_{0,3} + C_{3,1})$.

Para a identificação da célula combustível, foi utilizando um modelo NARMAX com $l=3$ e $n_y = n_u = 3$. A expansão não será apresentada neste trabalho, mas foram encontrados 136 termos regressores e, após uma racionalização dos termos repetidos, os termos candidatos foram reduzidos para 84 termos que são apresentados no apêndice B.

A seguir será apresentado o critério utilizado para selecionar quais os regressores irão compor o modelo final da célula combustível.

2.1.3 Critério para escolha dos regressores do modelo

Na representação NARMAX a equação 2.2 pode gerar uma enorme quantidade de regressores candidatos a fazerem parte do modelo. De acordo com Korenberg *et al.* (1988), o número de termos n_θ pode ser determinado pelas equações 2.4, 2.5 e 2.6.

$$n_\theta = M + 1 \quad (2.4)$$

$$M = \sum_{i=1}^l n_i \quad (2.5)$$

$$n_i = \frac{n_{i-1}(n_y + n_u + n_e + i - 1)}{i}, \quad n_0 = 1 \quad (2.6)$$

Onde n_y é o número de atraso para y , n_u é o número de atrasos para u , n_e é o número de atrasos para os termos de resíduo e l é o grau de não linearidade.

O valor de n_θ pode facilmente se tornar muito grande. No caso da célula combustível, por exemplo, foi utilizado $l=3$ e $n_y = n_u = 3$ foram gerados 84 termos candidatos. Obviamente todos estes termos não podem ser utilizados no modelo, pois tornaria muito complexo o modelo final sem contar que a utilização de termos que não são necessários pode fazer com que o modelo apresente um pior desempenho (AGUIRRE, 2007a). Wei (2004) acrescenta ainda que muitos termos candidatos são redundantes e podem ser removidos do modelo.

A fim de se estabelecer um critério para seleção dos regressores a serem utilizados no modelo, Aguirre (2007a) propõe a utilização da taxa de redução de erro (ERR – do inglês Error reduction Rate), a qual foi também utilizada nesse trabalho e é apresentada a seguir.

Considere o modelo NARMAX representado pela equação 2.7.

$$y(k) = \Psi(k-1)\hat{\theta} + \xi(k) \quad (2.7)$$

Onde,

$$y = \begin{bmatrix} y(1) \\ y(2) \\ \vdots \\ y(N) \end{bmatrix}; \quad \Psi = [\psi_1 \quad \dots \quad \psi_{n_\theta}]; \quad \hat{\theta} = \begin{bmatrix} \hat{\theta}_1 \\ \hat{\theta}_2 \\ \vdots \\ \hat{\theta}_{n_\theta} \end{bmatrix}; \quad \xi = \begin{bmatrix} \xi(1) \\ \xi(2) \\ \vdots \\ \xi(N) \end{bmatrix} \quad (2.8)$$

e

$$\psi_i = \begin{bmatrix} \psi_i(1) \\ \psi_i(2) \\ \vdots \\ \psi_i(N) \end{bmatrix}, \quad i=1, 2, \dots, n_\theta. \quad (2.9)$$

Nas equações 2.7 e 2.8 $y(k)$ é a saída do modelo e é um vetor de N observações, Ψ representa uma matriz formada pelos regressores candidatos, isto é, ψ_i são termos do tipo $y(k-1)$, $y(k-2)$, $y(k-1)u(k-2)$, etc., e são também tomados N vezes. $\xi(k)$ é o erro, dado pela diferença entre o valor medido e o valor estimado pelo modelo.

A seguir é ilustrado o processo de montagem da matriz de regressores. Seja a matriz de regressores dada pela equação 2.10.

$$\Psi = [y(k-1) \quad y(k-2) \quad u(k-1) \quad y(k-1)u(k-1)] \quad (2.10)$$

Os valores dos regressores da equação 2.10 estão representados na tabela 1.

Tabela 1 – Valores dos Regressores recolhidos em experimentos aos regressores.

K	$y(k-1)$	$y(k-2)$	$u(k-1)$	$y(k-1)u(k-1)$
1	1	0	0,5	0,5
2	0,051347	1	-0,5	-0,0257
3	0,298743	0,051347	0,5	0,1494
4	-0,65815	0,298743	-0,5	0,3291
5	0,758556	-0,65815	-0,5	-0,3793
6	-0,69703	0,758556	-0,5	0,3485

Fonte: Elaboração própria.

De acordo com a tabela 1 nota-se que a matriz de regressores é montada a partir das observações de entradas e saídas medidas no sistema a ser considerado para identificação e os tempos anteriores ao tempo zero como sendo zero. Assim, a primeira medida de $y(k-2)$, ou seja, para $k=1$, $y(-1) = 0$, se houvesse um regressor $y(k-3)$ para $k=1$ e $k=2$, $y(-2)$ e $y(-1)$ seriam também zero.

Com a matriz de regressores montada é necessário então utilizar um critério para selecionar quais regressores irão compor o modelo.

A equação 2.7 pode ser escrita da seguinte forma,

$$y(k) = \sum_{i=1}^{n_\theta} \hat{\theta}_i \psi_i(k-1) + \xi(k) \quad (2.11)$$

A matriz de regressores composta pelos vetores $\psi_i(k-1)$ pode ser ortogonalizada utilizando algum algoritmo de ortogonalização tais como Gram-Schmidt, Gram-Schmidt modificado e transformações de Householder (BILLINGS, 2013). Com a ortogonalização na matriz de regressores o seguinte modelo auxiliar é obtido:

$$y(k) = \sum_{i=1}^{n_\theta} \hat{g}_i w_i(k-1) + \xi(k) \quad (2.12)$$

Na equação 2.12 $w_i(k-1)$ são vetores ortogonais sobre os dados obtidos a partir da ortogonalização de $\psi_i(k-1)$ e \hat{g}_i é um vetor de parâmetros que relaciona a matriz de regres-

sores ortogonalizada aos valores de $y(k)$ e por meio de uma transformação adequada pode-se chegar a $\hat{\theta}_l$ utilizando \hat{g}_l (AGUIRRE, 2007a).

A soma dos valores quadráticos de $y(k)$ é dado por,

$$y(k)^2 = \left(\sum_{i=1}^{n_\theta} \hat{g}_l w_i(k-1) + \xi(k) \right) \times \left(\sum_{j=1}^{n_\theta} \hat{g}_j w_j(k-1) + \xi(k) \right) \quad (2.13)$$

Para desenvolver o produto da equação 2.13, vale notar que,

$$\sum_{i=1}^3 a_i \times \sum_{i=1}^3 a_i = (a_1 + a_2 + a_3) \times (a_1 + a_2 + a_3). \quad (2.14)$$

$$(a_1 + a_2 + a_3) \times (a_1 + a_2 + a_3) = (a_1 a_1 + a_1 a_2 + a_1 a_3 + a_2 a_1 + a_2 a_2 + a_2 a_3 + a_3 a_1 + a_3 a_2 + a_3 a_3). \quad (2.15)$$

Tendo em vista que $a_1 a_2 = a_2 a_1$ resulta que a equação 2.14 pode ser escrita da seguinte forma,

$$\sum_{i=1}^3 a_i \times \sum_{i=1}^3 a_i = \sum_{i=1}^3 a_i^2 + \sum_{i=1, j=1, \forall i \neq j}^3 2a_i a_j. \quad (2.16)$$

Assim a equação 2.13 resulta em,

$$y(k)^2 = \sum_{i=1}^{n_\theta} \hat{g}_i^2 \langle w_i, w_i \rangle + \sum_{i=1, j=1, \forall i \neq j}^{n_\theta} 2\hat{g}_i \hat{g}_j \langle w_i, w_j \rangle + 2 \sum_{i=1}^{n_\theta} \hat{g}_i \langle w_i, \xi \rangle + \langle \xi, \xi \rangle \quad (2.17)$$

O vetor $w_i(k-1)$ é ortogonal sobre os dados, logo o produto interno $\langle w_i, w_j \rangle$, para $i \neq j$ é zero, assim o termo $\sum_{i=1, j=1, \forall i \neq j}^{n_\theta} 2\hat{g}_i \hat{g}_j \langle w_i, w_j \rangle = 0$, e tendo em vista que os vetores w_i e ξ são não correlacionados, isto é, uma variação em w_i não produz nenhuma variação em ξ , conclui-se que tais vetores são ortogonais, e também que o produto interno destes é zero (RODGERS *et al.*, 1984), assim a equação 2.17 se resume em,

$$y(k)^2 = \sum_{i=1}^{n_\theta} \hat{g}_i^2 \langle w_i, w_i \rangle + \langle \xi, \xi \rangle \quad (2.18)$$

A equação 2.18 mostra que a saída do sistema ao quadrado pode ser explicada pelo somatório dos parâmetros de cada regressor ao quadrado, multiplicado pelo produto interno dos regressores, e que a parcela não explicada pelos regressores é igual ao produto interno dos resíduos. Assim se considerar um modelo sem nenhum regressor, a parcela não explicada pelo modelo é o próprio produto interno dos resíduos.

A taxa de redução do erro (ERR) pode ser escrita da seguinte forma,

$$[ERR]_i = \frac{\hat{g}_i^2 \langle w_i, w_i \rangle}{\langle y, y \rangle} \quad (2.19)$$

A seguinte interpretação pode ser dada para a equação 2.19, quanto mais próximo de $\langle y, y \rangle$ estiver o produto de $\hat{g}_i^2 \langle w_i, w_i \rangle$ mais próximo da unidade estará a taxa de redução do erro, mostrando que os dados podem ser explicados pelo regressor i , de tal forma que este deve ser inserido no modelo.

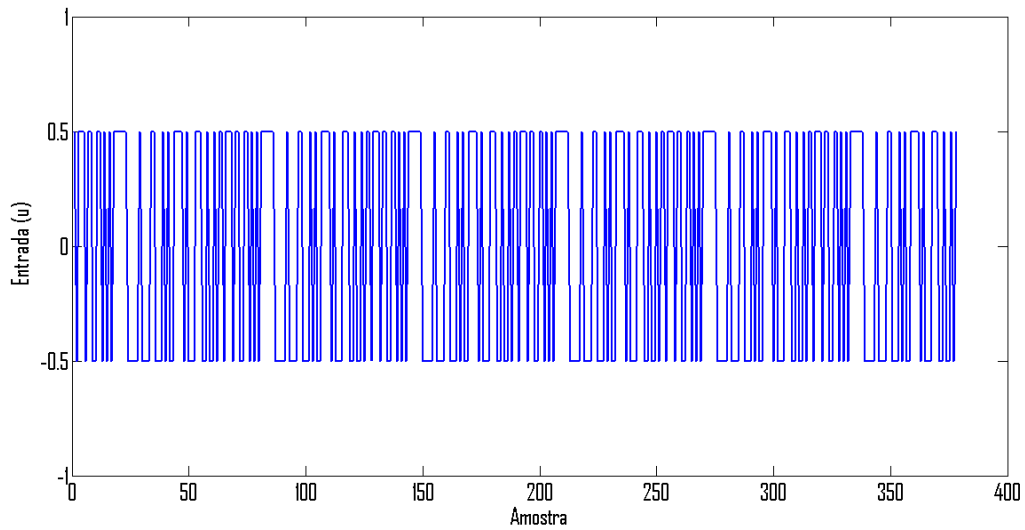
Para exemplificar, considere o sistema linear expresso pela equação 2.20,

$$y(k) = u(k-2) - 0,5 * y(k-1) - 0,2 * y(k-2) + e(k), \quad (2.20)$$

Onde, $y(k)$ é a saída do sistema, $u(k)$ é a entrada e $e(k)$ é um ruído branco de variância 0,01.

No sistema da equação 2.20, foi utilizado um sinal binário pseudo-aleatório, o qual se aproxima de um sinal aleatório com média zero (AVELAR, 2012) como entrada de excitação, representado na figura 6.

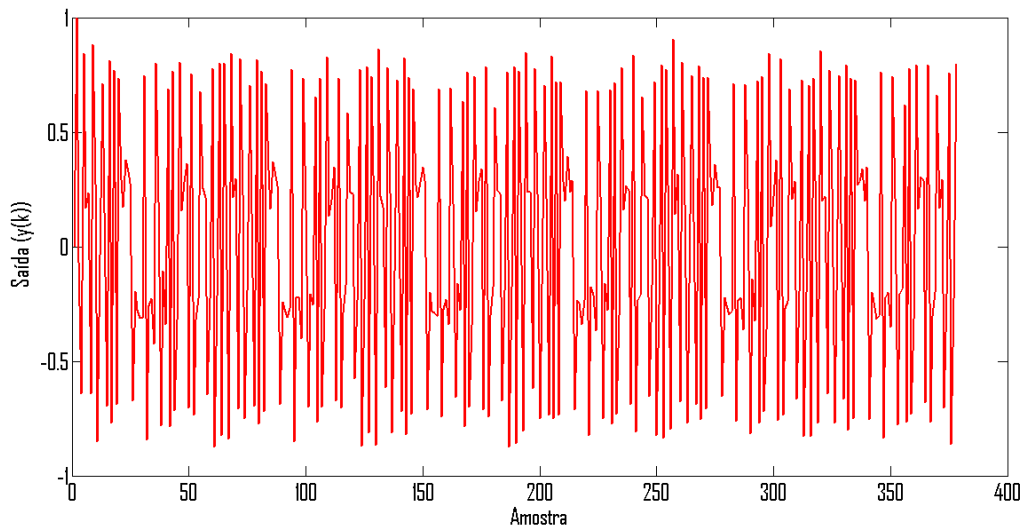
Figura 6 - Entrada de excitação do sistema ($u(k)$).



Fonte: Elaboração própria.

E a resposta ao sinal de entrada da figura 6 é representada na figura 7.

Figura 7 - Resposta do sistema da equação 2.20 ao sinal de entrada da figura 5.



Fonte: Elaboração própria.

Para a identificação desse sistema escolheu-se os parâmetros do sistema como sendo $l=3$ e $n_y = n_u = 2$ o que gerou um conjunto de 35 regressores.

A tabela 2 exhibe uma relação dos regressores e suas respectivas taxas de redução de erro. Como dito anteriormente, os regressores com maior ERR serão os escolhidos para compor o modelo final, os quais são exatamente os regressores apresentados na equação 2.20.

Tabela 2 - Valores dos parâmetros dos regressores.

Regressor	$\hat{\theta}_i$	ERR
$u(k-2)$	1,0006	0,7933
$y(k-1)$	0,50049	0,1716
$y(k-2)$	-0,19841	0,0320
$u(k-1)$	-0,00942	6,8613e-05
$y(k-1)^3$	-0,03017	3,0060e-05

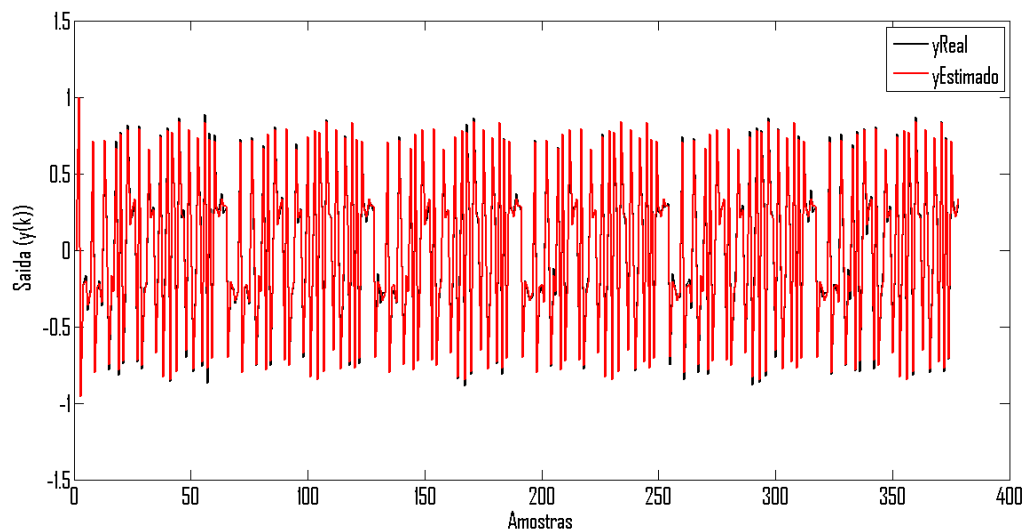
Fonte: Elaboração própria.

Observa-se na tabela 2 que os três regressores de maior ERR são exatamente os regressores do sistema da equação 2.20, enquanto que os outros regressores apresentam ERR muito baixo, de forma que não devem ser inseridos no modelo. O modelo identificado seria então,

$$y(k) = 1,0006u(k-2) - 0,50049 * y(k-1) - 0,19841 * y(k-2). \quad (2.21)$$

A figura 8 mostra a comparação do sistema da equação 2.20 com o modelo da equação 2.21.

Figura 8 - Comparação entre sistema da equação 2.20 e modelo da equação 2.21.



Fonte: Elaboração própria.

A determinação da taxa de redução do erro necessita de ser realizada juntamente com a estimação dos parâmetros, visto que os parâmetros são utilizados para se determinar o ERR

de cada regressor, assim o próximo assunto a ser discutido será a estimação dos parâmetros do modelo.

2.1.4 Estimação dos parâmetros do modelo

A estimação dos parâmetros θ do sistema na representação NARMAX da equação 2.22, é feita utilizando o estimador de mínimos quadrados (HELENE, 2006).

$$y = \Psi\theta + \xi \quad (2.22)$$

Sendo $\Psi \in \mathbb{R}^{n \times m}$ e $y \in \mathbb{R}^n$ conhecidos, $\theta \in \mathbb{R}^m$ o vetor a se determinar e $\xi \in \mathbb{R}^n$ o resíduo que representa a diferença entre y e $\Psi\theta$.

Considera-se a função custo quadrática por possuir uma solução global única, o custo funcional é definido como $J: \mathbb{R}^n \rightarrow \mathbb{R}$ dada por,

$$J(\theta) = \|\Psi\theta - y\|^2 = (\Psi\theta - y)^T(\Psi\theta - y) \quad (2.23)$$

O problema de minimização de $J(\theta)$ sem restrição pode ser escrito como,

$$\min_{\theta \in \mathbb{R}^n} J(\theta) \quad (2.24)$$

A solução ótima que minimiza 2.23 denotada agora por $\hat{\theta}$, é uma solução que satisfaz a relação 2.25,

$$\|\Psi\hat{\theta} - y\|^2 \leq \|\Psi\theta - y\|^2. \quad (2.25)$$

A relação 2.25 estabelece que para qualquer θ escolhido, o valor do resíduo ξ é mínimo para $\hat{\theta}$, e $\|\cdot\|^2$ representa o quadrado da norma euclidiana, sendo $\|a\|^2 = a^T a$.

Desenvolvendo $J(\theta)$ tem-se,

$$\begin{aligned} J(\theta) &= (\Psi\theta - y)^T(\Psi\theta - y) = (\theta^T \Psi^T - y^T)(\Psi\theta - y) \\ &= (\theta^T \Psi^T \Psi \theta - \theta^T \Psi^T y - y^T \Psi \theta + y^T y) \end{aligned} \quad (2.26)$$

Assim a equação 2.24 pode ser escrita da seguinte forma,

$$\hat{\theta} = \min_{\theta} (\theta^T \Psi^T \Psi \theta - \theta^T \Psi^T y - y^T \Psi \theta + y^T y) \quad (2.27)$$

Diferenciando e igualando a zero, a fim de obter o valor ótimo, tem-se,

$$\frac{\partial J}{\partial \theta} = (\Psi^T \Psi) \theta + (\Psi^T \Psi)^T \theta - \Psi^T y - \Psi^T y = 0 \quad (2.28)$$

$$(\Psi^T \Psi \theta + \Psi^T \Psi \theta - \Psi^T y - \Psi^T y) = 0 \quad (2.29)$$

$$2\Psi^T \Psi \theta - 2\Psi^T y = 0 \quad (2.30)$$

$$\Psi^T \Psi \theta = \Psi^T y \quad (2.31)$$

Se a matriz Ψ é não singular, isto é, possui determinante diferente de zero, o produto $\Psi^T \Psi$ é invertível e segue que,

$$(\Psi^T \Psi)^{-1} (\Psi^T \Psi) \theta = (\Psi^T \Psi)^{-1} \Psi^T y. \quad (2.32)$$

Logo a solução de $\hat{\theta}$ é dada por,

$$\hat{\theta} = (\Psi^T \Psi)^{-1} \Psi^T y. \quad (2.33)$$

Para o caso em que a matriz de regressores Ψ possui apenas valores conhecidos, tais como $y(k)$ e $u(k)$, a equação 2.33 é suficiente para estimar os parâmetros do modelo, já em casos em que o vetor de resíduos $\xi(k)$ é utilizado como parte do modelo e para números grandes de regressores, métodos mais avançados devem ser utilizados tais como, método de Gram-Schmidt clássico, método Gram-Schmidt modificado e método de Golub-Householder, estes métodos podem ser encontrados em Aguirre (2007a).

Após estimar os parâmetros do modelo, o próximo passo é validar o modelo encontrado, a fim de se observar se a dinâmica do mesmo é uma boa representação do sistema a se determinar, este assunto é abordado no próximo tópico.

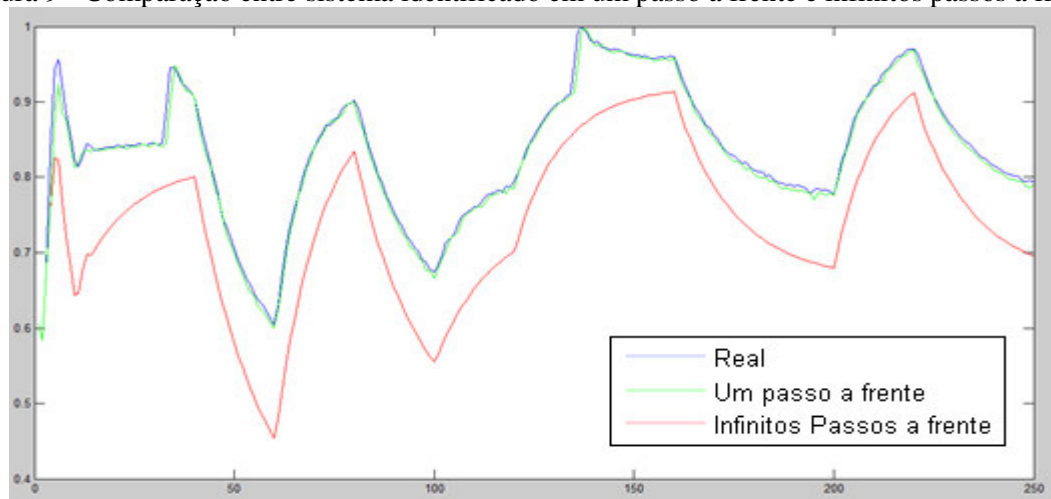
2.1.5 Validação do modelo

A validação de um modelo é o passo no qual é verificada a capacidade representativa do modelo obtido em relação ao sistema a ser identificado. Deve-se assumir um compromisso entre simplicidade do modelo e capacidade de representação, tendo em vista que raramente é possível obter um modelo que represente fielmente o sistema a ser identificado (NELLES, 2001).

Durante o processo de validação, o conjunto de dados a ser utilizado na validação deve ser diferente do conjunto de dados utilizado no processo de identificação, a motivação para isso é que se deseja encontrar um modelo geral, que consiga explicar outros dados provenientes do mesmo sistema, que não os usados na identificação.

Existem diversas formas utilizadas para simular um sistema. Uma forma comum, mas que não fornece bons resultados para o processo de identificação é a predição um passo a frente, neste tipo de simulação o valor estimado é sempre identificado utilizando os dados medidos a partir do sistema em um passo anterior, fazendo com que os dados estimados sejam muito próximos dos valores medidos. Esse tipo de simulação pode levar a enganos e fazer com que se pense que o modelo é melhor do que realmente é. A figura 9 apresenta uma simulação comparando dados reais, com simulação um passo a frente e simulação com infinitos passos a frente.

Figura 9 - Comparação entre sistema identificado em um passo a frente e infinitos passos a frente.



Fonte: Elaboração própria.

A simulação um passo a frente utiliza todas as N medidas do sistema para estimar a medida $N+1$, o que acaba por fazer com que o modelo apresente um ótimo desempenho. Contudo, neste tipo de predição os erros das estimativas não são levados em conta para estimar o próximo passo, e isso faz com que o modelo pareça ser melhor do que realmente é.

Na simulação infinitos passos à frente, são utilizados apenas a medida k , para estimar a medida $k+1$, de forma que, de posse de uma condição inicial é possível estimar o comportamento do sistema sujeito a uma determinada entrada.

Outra forma de avaliar a validade de um modelo não-linear é utilizando os conceitos estatísticos de correlação. A ideia aqui é garantir que os resíduos sejam imprevisíveis, isto é, garantir que estes não sejam nenhuma combinação linear ou não linear, das entradas, saídas ou resíduos passados (BILLINGS e ZHU, 1994).

Os resíduos consistem na parte dos dados de identificação que o modelo não conseguiu explicar, assim se os resíduos forem completamente aleatórios, isto é, se os resíduos forem um ruído branco, o modelo terá explicado tudo que havia para explicar e o que não é explicado é devido a erros aleatórios. Em contrapartida, caso os resíduos sejam um ruído colorido, isto é um indício de que o modelo não conseguiu explicar toda dinâmica contida nos dados.

Uma forma de analisar se os resíduos são um ruído branco é por análise de correlação cruzada. Contudo para sistemas não lineares, é de se esperar que haja correlações não lineares, assim é definido um conjunto de testes em Billings (2013), que devem ser satisfeitos para validar o modelo, tais condições são expressas pelas equações 2.34 a 2.38.

$$\theta_{\xi\xi}(\tau) = E[\xi(k - \tau)\xi(k)] = \delta(\tau), \quad (2.34)$$

$$\theta_{u\xi}(\tau) = E[u(k - \tau)\xi(k)] = 0, \forall \tau, \quad (2.35)$$

$$\theta_{u^2\xi}(\tau) = E[(u^2(k - \tau) - \overline{u^2(k)})\xi(k)] = 0, \forall \tau, \quad (2.36)$$

$$\theta_{u^2\xi^2}(\tau) = E[(u^2(k - \tau) - \overline{u^2(k)})\xi^2(k)] = 0, \forall \tau, \quad (2.37)$$

$$\theta_{\xi(\xi u)}(\tau) = E[(\xi(k)\xi(k - 1 - \tau)u(k - 1 - \tau))] = 0, \tau \geq 0. \quad (2.38)$$

Sendo $\delta(\tau)$ a função impulso unitário e $E[*]$ a esperança matemática.

Os passos do processo de identificação de sistemas apresentados neste capítulo constituem o processo básico de identificação e, quando seguidos adequadamente, permitem a obtenção de modelos capazes de descrever as características dinâmicas de alguns sistemas, isto é, permite identificar o que é identificável.

Um ponto importante no processo de identificação de sistemas que não é abordado neste capítulo de forma proposital se trata da influência dos tempos de amostragem de um sistema. Como será visto no capítulo 3 o tempo de amostragem dos sinais de entrada e saída tem uma influência muito grande no modelo a ser identificado e não foi tratado aqui por ser ponto primordial no desenvolvimento deste trabalho.

O próximo item aborda os algoritmos genéticos, assunto base para o desenvolvimento do presente trabalho.

2.2 ALGORITMOS GENÉTICOS

Alguns problemas podem apresentar soluções demasiado complexas devido a grande quantidade de variáveis envolvidas ou por apresentar um espaço de solução muito grande. Os algoritmos genéticos (AG) surgiram com o objetivo de atacar tais problemas, encontrando boas soluções ou simplesmente reduzindo o espaço de solução (GOLDBERG *et al.*, 1989b).

No início dos anos 70 John Henry Holland, com base nos postulados de seleção natural descritos por Charles Darwin, desenvolveu algoritmos computacionais capazes de manipular cadeias de genes de forma a construir organismos complexos e melhores adaptados para resolver problemas complexos. Tais algoritmos deram origem à teoria de algoritmos genéticos (PETROLI, 2011).

Os algoritmos genéticos são utilizados para resolver uma enorme gama de problemas tais como *design* de redes de comunicação, otimização de busca em banco de dados, controle para sistemas físicos, identificação de sistemas, problemas de logística dentre muitos outros (COELHO e COELHO, 1999).

A busca de uma solução utilizando algoritmos genéticos parte assumindo que cada indivíduo é uma possível solução para um problema específico. Desta forma os algoritmos genéticos permitem a busca de uma solução de forma paralelizada, isto é, a busca por uma solução é feita a partir de diversos pontos do espaço de solução. Partindo de diversos indivíduos os algoritmos genéticos simulam a evolução natural introduzida por Charles Darwin, os indivíduos que apresentam um melhor desempenho são escolhidos para se reproduzirem criando novos indivíduos e os que apresentam desempenho insatisfatório são descartados.

De acordo com (GOLDBERG, 1994) os algoritmos genéticos se caracterizam por:

- Operar em uma população de pontos;
- Não requerem derivadas;

- Trabalhar com a codificação de seu conjunto de parâmetros, não com os próprios parâmetros (representação binária);
- Realizar transições probabilísticas, não regras determinísticas;
- Necessitar apenas de informação sobre o valor de uma função objetivo para cada integrante da população de indivíduos.

As próximas sessões descrevem os passos necessários para a implantação de um algoritmo genético.

2.2.1 Representação de indivíduos em algoritmos genéticos

O processo de inicialização da população começa com uma escolha apropriada de qual representação deve ser utilizada. Existem basicamente dois tipos de representação, a canônica ou binária e a representação real.

Na representação canônica os indivíduos são representados por vetores binários por meio de algoritmos que permitem codificá-los e decodificá-los em sequências de bits.

Dado um número de bits n é possível representar 2^n valores diferentes, de forma que à medida que se aumenta o intervalo, aumenta-se também a granulometria, isto é, a diferença entre valores consecutivos.

A granulometria de um intervalo $[x_{min}, x_{max}]$ é dada pela equação 2.39.

$$Granulometria = \frac{x_{max} - x_{min}}{2^n} \quad (2.39)$$

Quanto maior a granulometria pior é a representação do intervalo em questão na forma canônica, tendo em vista que quanto maior a diferença entre dois valores consecutivos, maior a chance do ponto ótimo não estar representado pela codificação selecionada. Assim faz-se necessário aumentar o número de bits n sempre que aumentar o intervalo a ser avaliado.

A representação binária é mais adequada para tratamento de problemas em que os indivíduos, ou possíveis soluções, apresentam valores discretos (JANIKOW e MICHALEWICZ, 1990)

Na representação real os indivíduos são representados diretamente por números reais, por não ter sido utilizada neste trabalho não serão detalhadas, mas informações adicionais podem ser encontradas em Radcliffe (1991) e Wright (1991).

Como as taxas de decimação são valores inteiros, o tipo de representação utilizada no presente trabalho foi a representação binária.

2.2.2 Inicialização da população e classificação

Feita a escolha de qual representação será utilizada, a população é então criada, isto é, são gerados diversos indivíduos pertencentes ao espaço de solução. Cada um dos indivíduos pertencentes à população inicial representa uma possível solução para o problema em questão.

A população inicial deve apresentar uma grande diversidade em questão dos indivíduos para que o algoritmo genético possa explorar a região factível do problema em sua totalidade, caso contrário o algoritmo pode converter para mínimos locais dando uma falsa ideia de otimalidade (SIVANANDAM e DEEPA, 2008).

Para gerar a população inicial se utiliza normalmente procedimentos aleatórios e até mesmo algoritmos heurísticos (MALAQUIAS, 2006).

O próximo passo é então avaliar o quão bom é cada indivíduo. Usualmente esta etapa é feita por meio de uma função de aptidão, chamada também de *fitness*, que quantifica o quão bom ou o quão aceitável é o indivíduo em questão para solucionar o problema. Em casos de problemas com muitas restrições, funções baseadas em penalidades são então utilizadas (SIVANANDAM e DEEPA, 2008).

2.2.3 Seleção

A seleção é a parte do algoritmo genético que representa o princípio da sobrevivência do indivíduo mais apto, selecionando para reprodução aqueles que apresentam valores de *fitness* melhores para o objetivo do problema (COELHO e COELHO, 1999).

Um dos métodos mais utilizados para a seleção é o método da Roleta. Neste método cada indivíduo ocupa um espaço em uma roleta, proporcional ao seu *fitness*, assim quanto maior for a aptidão de um indivíduo, maior será a área ocupada pelo mesmo na roleta.

Para montar a roleta é considerada uma probabilidade individual, com base no *fitness* e na soma do *fitness* de todos os indivíduos, e uma probabilidade acumulada, que é a soma da probabilidade individual de todos os indivíduos. Assim, a cada intervalo de probabilidade acumulada está associado um indivíduo, logo um número é sorteado, e o intervalo de probabilidade acumulada que contiver este número terá o indivíduo associado a este intervalo esco-

lhido (MELANIE, 1999). Para exemplificar o processo de montagem da roleta, considere os dados dispostos na tabela 3.

Tabela 3 – Tabela mostrando valores para montagem do algoritmo da Roleta.

Indivíduo	<i>Fitness</i>	Probabilidade Individual	Probabilidade Acumulada
01101	169	14,4	0-14,4
11000	576	49,2	14,4-63,6
01000	64	5,5	63,6-69,1
10011	361	30,9	69,1-100
Total	1170	100,0	

Fonte: Adaptado de Goldberg (1989).

Assim se um número x sorteado pertencer ao intervalo $63,6 < x \leq 69,1$ o indivíduo selecionado será o valor de 01000. Observa-se que quanto maior a probabilidade individual (que por sua vez é proporcional ao *fitness* do indivíduo) maior é a chance do indivíduo ser selecionado, pois maior será o intervalo da probabilidade acumulada (GOLDBERG, 1989).

Outros métodos para seleção de indivíduos que não serão tratados neste trabalho são: ranqueamento, torneio, *breeder*, *steady-state* e boltzman. Informações sobre estes métodos podem ser encontrados em (SIVANANDAM e DEEPA, 2008), (GOLDBERG, 1989), (BÄCK *et al.*, 1997) e (TANOMARU, 1995).

Selecionados os pais, parte-se então para o processo de recombinação e mutação que irão gerar os novos indivíduos a partir dos pais selecionados.

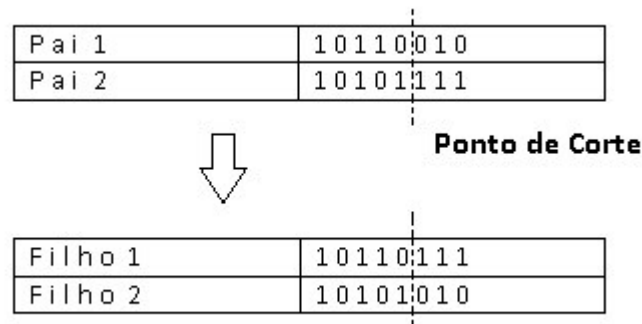
2.2.4 Recombinação e mutação

A recombinação é o processo pelo qual os pais selecionados se reproduzem gerando novos descendentes. O operador de recombinação mais comum é o *crossover* e, segundo Sivanandam e Deepa (2008), procede em três passos:

- Dois pais são selecionados de forma aleatória;
- Um ponto de corte é sorteado ao longo da cadeia de *bits*;
- As cadeias de bits dos pais antes e depois do ponto de corte são cruzadas gerando novos indivíduos.

Para exemplificar observem a figura 10.

Figura 10 - Exemplo de recombinação por crossover

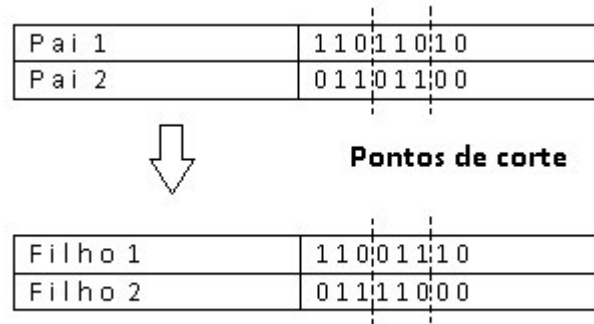


Fonte: Elaboração própria.

De acordo com a figura 10, observa-se que para realizar a operação de *crossover*, basta pegar a cadeia de bits antes do ponto de corte de um dos pais, e então pegar toda a cadeia de bits após o ponto de corte do outro pai, e assim formar um novo indivíduo.

O processo de *crossover* pode ser também realizado com ponto de corte duplo, como ilustra a figura 11.

Figura 11 - Exemplo de recombinação por crossover com ponto de corte duplo.



Fonte: Elaboração própria.

Na recombinação com duplo ponto de corte, apenas a cadeia de *bits* entre os dois pontos de corte é intercambiada.

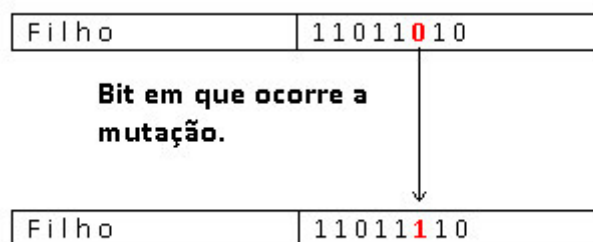
De uma forma geral, o processo de busca utilizando algoritmos genéticos tenta balancear dois pontos em oposição: a busca pelo melhor indivíduo e a expansão do espaço de busca (ÜNAL *et al.*, 2013). Estes dois pontos estão em oposição devido ao fato de que, a cada geração, a população converge para um determinado indivíduo que apresenta melhor desempenho, para evitar esse tipo de convergência é utilizado o processo de mutação.

A mutação é uma forma de evitar que o algoritmo fique preso em mínimos locais e também uma forma de recuperar materiais genéticos que possam ser perdidos com o desaparecimento de determinados pais (SIVANANDAM e DEEPA, 2008).

Tradicionalmente a mutação é um operador raramente usado, contudo sua incidência aumenta com o número de bits que representa cada indivíduo e com o tamanho da população (ÜNAL *et al.*, 2013). A probabilidade de ocorrência do crossover é definida em torno de 0,25 enquanto que a probabilidade de ocorrência de mutação em um bit varia de 0,001 a 0,01.

Para se realizar a mutação é sorteado um número aleatório x para cada *bit* da cadeia que representa o indivíduo. Se o valor de x for menor do que a probabilidade de ocorrência da mutação, então o *bit* em questão é mudado de valor, isto é, se torna *zero* caso seja *um*, e se torna *um* caso seja *zero*. A figura 12 ilustra o procedimento.

Figura 12 - Mutação representada em vermelho.



Fonte: Elaboração própria.

Com o *crossover* e a mutação são gerados os novos indivíduos que irão compor a geração seguinte da população original. A nova população será avaliada de acordo com a função de aptidão, e então se reiniciará o processo de seleção, recombinação e mutação. Estes passos se repetem até que um critério de parada seja satisfeito.

2.2.5 Critérios de parada

Os critérios de parada são estabelecidos de acordo com o tipo de problema e são utilizados para que o algoritmo genético seja finalizado quando uma solução satisfatória for atingida.

Os critérios de parada mais comuns são (SIVANANDAM e DEEPA, 2008):

- Número máximo de gerações: o algoritmo genético é finalizado ao atingir um número máximo de gerações.

- Tempo máximo: um tempo máximo é definido para o algoritmo genético ser finalizado.
- Nenhuma mudança no *fitness*: Se ao longo das gerações nenhuma mudança for notada no *fitness*, o algoritmo genético é finalizado.

Os critérios acima são estabelecidos para que o algoritmo genético pare quando uma solução satisfatória for encontrada, tendo em vista que o algoritmo genético não garante a melhor solução, apenas garante que uma boa solução foi encontrada.

O próximo capítulo aborda o problema da escolha do tempo de amostragem no processo de identificação de sistemas. Será mostrado que a escolha desse tempo pode influenciar tanto a escolha da estrutura quanto a estimação dos parâmetros.

3 ESCOLHA DO TEMPO DE AMOSTRAGEM EM IDENTIFICAÇÃO DE SISTEMAS NÃO LINEARES

Usualmente as variáveis de interesse em um sistema apresentam valores contínuos no tempo, como exemplo pode-se citar a temperatura, pressão, vazão, pH, nível, etc. Para que estas variáveis possam ser tratáveis em sistemas computadorizados elas devem ser amostradas em intervalos regulares de tempo T_s .

O nível de representação do sinal amostrado em relação ao sinal original depende do tempo de amostragem T_s , de forma que o tempo de amostragem se torna importante, pois um comportamento dinâmico que não esteja presente nos dados não pode ser modelado (NELLES, 2001).

A fim de que um sinal amostrado seja uma boa representação de um determinado sinal, o teorema de Shannon (SHANNON, 1949) afirma que a frequência de amostragem deve ser duas vezes maior do que a maior frequência de interesse, já Nyquist (NYQUIST, 1928) afirma que a frequência de amostragem deve ser de 5 a 10 vezes a maior frequência de interesse.

Contudo, dessas afirmações surge um problema para os processos de identificação de sistemas, pois nem sempre a maior frequência de interesse é conhecida, o que torna os teoremas de Shannon e Nyquist não aplicáveis para o processo de identificação de sistemas não lineares.

Por que não amostrar o sinal com a maior taxa de amostragem disponível, de forma que T_s se aproxime de zero e o sinal amostrado seja exatamente igual ao sinal de saída do sistema? Para responder essa pergunta, é necessário analisar os processos de identificação de sistema e qual a influência do tempo de amostragem nas etapas necessárias à obtenção de um modelo.

Nas próximas sessões serão apresentadas algumas considerações e análises a respeito destas etapas e da influência do tempo de amostragem, bem como a apresentação de um método que tem sido muito utilizada na bibliografia e os problemas relacionados com o mesmo. Também será abordada a questão principal sobre a qual reside este trabalho: É possível a otimização dos processos de identificação de sistemas utilizando algoritmos genéticos para selecionar a taxa de amostragem dos sinais? A resposta para esta pergunta é respondida ao final deste trabalho.

3.1 Seleção da estrutura e o tempo de amostragem

O processo de seleção da estrutura na identificação de sistemas é o passo no qual os regressores são escolhidos para compor o modelo, tais regressores são usualmente as variáveis de entrada e saída. A presente sessão tem por objetivo mostrar a influência que o tempo de amostragem tem na seleção da estrutura. Para tanto, considere um sistema dinâmico dado pela equação 3.1.

$$\dot{y} + 3y = u \quad (3.1)$$

Esta equação pode ser discretizada utilizando o método de Euler dado por 3.2.

$$\dot{y} \cong \frac{y(k) - y(k - t_d)}{t_d} + e_d(k) \quad (3.2)$$

Assim a equação 3.1 pode ser escrita como:

$$\frac{y(k) - y(k - t_d)}{t_d} + e_d(k) + 3y(k) = u(k) \quad (3.3)$$

$$\frac{y(k) - y(k - t_d) + t_d e_d(k) + t_d 3y(k)}{t_d} = u(k) \quad (3.4)$$

$$y(k)(1 + 3t_d) = t_d u(k) + y(k - t_d) - t_d e_d(k) \quad (3.5)$$

$$y(k) = \frac{t_d u(k)}{(1 + 3t_d)} + \frac{y(k - t_d)}{(1 + 3t_d)} - \frac{t_d e_d(k)}{(1 + 3t_d)} \quad (3.6)$$

De acordo com a equação 3.6, observa-se que à medida que se diminui o tempo de amostragem t_d a influência das variáveis de entrada $u(k)$ e do erro $e_d(k)$ também diminui por serem diretamente proporcionais ao tempo de amostragem. Assim o limite de 3.6 quando t_d tende a zero é dado por 3.7.

$$\lim_{t_d \rightarrow 0} y(k) = y(k) \quad (3.7)$$

A equação 3.7 mostra que quando o tempo de amostragem t_d tende a zero, a identificação do sistema se torna impossível de ser realizada.

Um resultado semelhante ao da equação 3.7 pode ser encontrado quando se analisa a escolha da estrutura do ponto de vista da taxa de redução do erro (ERR). Considere a taxa de redução do erro dado pela equação 2.19, a qual pode ser escrita como na equação 3.8.

$$[ERR]_i = \frac{\sum_{k=1}^N \hat{g}_i^2 w_i^2}{\sum_{k=1}^N y(k)^2} \quad (3.8)$$

De acordo com Aguirre e Billings (1995b) o vetor de parâmetros \hat{g} pode ser estimado utilizando 3.9.

$$\hat{g}_i = \frac{\sum_{k=1}^N y(k) w_i(k)}{\sum_{k=1}^N w_i(k)^2} \quad (3.9)$$

Substituindo 3.9 em 3.8 tem-se a equação 3.10.

$$[ERR]_i = \frac{\sum_{k=1}^N \left(\frac{\sum_{k=1}^N y(k) w_i(k)}{\sum_{k=1}^N w_i(k)^2} \right)^2 w_i^2}{\sum_{k=1}^N y(k)^2} \quad (3.10)$$

O regressor mais comum nos processos de identificação é $y(k - t_d)$. Assim, fazendo $w_1 = y(k - t_d)$ e substituindo na equação 11, tem-se:

$$[ERR]_1 = \frac{\sum_{k=1}^N \left(\frac{\sum_{k=1}^N y(k) y(k - t_d)}{\sum_{k=1}^N y(k - t_d)^2} \right)^2 y(k - t_d)^2}{\sum_{k=1}^N y(k)^2} \quad (3.11)$$

Calculando o limite da equação 3.11 quando t_d tende a zero, tem-se:

$$\lim_{t_d \rightarrow 0} [ERR]_1 = 1. \quad (3.12)$$

O resultado em 3.12 é facilmente obtido quando se nota que, quando t_d tende a zero, $y(k - t_d)$ se aproxima de $y(k)$.

Em relação ao processo e escolha da estrutura, a equação 3.12 mostra que o único regressor a ser inserido no modelo seria $y(k - t_d)$, por apresentar taxa de redução do erro igual a um.

Na prática o tempo de amostragem não se torna zero, de forma que ERR será muito próxima de um, e o valor de ERR para todos os outros regressores serão tão pequenos que será impossível saber qual regressor deve ser inserido no modelo.

A sessão seguinte apresenta a influência do tempo de amostragem na estimação dos parâmetros do modelo.

3.2 Estimação dos parâmetros do modelo e o tempo de amostragem

A relação da estimação dos parâmetros e do tempo de amostragem é um pouco mais complexa do que no caso da seleção da estrutura, visto que uma taxa de amostragem rápida fará com que os regressores do tipo $y(k - t_d)$, $y(k - 2t_d)$ e $y(k - 3t_d)$ sejam praticamente iguais, de forma que a matriz de regressores Ψ terá suas colunas muito próximas de serem linearmente dependentes, causando assim um mau condicionamento e, conseqüentemente, a estimação dos parâmetros $\hat{\theta}$ pela equação 2.33 não será uma boa estimativa. Por outro lado, uma taxa de amostragem maior permitiria um maior número de amostras, o que pode garantir uma melhor exatidão nos valores dos parâmetros.

O aumento da taxa de amostragem para sistemas discretizados está também ligado ao aumento do erro. Considere a equação 3.6 escrita da forma apresentada na equação 3.13.

$$y(k) = \psi\theta - \frac{t_d e_d(k)}{(1 + 3t_d)} \quad (3.13)$$

O *bias*, ou viés, ou tendência do valor estimado de θ , é dado pela equação 3.14.

$$B = E\{\hat{\theta}\} - \theta \quad (3.14)$$

E de acordo com a equação 2.33 tem-se:

$$B = E\{(\Psi^T \Psi)^{-1} \Psi^T y(k)\} - \theta. \quad (3.15)$$

Substituindo a equação 3.13 na equação 3.15, tem-se:

$$B = E\left\{(\Psi^T \Psi)^{-1} \Psi^T \left(\Psi \theta - \frac{t_d e_d(k)}{(1 + 3t_d)}\right)\right\} - \theta. \quad (3.16)$$

$$B = E\left\{\theta - (\Psi^T \Psi)^{-1} \Psi^T \frac{t_d e_d(k)}{(1 + 3t_d)}\right\} - \theta. \quad (3.17)$$

$$B = E\{\theta\} - E\left\{(\Psi^T \Psi)^{-1} \Psi^T \frac{t_d e_d(k)}{(1 + 3t_d)}\right\} - \theta. \quad (3.18)$$

Como θ não é uma estimativa, o seu valor esperado é ele mesmo e, subtraindo de θ no final da equação, tem-se apenas:

$$B = \frac{t_d}{(1 + 3t_d)} E\{(\Psi^T \Psi)^{-1} \Psi^T e_d(k)\}. \quad (3.19)$$

Observa-se assim que o bias do estimador depende do tempo de amostragem t_d , diminuindo quando este diminui.

De acordo com Aguirre e Billings (1995b), a seleção da estrutura e a estimação dos parâmetros dependem do tempo de amostragem em sentido opostos. Enquanto a seleção da estrutura é melhor com um tempo de amostragem maior, a estimação dos parâmetros funciona de forma mais adequada quando utilizado um tempo de amostragem menor. Assim parece ser razoável aplicar diferentes taxas de amostragem na execução de ambos os processos.

A sessão seguinte apresenta um método para determinação do tempo de amostragem baseado em análise de correlação dos sinais de saída e tem sido aplicada em diversos trabalhos na literatura.

3.3 Determinação do tempo de amostragem por análise de correlação

Como visto anteriormente faz-se necessário a escolha de uma taxa de amostragem que possa ser utilizada sem prejudicar o processo de seleção da estrutura e da estimação dos parâmetros.

Assim, foi proposto em Aguirre e Billings (1995b) um método para seleção do tempo de amostragem baseado na análise de correlação do sinal de saída.

A correlação de um sinal pode ser entendida como o nível de redundância de um sinal, isto é, quão redundantes são as amostras. O método descrito utilizando correlação será detalhado nas próximas linhas e pode ser encontrado com mais detalhes em (BILLINGS e AGUIRRE, 1995b).

O primeiro passo para realizar a escolha do tempo de amostragem por análise de correlação é superamostrar o sinal, isto é, amostrar o sinal com um tempo de amostragem muito pequeno. Com o sinal amostrado, agora se deseja determinar uma taxa de decimação $\Delta \in \mathbb{N}$ com a qual o sinal original será decimado, gerando assim um sinal com um tempo de amostragem maior.

Dado um sinal $y(k)$ superamostrado, se determina as autocovariâncias linear e não-linear dadas pelas equações 3.20 e 3.21.

$$r_{y^*}(\tau) = E[(y^*(k) - \overline{y^*(k)})((y^*(k - \tau) - \overline{y^*(k)}))] \quad (3.20)$$

$$r_{y^{*2}}(\tau) = E[(y^{*2}(k) - \overline{y^{*2}(k)})((y^{*2}(k - \tau) - \overline{y^{*2}(k)}))] \quad (3.21)$$

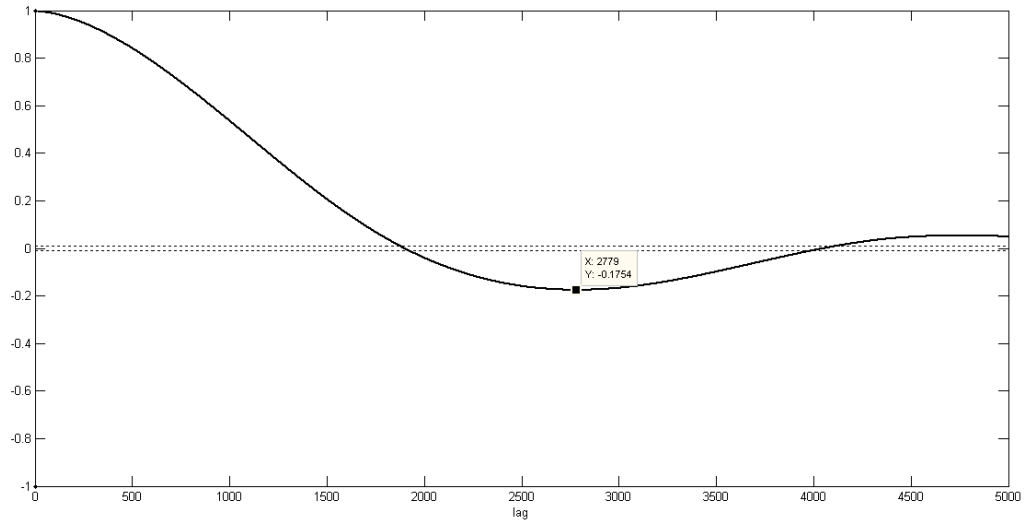
A partir das funções de autocovariâncias são determinados os primeiros mínimos τ_{y^*} e $\tau_{y^{*2}}$, em relação às funções 3.20 e 3.21, respectivamente, os quais servirão para determinar o valor de trabalho τ_m de acordo com 3.22.

$$\tau_m = \min[\tau_{y^*}, \tau_{y^{*2}}] \quad (3.22)$$

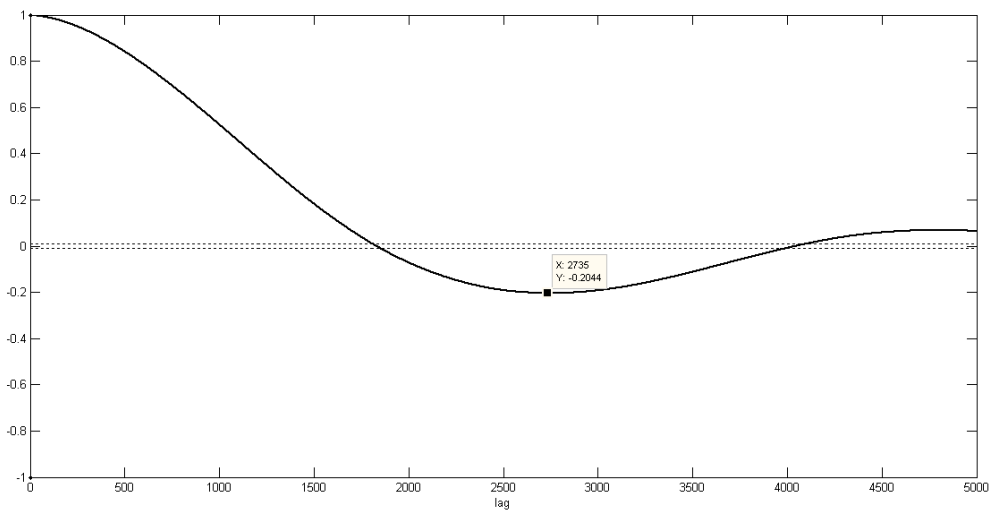
O valor de decimação Δ é escolhido de forma a satisfazer a equação 3.23.

$$10\Delta \leq \tau_m \leq 20\Delta \quad (3.23)$$

Como exemplo será mostrado o processo de escolha do tempo de amostragem aplicado na identificação de um modelo de temperatura para a célula a combustível, as figuras 13 e 14 mostram as funções de autocovariância linear e não linear do sinal de saída da célula.

Figura 13 - Função de autocovariância de $y(k)$.

Fonte: Elaboração própria.

Figura 14 - Função de autocovariância de $y^2(k)$.

Fonte: Elaboração própria.

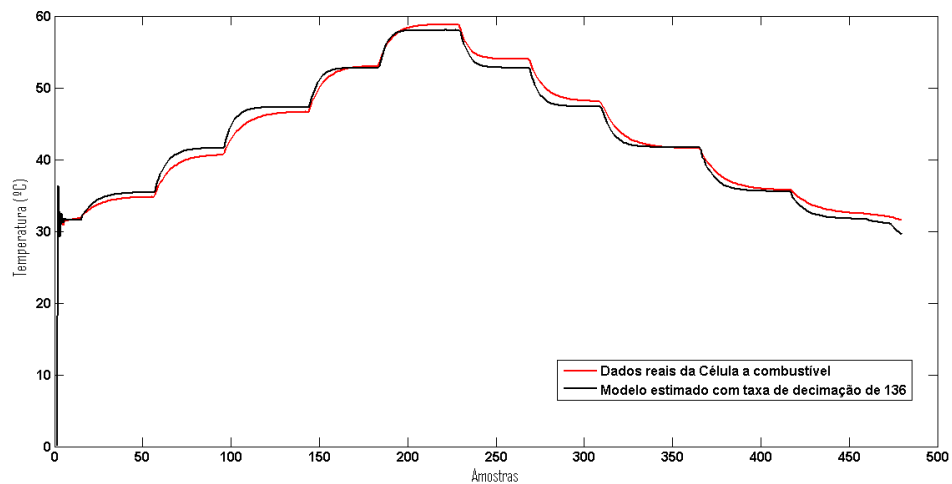
De acordo com a figuras 13 e 14, tem-se os pontos de mínimos em 2779 para a autocovariância linear e 2735 para autocovariância não linear. O valor de trabalho τ_m é o menor desses dois valores, sendo assim 2735.

A equação 3.23 pode ser escrita da seguinte forma.

$$10\Delta \leq 2735 \leq 20\Delta \quad (3.24)$$

De onde se tem que o valor de Δ está entre 136 e 273. De fato bons modelos podem ser encontrados com valores nesta faixa. A figura 15 exemplifica um modelo encontrado com a taxa de decimação de 136.

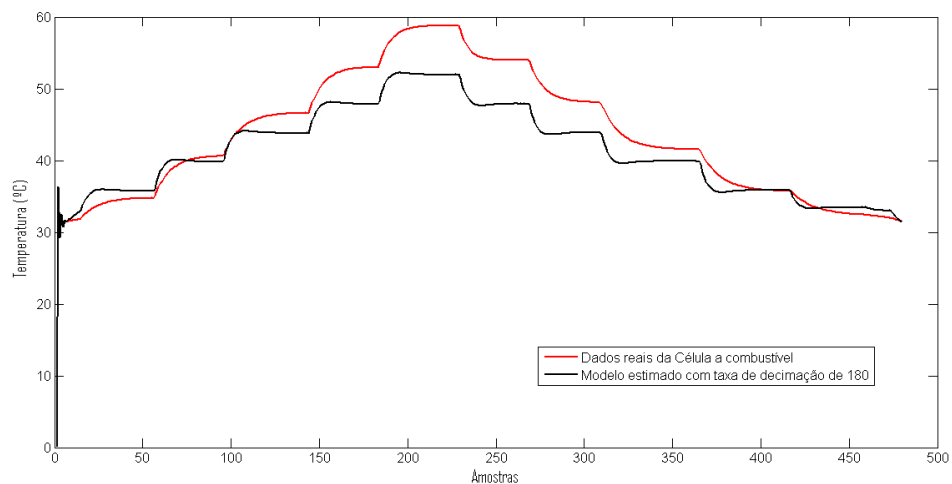
Figura 15 - Validação de modelo encontrado com taxa de decimação de 136.



Fonte: Elaboração própria.

Contudo, algumas taxas de decimação dentro desta faixa acabam por resultar em modelos ruins, observe a figura 16.

Figura 16 - Validação de modelo encontrado com taxa de decimação de 180.



Fonte: Elaboração própria.

O fato de algumas taxas de decimação dentro da faixa definida pela equação 3.23 resultar em modelos ruins, faz com que seja necessário o ajuste da taxa de decimação por tentativa e erro. Assim sendo, a proposição de um algoritmo genético para escolha do tempo de amostragem tem por objetivo a seleção de uma taxa de decimação que apresente desempenho melhor do que as demais, sem que o processo necessite de ajustes com tentativa e erro.

O capítulo a seguir apresenta a metodologia utilizada para o desenvolvimento do presente trabalho, bem como o *software* desenvolvido e a célula a combustível utilizada para realizar os testes.

4 METODOLOGIA

Como visto no capítulo anterior do presente trabalho, a taxa de amostragem do sinal que será utilizada no processo de identificação tem forte influência sobre os resultados, tanto para a determinação da estrutura, quanto para a estimação dos parâmetros do modelo a ser identificado. A escolha de um tempo de amostragem muito pequeno é tão prejudicial quanto a escolha de um tempo de amostragem grande. Foi mostrado também que o método para seleção do tempo de amostragem disponível na literatura não satisfaz condições de otimalidade e muitas vezes fornecem tempos que impossibilitam o processo de identificação.

Tendo estes pontos em vista, o presente trabalho tem por objetivo a obtenção de um novo método para a determinação do tempo de amostragem por meio da decimação do sinal original. As próximas sessões têm por objetivo descrever os passos realizados para a implementação de algoritmos genéticos aplicados na determinação da taxa de decimação.

4.1 Algoritmo genético aplicado à determinação do tempo de amostragem

Para implementação do método proposto foram escolhidas técnicas e convenções já consagradas na literatura e algumas considerações foram feitas.

Os algoritmos genéticos permitem a obtenção de soluções satisfatórias para problemas complexos de uma forma simplificada. A incorporação de indivíduos na forma de cromossomos e a aplicação de operadores de recombinação e mutação permite que características desejadas da solução sejam mantidas entre gerações e que a busca pela melhor solução seja realizada.

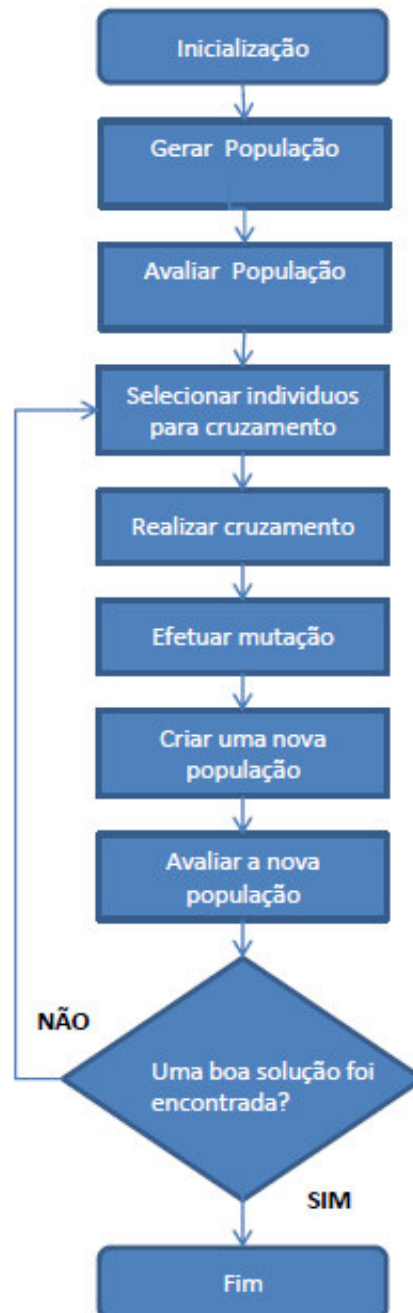
A implementação dos algoritmos genéticos se dá por meio de programação. O fluxograma para implementação dos mesmos é apresentado na figura 17.

O fluxograma se inicia com a geração da população inicial, que irá depender do tipo de problema e o espaço de busca a ser percorrido. O tamanho da população deve ser cuidadosamente pensado, tendo em vista que quanto maior a população, maior será o processamento necessário.

O passo seguinte à geração da população é a avaliação da função de aptidão, que irá permitir que os indivíduos sejam classificados e selecionados para a recombinação. Em seguida são aplicadas aos indivíduos selecionados as operações de recombinação e mutação, gerando assim a nova população.

A nova população é então avaliada de acordo com a função de aptidão, é verificado o critério de parada definido, caso o critério de parada seja satisfeito o algoritmo genético é finalizado, caso contrário o algoritmo retorna ao passo de selecionar os melhores indivíduos, aplicar os operadores de recombinação e mutação repetindo assim até que o critério de parada seja satisfeito.

Figura 17 - Fluxograma utilizado em algoritmos genéticos.



Fonte: Adaptado de (COELHO e COELHO, 1999).

No presente trabalho foi criado um algoritmo genético para a determinação da taxa de amostragem do sinal de saída, a fim de realizar uma identificação NARMAX.

As próximas sessões têm por objetivo descrever as etapas de implementação, bem como as técnicas e considerações supracitadas.

4.1.1 Representação do cromossomo

A representação dos cromossomos foi feita utilizando a representação canônica apresentada no item 2.2.1.

Cada indivíduo da população foi representado por dois genes, sendo estes genes as taxas de decimação para a seleção da estrutura e estimação dos parâmetros.

Assim, cada par de taxas de decimação define um único indivíduo. Como as taxas de decimação são números inteiros, tais valores foram convertidos em base binária. A tabela 4 representa alguns indivíduos e suas representações binárias.

Tabela 4 – Representação dos indivíduos no algoritmo genético proposto.

Indivíduo	Taxas de decimação		Representação binária
1	56	67	00111000 01000011
2	44	55	00101100 00110111
3	121	215	01111001 11010111

Fonte: Elaboração própria.

Com base em um sinal superamostrado, as taxas de decimação geram novos sinais com um tempo de amostragem maior. Estes sinais serão utilizados no processo de identificação não linear.

4.1.2 Função de avaliação

Para avaliar cada um dos indivíduos foi realizado um processo de identificação utilizando suas respectivas taxas de decimação nos processos de escolha da estrutura e estimação dos parâmetros. No apêndice C é possível ver a função de *fitness* desenvolvida.

Com os modelos obtidos a partir de cada indivíduo, foi realizada a validação de cada modelo utilizando uma massa de dados comum a todos.

Para avaliar os indivíduos, o primeiro método proposto foi a soma do erro quadrático (*Sum Square Error - SSE*), dada pela equação 4.1.

$$SSE = \sum_{t=1}^N [y(t) - \hat{y}(t)]^2 \quad (4.1)$$

O *SSE* define o quanto o modelo identificado difere dos dados de validação e quanto menor for esta quantidade, melhor será o modelo. Contudo o *SSE* não mede a complexidade do modelo, de forma que, se apenas este critério for avaliado, os modelos resultantes podem apresentar uma complexidade muito grande.

Para contornar este problema foi, então, utilizado como método para avaliar os indivíduos o critério da informação Bayesiana (*Bayesian Information Criterion - BIC*).

O *BIC* proposto por Schwarz (1978) define a qualidade do modelo com base no número de termos e na soma do erro quadrático. Este método evita que modelos demasiadamente complexos sejam escolhidos pelo algoritmo genético.

O método *BIC* é definido pela equação 4.2.

$$BIC = N \ln\left(\frac{1}{N} \sum_{t=1}^N [y(t) - \hat{y}(t)]^2\right) + p \ln N \quad (4.2)$$

Onde:

N – tamanho da amostra;

$y(t)$ – variável de saída dos dados de validação;

$\hat{y}(t)$ – variável de saída do modelo estimado;

p – número de termos do modelo.

Como pode ser notado na equação 4.2, o critério da informação bayesiana depende do tamanho da amostra e as taxas de decimação fazem com que as amostras tenham diferentes tamanhos. Nota-se que o *BIC* depende do tamanho da amostra, logo a fim de se utilizar este critério convencionou-se utilizar todas as amostras no processo de validação com taxa de decimação de cem de forma que todos os indivíduos fossem de mesmo tamanho, mesmo esta taxa não correspondendo às taxas originais de seleção da estrutura e estimação dos parâmetros dos indivíduos.

4.1.3 Seleção, recombinação e mutação

O processo de seleção foi feito utilizando o método Roleta já explicado na sessão 2.2.3, por apresentar melhor desempenho para problemas complexos (GOLDBERG, 1989). Foi utilizado também o conceito de Elitismo, que consiste em manter de uma geração para a próxima os melhores indivíduos da geração atual.

O processo de recombinação foi feito utilizando *crossover* de um ponto e a mutação foi por alternância simples de bit, ambos os processos foram descritos na sessão 2.2.4.

Os algoritmos desenvolvidos para o método proposto podem ser encontrados no apêndice C.

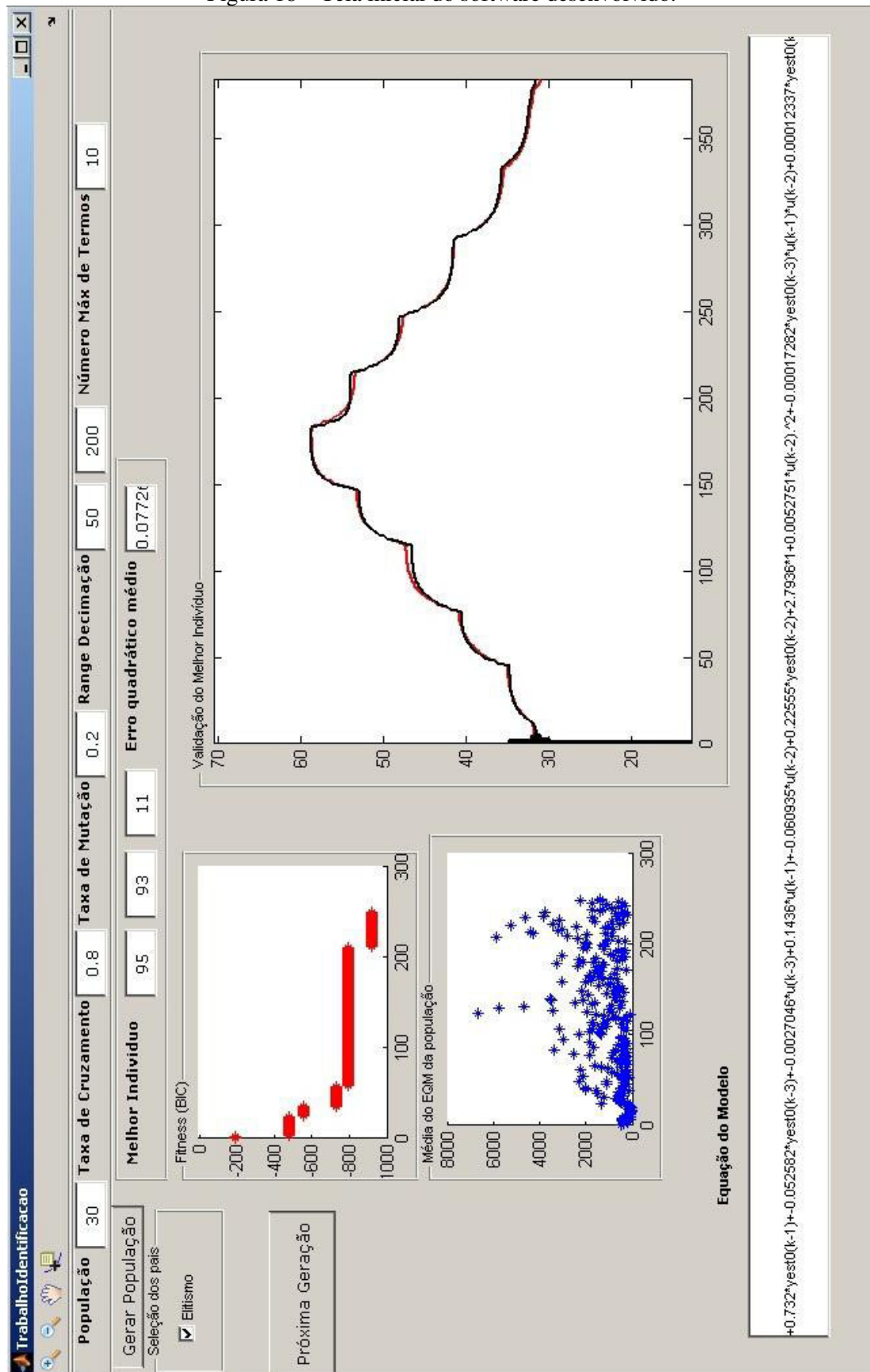
A próxima sessão apresenta o software desenvolvido para aplicação do método proposto.

4.2 Software para determinação da taxa de decimação por meio de algoritmos genéticos

Para aplicar o novo método proposto neste trabalho, foi desenvolvido um software utilizando o ambiente Matlab®. Este ambiente foi escolhido por apresentar funções prontas, que facilitam tanto o processo de identificação de sistemas quanto o algoritmo genético.

A figura 18 apresenta a tela do software desenvolvido.

Figura 18 – Tela inicial do software desenvolvido.



Fonte: Elaboração própria.

A seguir é feita uma breve descrição de cada campo e cada objeto presentes na interface gráfica do software desenvolvido.

População: Este campo determina o tamanho da população inicial do algoritmo genético;

Taxa de cruzamento: Define a probabilidade de ocorrência de cruzamento entre dois indivíduos;

Taxa de mutação: Define a probabilidade de ocorrência de mutação;

Range Decimação: Permite definir um intervalo em que os indivíduos da população inicial serão gerados, esta função é utilizada para limitar o espaço de busca;

Número máximo de termos: Permite que o usuário defina um número máximo de termos para os modelos a serem encontrados;

Melhor Indivíduo: Mostra as taxas de decimação e o número de termos do melhor indivíduo da população atual;

Soma do erro quadrático: Mostra a soma do erro quadrático do melhor indivíduo;

Carregar dados: Permite carregar os dados de identificação e validação no programa;

Gerar População: Gera a população inicial com base no tamanho da população definido no campo “População”;

Elitismo: Aplica elitismo ao algoritmo genético;

Fitness: Gráfico do *Fitness* do melhor indivíduo de cada população;

Média do EQM da população: Gráfico do erro quadrático médio da população;

Validação do melhor indivíduo: Mostra um gráfico comparativo do melhor indivíduo com os dados de validação;

Equação do modelo: Mostra a equação do melhor modelo encontrado.

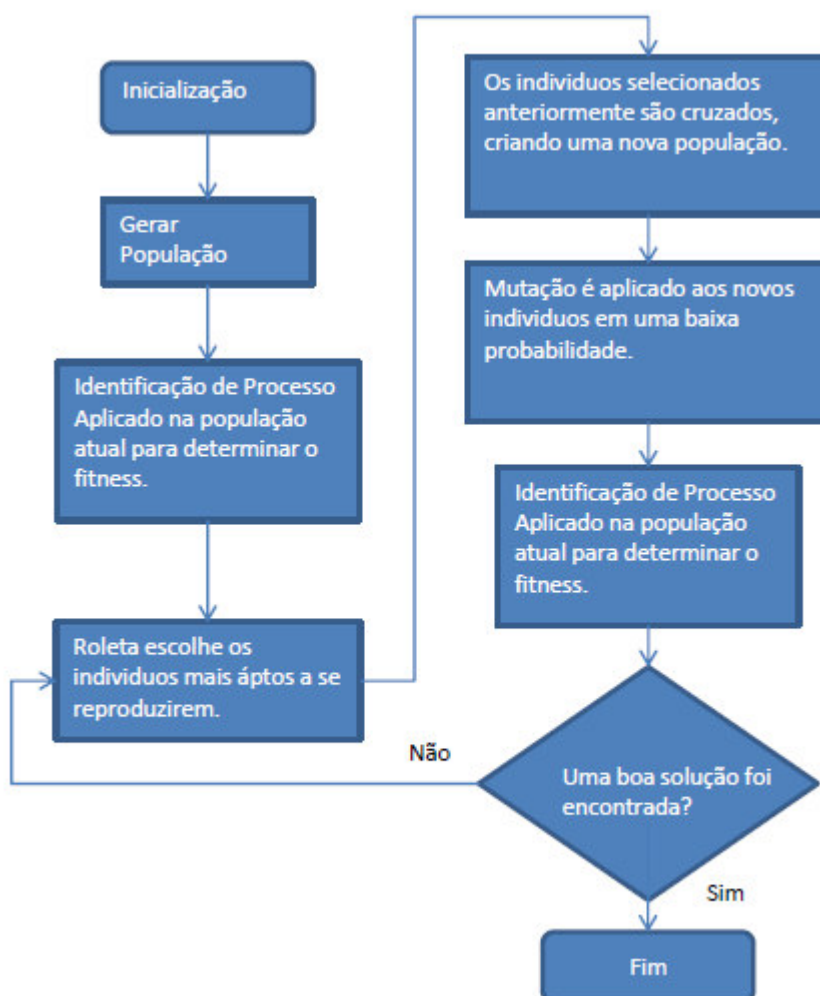
A operação do software é bem simples, o primeiro passo é carregar os dados de identificação e validação. A seguir configuram-se os parâmetros do algoritmo genético, tais como tamanho da população, taxa de cruzamento e mutação, range da taxa de decimação e número máximo dos termos.

Definidos estes parâmetros, utiliza-se o botão “Gerar população” para inicializar a população e então o algoritmo genético está pronto para iniciar. Para rodar o algoritmo genético basta clicar em “Próxima Geração”, o algoritmo inicia o processo de evolução da população, até que um modelo satisfatório seja encontrado.

No *software* desenvolvido não se estabeleceu nenhum critério de parada, deixando a cargo do usuário a finalização do algoritmo.

O software desenvolvido segue basicamente o algoritmo da figura 19.

Figura 19 - Fluxograma do algoritmo genético desenvolvido.



Fonte: Elaboração própria.

4.3 Célula a combustível

Para testar o método proposto neste trabalho, foi realizado um processo de identificação do modelo de temperatura de uma célula a combustível. Tal sistema foi escolhido por ainda não ter sido apresentado na literatura até a data de execução deste trabalho, nenhum modelo NARMAX da temperatura em função da corrente drenada nos terminais da célula a combustível.

Os dados de identificação usados neste trabalho foram provenientes de um simulador de célula a combustível desenvolvido por Avelar (2012). Tal procedimento foi utilizado devido à agilidade em se recolher os dados provenientes do simulador e por evitar a exposição aos perigos inerentes a uma célula a combustível.

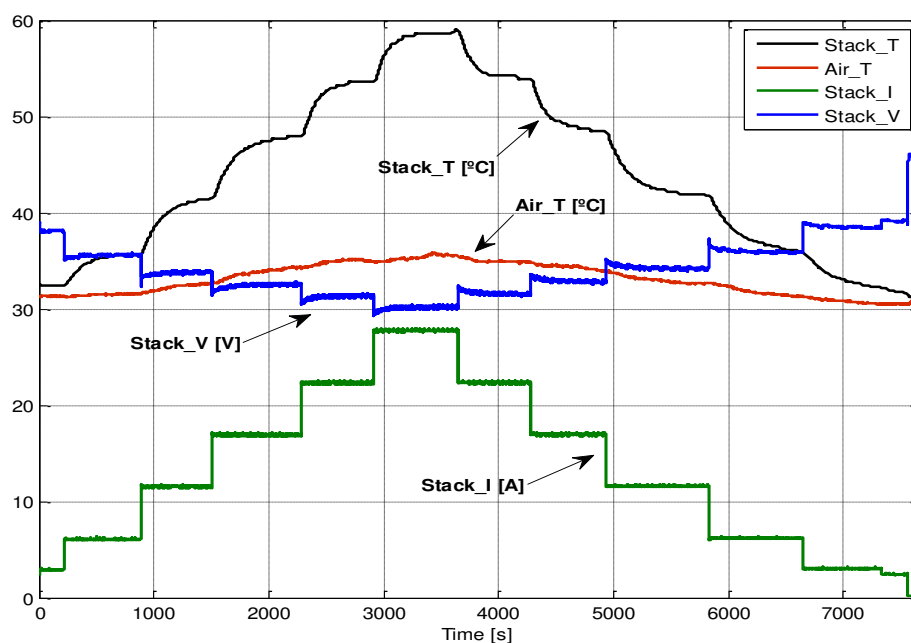
Contribuiu ainda para a utilização do simulador a grande fidelidade do simulador desenvolvido no trabalho supracitado quando comparado com dados reais da célula.

Ainda há que se acrescentar que, mesmo sendo os dados de identificação provenientes do simulador da célula a combustível, os dados de validação foram escolhidos como dados reais provenientes da célula a combustível.

As células a combustível apresentam diversas vantagens quando comparadas com outras fontes de energia, dentre as principais pode-se citar maior eficiência em relação a outras fontes, baixa emissão de poluentes, baixo nível de ruído e tamanhos e pesos variados, permitindo seu uso desde aparelhos eletrônicos e até em automóveis (Barbir, 2005 apud AVELAR, 2012).

A temperatura na célula a combustível de membrana polimérica varia de acordo com a corrente elétrica drenada dos seus terminais (AVELAR *et al.*, 2009). A figura 20 mostra uma relação da temperatura da célula (Stack_T) e a corrente drenada de seus terminais (Stack_I).

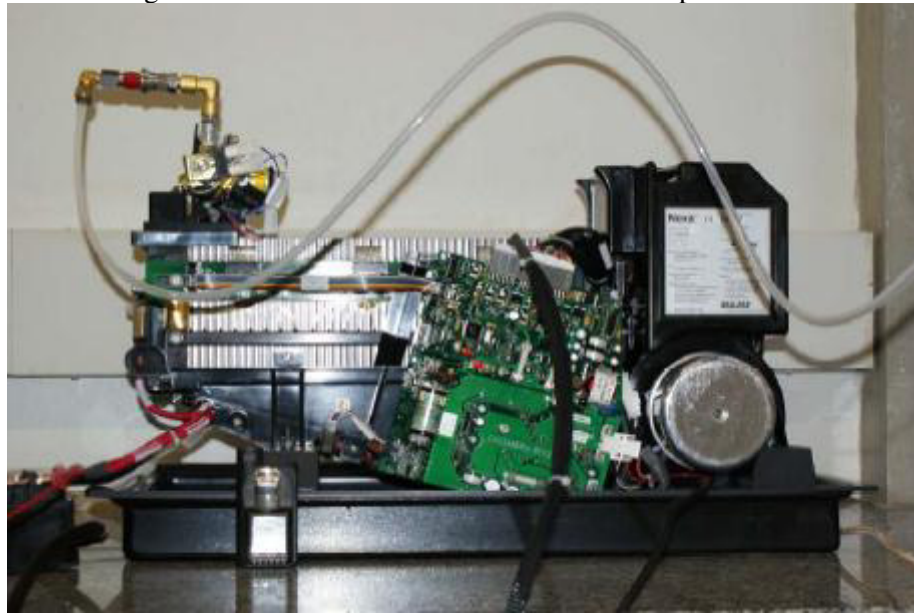
Figura 20 - Gráfico mostrando a variação da temperatura com a corrente drenada da célula.



Fonte: (AVELAR, 2012).

A figura 21 mostra a célula a combustível utilizada por (AVELAR, 2012) para a produção do simulador da célula utilizado neste trabalho.

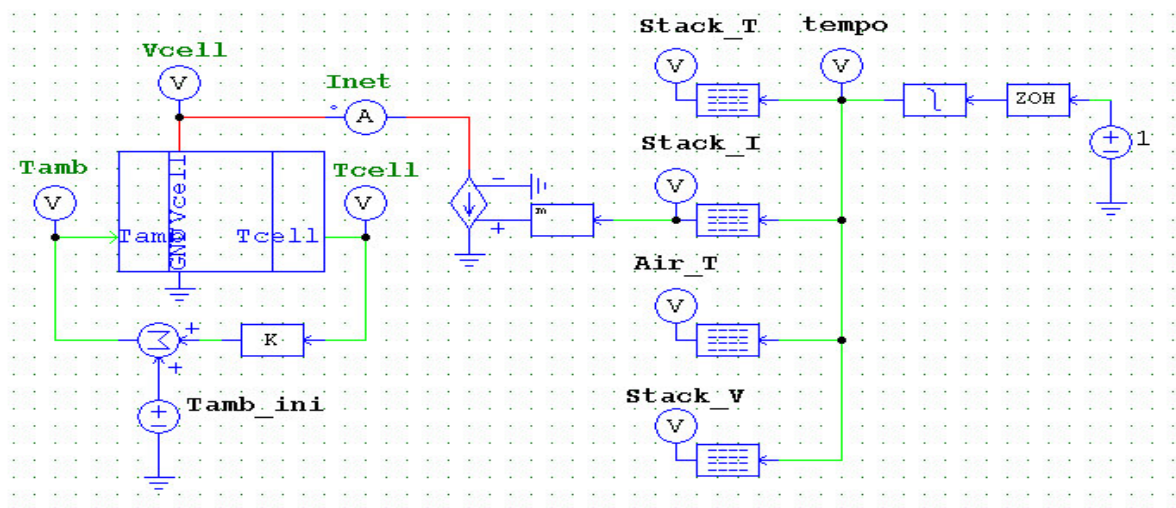
Figura 21 - Célula a combustível de membrana polimérica.



Fonte: (AVELAR, 2012).

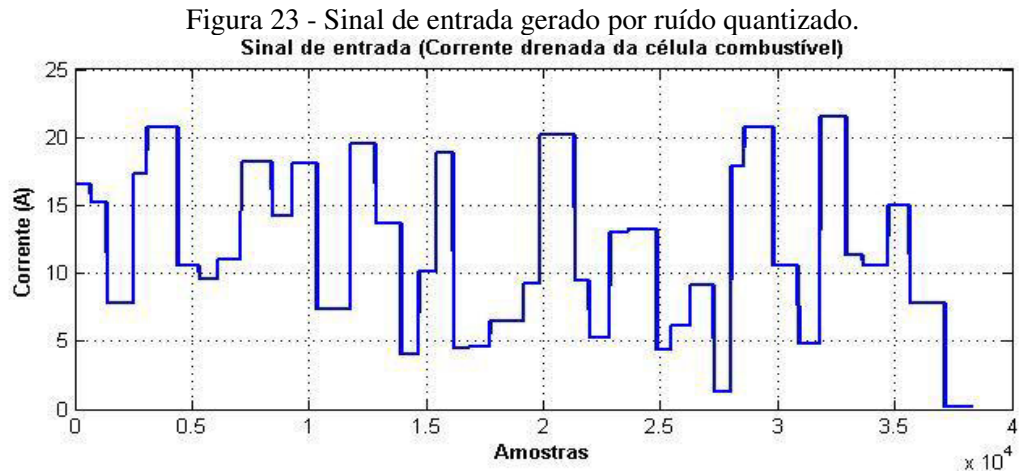
O simulador da célula combustível desenvolvido por AVELAR (2012) foi feito no software PSIM® e possibilitou aplicar entradas variadas no simulador e recolher o sinal de saída. A figura 22 mostra o simulador da célula desenvolvido no software supracitado.

Figura 22 - Simulador de célula a combustível.



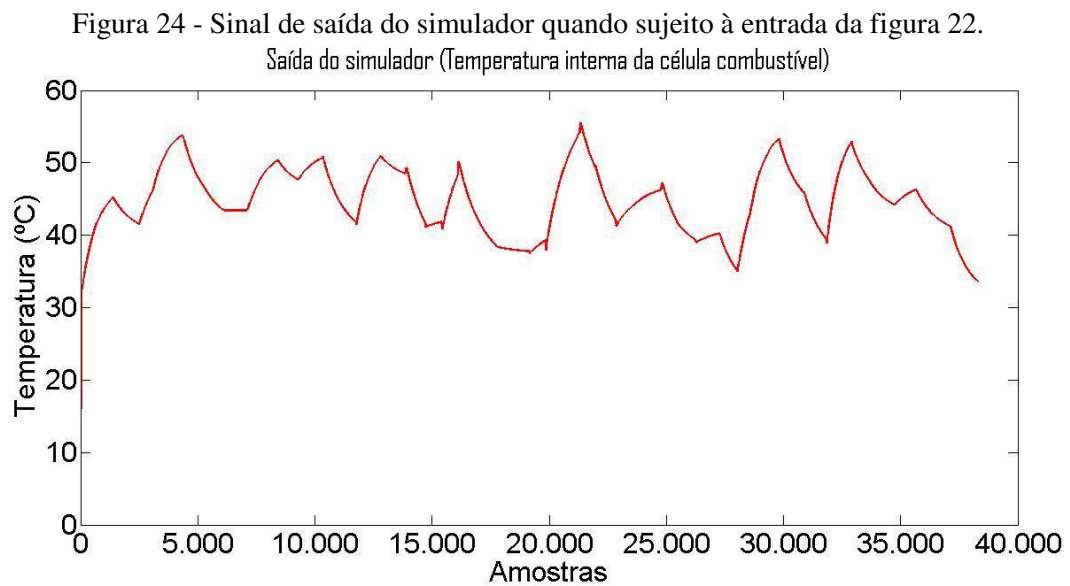
Fonte: (AVELAR, 2012).

O simulador mostrado na figura 22 permite que se obtenha a temperatura da célula (Tcell) de acordo com a entrada que é fornecida (Stack_I), assim, no presente trabalho foi utilizado o método do ruído quantizado já tratado na sessão 2.1 para gerar o sinal de entrada e este é mostrado na figura 23.



Fonte: Elaboração própria.

A figura 24 mostra o sinal de saída do simulador quando sujeito ao sinal de entrada da figura 23.



Fonte: Elaboração própria.

O próximo capítulo apresenta os resultados obtidos com o método proposto aplicado na identificação do modelo de temperatura da célula a combustível. É mostrado uma comparação do modelo obtido utilizando análise de correlação para determinação da taxa de decimação e consequente obtenção da taxa de amostragem, como também o método proposto utilizando algoritmos genéticos para determinação da taxa de amostragem em função da decimação do sinal original.

5 RESULTADOS

Os resultados apresentados neste capítulo se referem à comparação do método de determinação da taxa de amostragem utilizando algoritmos genéticos e o método da correlação aplicado na identificação do modelo de temperatura de uma célula a combustível.

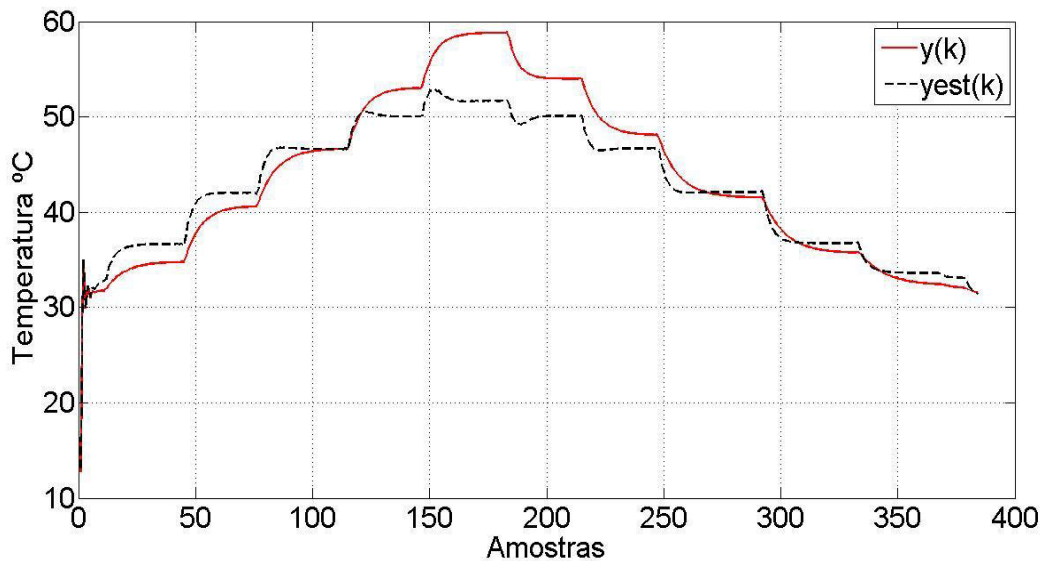
Assim, com os dados apresentados nas figuras 22 e 23 foi feita a identificação do modelo de temperatura utilizando um modelo NARMAX de 3ª ordem, tempo de atraso 3 para n_y e n_u , o que gerou 84 termos candidatos.

A estrutura foi selecionada utilizando taxa de redução do erro detalhada na sessão 2.1.3 do presente trabalho, já os parâmetros foram estimados utilizando mínimos quadrados. O modelo obtido é dado pela equação 5.1.

$$\begin{aligned} y(k) = & 0,9958y(k-1) + 0,1782y(k-2) - 0,5992u(k-2) + 0,5057u(k-1) \quad (5.1) \\ & - 0,0067y(k-1)y(k-3) + 0,00003y(k-2)^2 - 0,0433u(k-3) \\ & - 0,00007y(k-3)^2u(k-1) + 0,0109y(k-2)u(k-2) \\ & - 0,00009u(k-2)^2u(k-3) \end{aligned}$$

O modelo da equação 5.1 foi validado utilizando dados reais da célula combustível e o resultado é apresentado na figura 25.

Figura 25 - Validação do modelo da equação 5.1.



Fonte: Elaboração própria.

Na figura 25, $y(k)$ é o valor real da temperatura medida da célula a combustível e $y_{est}(k)$ é o valor estimado pelo modelo 5.1. Observa-se que, para a faixa de temperatura entre 50 e 60° C, o modelo difere muito dos valores reais.

Com objetivo de tentar obter um modelo que represente de forma mais fiel o comportamento da temperatura em função da corrente na célula a combustível, o software desenvolvido neste trabalho foi utilizado para obter um modelo a partir dos dados das figuras 23 e 24.

Os parâmetros do algoritmo genético são apresentados na tabela 5.

Tabela 5 – Parâmetros de configuração do algoritmo genético.

Tamanho da população	50
Número máximo de gerações	250
Tipo de Seleção	Roleta
Tipo de mutação	Inversão de bit
Tipo de Crossover	Um ponto (2 Pais)
Tipo de substituição	Elitismo
Probabilidade de crossover	0,8
Probabilidade de mutação	0,05
Número máximo de termos	10

Fonte: Elaboração própria.

Já a tabela 6 apresenta os resultados obtidos com o algoritmo genético.

Tabela 6 – Resultados do algoritmo genético.

Número de termos	Erro quadrático médio	Número de Gerações
7	1,159	1
7	0,244	5
10	0,084	20
8	0,075	40
10	0,067	240

Fonte: Elaboração própria.

De acordo com a tabela 6, nota-se uma melhoria substancial no erro quadrático médio, quando se compara o primeiro modelo de 7 termos, com o modelo de 8 termos, o erro diminuiu cerca de 15 vezes em apenas 40 gerações. Contudo após 240 gerações o erro foi de 0,075

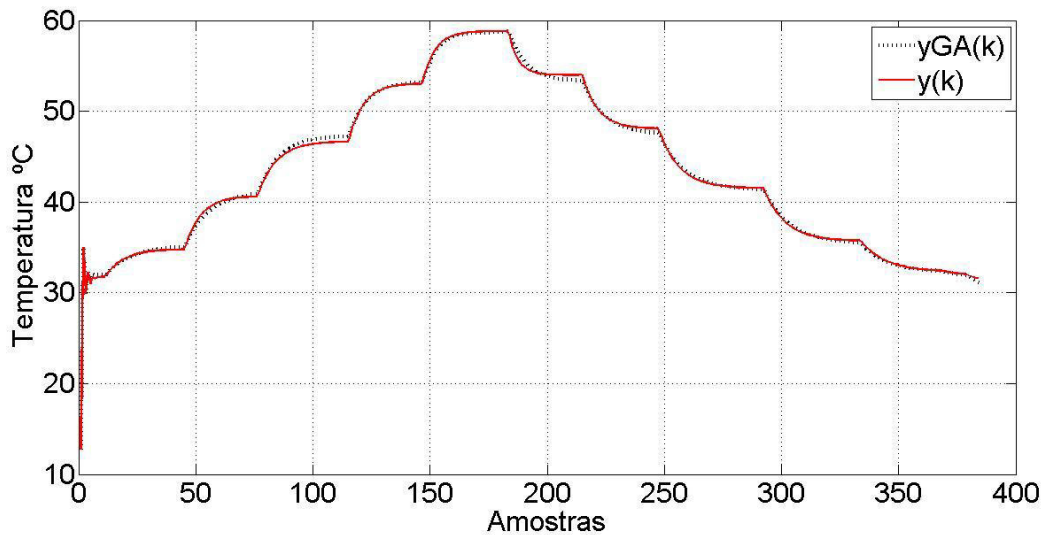
para 0,067, uma redução de aproximadamente 10%, o que se pode considerar uma melhoria muito pequena frente à quantidade de gerações. Assim, optou-se por parar na geração de número 240 e escolher o melhor modelo desta geração como o modelo final.

O melhor modelo obtido com o algoritmo genético é mostrado na equação 5.2.

$$\begin{aligned} y(k) = & 0,89015y(k-1) + 0,1408y(k-2) + 0,0469u(k-1) + 2,359 \\ & + 0,0038u(k-1)^2 + 0,0011y(k-3)u(k-1) + 0,00006u(k-2)^3 \\ & - 0,0001y(k-3)u(k-1)u(k-2) - 0,1106y(k-3) - 0,0139u(k-3) \end{aligned} \quad (5.2)$$

O modelo da equação 5.2 foi validado utilizando os dados reais da célula a combustível e o resultado é apresentado na figura 26.

Figura 26 - Validação do modelo da equação 5.2.

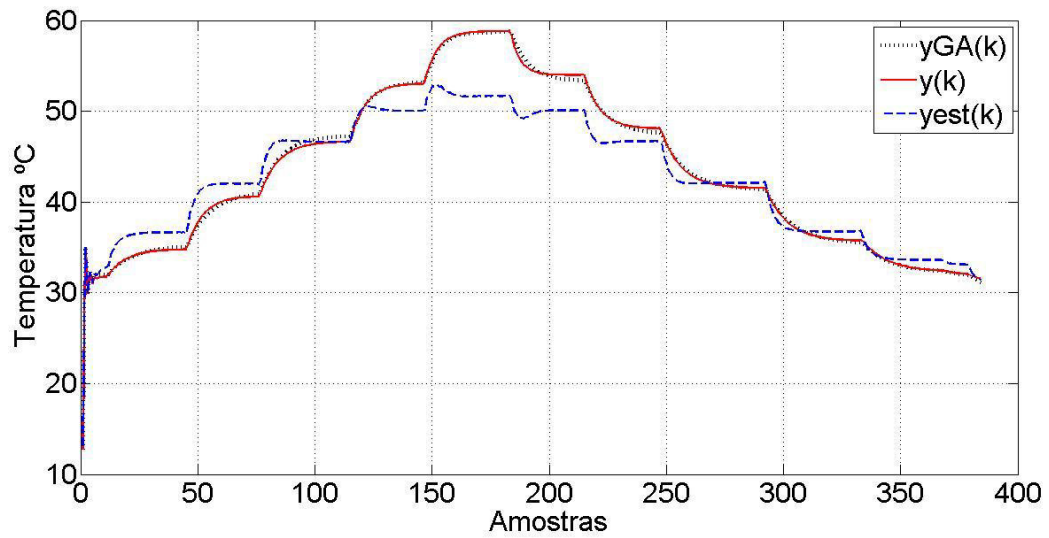


Fonte: Elaboração própria.

Na figura 26 $y(k)$ representa os dados originais provenientes da célula a combustível e $yGA(k)$ representa o modelo obtido com o uso de algoritmos genéticos.

Observa-se que o novo modelo encontrado, utilizando algoritmos genéticos, conseguiu representar o modelo de temperatura da célula melhor do que o modelo obtido pelo método de escolha do tempo de amostragem por análise de correlação, a figura 27 mostra o teste de validação onde é possível comparar o modelo da equação 5.1 com o modelo da equação 5.2.

Figura 27 - Comparação entre os modelos da equação 5.1 e 5.2 em relação aos dados de validação.



Fonte: Elaboração própria.

Na figura 27 $y(k)$ representa os dados reais provenientes da célula a combustível, e $y_{est}(k)$ os dados do modelo da equação 5.1 e $y_{GA}(k)$ os dados do modelo da equação 5.2.

Por meio da figura 27 nota-se claramente uma melhora no desempenho do modelo da equação 5.2 em relação ao modelo da equação 5.1, mostrando que o uso de algoritmos genéticos para seleção da taxa de amostragem pode garantir uma melhora na qualidade do modelo final.

6 Conclusão

O presente trabalho abordou o problema de seleção da taxa de amostragem por meio de algoritmos genéticos aplicados na determinação da melhor taxa de decimação de um sinal superamostrado.

Foi mostrado também que a escolha de um tempo de amostragem pequeno pode impossibilitar o processo de identificação de sistemas e a escolha de um tempo grande pode descharacterizar o sinal de forma que informações importantes a respeito do sinal sejam tiradas dos dados.

O método de análise de correlação para a escolha do tempo de amostragem já consagrado na literatura, garante uma faixa de taxas de decimação que podem fazer com que o processo de identificação resulte em modelos que não descrevem bem a dinâmica do sistema a ser identificado.

Assim, foi apresentada uma abordagem utilizando algoritmos genéticos para selecionar uma taxa de decimação de um sinal superamostrado, resultando em um tempo de amostragem específico.

A escolha da taxa de decimação por algoritmos genéticos apresentou um desempenho melhor do que o método de análise de correlação dos sinais, garantindo assim uma alternativa para a seleção da taxa de amostragem de sinais no processo de identificação de sistemas.

Outro ponto a se destacar no presente trabalho foi a obtenção de um modelo NAR-MAX de temperatura em função da corrente de uma célula a combustível de membrana polimérica. Há de se destacar a importância de um modelo para a célula a combustível, a fim de auxiliar na compreensão de tal equipamento, evitando os perigos inerentes à célula, bem como diminuindo gastos e o tempo dos testes e recolhimento de dados.

Como trabalho futuros sugere-se a combinação de programação evolutiva nos processos de seleção da taxa de amostragem e seleção da estrutura, os quais já apresentam diversos estudos e também a utilização de programação distribuída como clusters para aumentar a velocidade de processamento dos algoritmos.

REFERÊNCIAS

AGUIRRE, L. A. **Introdução à Identificação de sistemas: Técnicas lineares e não lineares aplicadas a sistemas reais**. Belo Horizonte: UFMG: 722 p. 2007a.

AGUIRRE, L. A.; BILLINGS, S. A. Dynamical effects of overparametrization in nonlinear models. **Physica D: Nonlinear Phenomena**, v. 80, n. 1-2, p. 26-40, 1 January 1995a.

AVELAR, H. J. **Estudo e desenvolvimento de um sistema de energia baseado em célula a combustível para injeção de potência na rede elétrica**. 2012. (Doutorado). Universidade Federal de Uberlândia

AVELAR, H. J. et al. **PEM fuel cell dynamic model for electronic circuit simulator** Electrical Power & Energy Conference (EPEC), 2009 IEEE: 1-6 p. 2009.

BÄCK, T.; FOGEL, D. B.; MICHALEWICS, Z. **Handbook of evolutionary computation**. Bristol ; Philadelphia New York: Institute of Physics Pub. ;Oxford University Press, 1997.

BAI, Y.; ZHU, Y.; JIANG, Y. **A New Nonlinear System Identification Method Using Gene Expression Programming**: 2951-2956 p. 2007.

BILLINGS, S. A. **NONLINEAR SYSTEM IDENTIFICATION NARMAX: Methods in the Time, Frequency, and Spatio-Temporal Domains**: 574 pages p. 2013.

BILLINGS, S. A.; AGUIRRE, L. A. Effects of the Sampling Time on the Dynamics and Identification of Nonlinear Models. **International Journal of Bifurcation and Chaos**, v. 05, n. 06, p. 1541-1556, 1995b. ISSN 0218-1274 1793-6551.

BILLINGS, S. A.; ZHU, Q. M. Nonlinear model validation using correlation tests. **International Journal of Control**, v. 60, n. 6, p. 1107-1120, 1994. ISSN 0020-7179 1366-5820.

CAMASCA, C.; SWAIN, A. K.; PATEL, N. D. **Intelligent Structure Selection of Polynomial Nonlinear Systems using Evolutionary Programming**: 1-6 p. 2006.

CASSINI, C. C. D. S. **Estimação Recursiva de Características Estáticas Não Lineares Utilizando Modelos Polinomiais NARMAX**. 1999. 154 p. (Dissertação de Mestrado). UFMG, Belo Horizonte.

COELHO, L. D. S.; COELHO, A. A. R. **ALGORITMOS EVOLUTIVOS EM IDENTIFICAÇÃO E CONTROLE DE PROCESSOS: UMA VISÃO INTEGRADA E PERSPECTIVAS**. **SBA Controle & Automação** v. Vol. 10 p. 18, 1999.

CORREA, M. V. **Identificação de Sistemas Dinâmicos Não-Lineares Utilizando Modelos NARMAX Racionais - Aplicação a Sistemas Reais**. 1997. 110 p. (Dissertação (Mestre)). Universidade Federal de Minas Gerais, Belo Horizonte.

GOLDBERG, D. E. **Genetic algorithms in search, optimization, and machine learning**. Reading, Mass.: Addison-Wesley Pub. Co., 1989. xiii, 412 p. ISBN 0201157675.

GOLDBERG, D. E. Genetic and evolutionary algorithms come of age. **Communications of the ACM**, v. 37, n. 3, p. 113-119, 1994. ISSN 00010782.

GOLDBERG, D. E.; KORB, B.; DEB, K. Messy Genetic Algorithms: Motivation, Analysis, and First Results. **Complex Systems**, v. 3, p. 493-530, 1989b.

HELENE, O. **Método dos Mínimos Quadrados com Formalismo Matricial**. São Paulo: Editora Livraria da Física, 2006. 102.

HORÁČEK, P. et al. **On identification of a flexible mechanical system using decimated data**. p. 12-12, 2005. ISSN 1474-6670.

JANIKOW, C. Z.; MICHALEWICZ, Z. **A specialized genetic algorithm for numerical optimization problems**: 798-804 p. 1990.

JØRGENSEN, S. B.; HANGOS, K. M. Grey box modelling for control: Qualitative models as a unifying framework. **International Journal of Adaptive Control and Signal Processing**, v. 9, n. 6, p. 547-562, 1995. ISSN 08906327
10991115.

KORENBERG, M. et al. Orthogonal parameter estimation algorithm for non-linear stochastic systems. **International Journal of Control**, v. 48, n. 1, p. 193-210, 1988. ISSN 0020-7179
1366-5820.

LEONTARITIS, I. J.; BILLINGS, S. A. **Input-Output Parametric Models for Nonlinear Systems**. Part 1- Deterministic Nonlinear Systems. Sheffield: University of Sheffield 1984.

LEONTARITIS, I. J.; BILLINGS, S. A. Experimental design and identifiability for non-linear systems. **International Journal of Systems Science**, v. 18, n. 1, p. 189-202, 1987. ISSN 0020-7721
1464-5319.

LJUNG, L. **System identification : theory for the user**. Upper Saddle River, NJ: Prentice Hall PTR: xxii, 609 p. p. 1999.

MALAQUIAS, N. G. L. **Uso dos Algoritmos Genéticos para a Otimização de Rotas de Distribuição**. 2006. 97 p. (Dissertação de Mestrado). Engenharia Elétrica, Universidade Federal de Uberlândia, Uberlândia.

MELANIE, M. **An Introduction to Genetic Algorithms**. London: MIT Press, 1999. 162 p.

MOHAMED VALL, O. M.; M'HIRI, R. An approach to polynomial NARX/NARMAX systems identification in a closed-loop with variable structure control. **International Journal of Automation and Computing**, v. 5, n. 3, p. 313-318, 2008. ISSN 1476-8186 1751-8520.

MORAIS, J. S. D. **Algoritmo Diferencial Evolutivo Aplicado À Identificação de Sistemas Dinâmicos Não-Lineares**. 2013. 127 p. (Tese de Doutorado). Universidade Federal de Uberlândia, Uberlândia.

NELLES, O. **Nonlinear System Identification**. Berlin: Springer-Verlag Berlin Heidelberg, 2001. 786 p. ISBN 978-3-540-67369-9.

NYQUIST, H. Certain Topics in Telegraph Transmission Theory. **Transactions of the American Institute of Electrical Engineers**, v. 47, n. 2, p. 617-644, 1928. ISSN 0096-3860.

PETROLI, S. Computação Evolutiva: desvendando os algoritmos genéticos **Ubiquidade**, v. Vol 1, 2011. ISSN 2236-9031.

PU, H.; SHILIANG, Z.; DONGFENG, W. **A Multi-objective Genetic Programming/NARMAX Approach to Chaotic Systems Identification**: 1735-1739 p. 2006.

RADCLIFFE, N. J. Formal Analysis and Random Respectful Recombination. **Proc. 4th Int. Conf. on Genetic Algorithms**, 1991.

RODGERS, J. L.; NICEWANDER, W. A.; TOOTHAKER, L. Linearly Independent, Orthogonal, and Uncorrelated Variables. **The American Statistician**, v. 38, n. 2, p. 133-134, 1984. ISSN 0003-1305 1537-2731.

RODRIGUES, G. G. **Identificação de sistemas dinâmicos não-lineares utilizando modelos NARMAX polinomiais**: aplicação a sistemas reais. Belo Horizonte: Universidade Federal de Minas Gerais. Dissertação (Dissertação de Mestrado): 105 p. p. 1996.

RODRIGUEZ-VAZQUEZ, K.; FLEMING, P. J. **A genetic programming/NARMAX approach to nonlinear system identification**. Genetic Algorithms in Engineering Systems: Innovations and Applications, 1997. GALEZIA 97. Second International Conference On (Conf. Publ. No. 446), 1997, 2-4 Sep 1997. p.409-414.

SCHWARZ, G. Estimating the Dimension of a Model. **The Annals of Statistics**, v. 6, n. 2, p. 461-464, 1978. ISSN 0090-5364.

SHANNON, C. E. Communication in the Presence of Noise. **Proceedings of the IRE**, v. 37, n. 1, p. 10-21, 1949. ISSN 0096-8390.

SINHA, N. K.; PUTHENPURA, S. Choice of the sampling interval for the identification of continuous-time systems from samples of input/output data. **IEE Proceedings D Control Theory and Applications**, v. 132, n. 6, p. 263, 1985. ISSN 01437054.

SIVANANDAM, S. N.; DEEPA, S. N. **Introduction to Genetic Algorithms**. New York: Springer. 1: 453 p. 2008.

TANOMARU, J. Motivação, Fundamentos e Aplicações de Algoritmos Genéticos. **II Congresso Brasileiro de Redes Neurais**, p. p. 373-403, 1995.

ÜNAL, M. et al. **Optimization of PID Controllers Using Ant Colony and Genetic Algorithms**. 449: 88 p. 2013.

WEI, H. L. Prediction of theDstindex using multiresolution wavelet models. **Journal of Geophysical Research**, v. 109, n. A7, 2004. ISSN 0148-0227.

WRIGHT, A. H. Genetic Algorithms for Real Parameter Optimization. **J. E. Rawlins Foundations of Genetic Algorithms**, p. 205-218, 1991.

XIN, J.; TSUJI, H.; SANO, A. Optimal sampling interval for system identification based on decimation and interpolation. **IEE Proceedings - Control Theory and Applications**, v. 142, n. 1, p. 15-22, 1995. ISSN 1350-2379
1359-7035.

YUAN, X.-L.; BAI, Y. Identifying Stochastic Nonlinear Dynamic Systems Using Multi-Objective Hierarchical Fair Competition Parallel Genetic. **Multiple-Valued Logic and Soft Computing**, v. 16, p. 643-660, 2010. ISSN 1542-3980.

APÊNDICE A – Código para gerar o sinal de entrada que foi aplicado à célula a combustível.

```
%Gera sinal de entrada para identificação de sistemas não lineares
%Código desenvolvido por Luis Paulo Fagundes
%Data: 05/03/2015

z=1;                %valor inicial do sinal de entrada;
k=1;                %passo inicial
amostras=input('Entre com o tamanho das amostras: ');      %o tamanho de
                                                             %38321 foi escolhido arbitrariamente.
amplitudeMedia=input('Entre com o valor da amplitude média: '); %o
                                                             tamanho de %38321 foi escolhido arbitra-
                                                             riamente.

while (size(z)<amostras)                                     %Inicialização do loop.

    amplitude=rand();                                       %Inicia a amplitude com um valor randômico;

    passos=randi([500,1500],1,1); %Define o intervalo em que o sinal irá
                                %permanecer em determinado valor;
    ponto_final=passos+k;
    for j=k:(ponto_final)
        z(j)=amplitude* amplitudeMedia;                  %A amplitude é múltiplo de
30, esse                                                    valor foi escolhido em função do sistema
                                                            que se trabalho.

    end
    k=(ponto_final+1);
end
```

APÊNDICE B – Termos candidatos para a identificação do modelo de temperatura da célula a combustível.

Termos de primeiro grau	Termos de Segundo Grau	Termos de terceiro grau
$u(k-1)$	$u(k-1)^2$	$u(k-1)^3$
$u(k-2)$	$u(k-1)*u(k-2)$	$u(k-1)^2*u(k-2)$
$u(k-3)$	$u(k-1)*u(k-3)$	$u(k-1)^2*u(k-3)$
$y(k-1)$	$u(k-2)^2$	$u(k-1)*u(k-2)^2$
$y(k-2)$	$u(k-2)*u(k-3)$	$u(k-1)*u(k-2)*u(k-3)$
$y(k-3)$	$u(k-3)^2$	$u(k-1)*u(k-3)^2$
	$y(k-1)*u(k-1)$	$u(k-2)^3$
	$y(k-1)*u(k-2)$	$u(k-2)^2*u(k-3)$
	$y(k-1)*u(k-3)$	$u(k-2)*u(k-3)^2$
	$y(k-2)*u(k-1)$	$u(k-3)^3$
	$y(k-2)*u(k-2)$	$y(k-1)*u(k-1)^2$
	$y(k-2)*u(k-3)$	$y(k-1)*u(k-1)*u(k-2)$
	$y(k-3)*u(k-1)$	$y(k-1)*u(k-1)*u(k-3)$
	$y(k-3)*u(k-2)$	$y(k-1)*u(k-2)^2$
	$y(k-3)*u(k-3)$	$y(k-1)*u(k-2)*u(k-3)$
	$y(k-1)^2$	$y(k-1)*u(k-3)^2$
	$y(k-1)*y(k-2)$	$y(k-2)*u(k-1)^2$
	$y(k-1)*y(k-3)$	$y(k-2)*u(k-1)*u(k-2)$
	$y(k-2)^2$	$y(k-2)*u(k-1)*u(k-3)$
	$y(k-2)*y(k-3)$	$y(k-2)*u(k-2)^2$
	$y(k-3)^2$	$y(k-2)*u(k-2)*u(k-3)$
		$y(k-2)*u(k-3)^2$
		$y(k-3)*u(k-1)^2$
		$y(k-3)*u(k-1)*u(k-2)$
		$y(k-3)*u(k-1)*u(k-3)$
		$y(k-3)*u(k-2)^2$
		$y(k-3)*u(k-2)*u(k-3)$
		$y(k-3)*u(k-3)^2$
		$y(k-1)^2*u(k-1)$
		$y(k-1)^2*u(k-2)$
		$y(k-1)^2*u(k-3)$
		$y(k-1)*y(k-2)*u(k-1)$
		$y(k-1)*y(k-2)*u(k-2)$
		$y(k-1)*y(k-2)*u(k-3)$
		$y(k-1)*y(k-3)*u(k-1)$
		$y(k-1)*y(k-3)*u(k-2)$
		$y(k-1)*y(k-3)*u(k-3)$
		$y(k-2)^2*u(k-1)$
		$y(k-2)^2*u(k-2)$
		$y(k-2)^2*u(k-3)$
		$y(k-2)*y(k-3)*u(k-1)$
		$y(k-2)*y(k-3)*u(k-2)$
		$y(k-2)*y(k-3)*u(k-3)$
		$y(k-3)^2*u(k-1)$
		$y(k-3)^2*u(k-2)$
		$y(k-3)^2*u(k-3)$
		$y(k-1)^3$
		$y(k-1)^2*y(k-2)$
		$y(k-1)^2*y(k-3)$
		$y(k-1)*y(k-2)^2$
		$y(k-1)*y(k-2)*y(k-3)$
		$y(k-1)*y(k-3)^2$
		$y(k-2)^3$
		$y(k-2)^2*y(k-3)$
		$y(k-2)*y(k-3)^2$
		$y(k-3)^3$

APÊNDICE C- Software desenvolvido na plataforma MATLAB®

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Função responsável por calcular o fitness de cada indivíduo, o erro
%quadrático médio e a equação em forma de string.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [fitness,erro,equation]= fitness_function(Tx1,Tx2,Nr_Termos);

erro_quadratico=0;
equacao=''; %Equação do modelo que será montada
load identificacao.dat; %Carrega os dados de identificação
dados=identificacao;
u_n_decimado=dados(:,2); %coluna um é os dados de entrada

u_tx1=decimate(u_n_decimado,Tx1); %O sinal de entrada é decimado com a
taxa Tx1 que será usado na seleção da estrutura

u_tx2=decimate(u_n_decimado,Tx2); %O sinal de entrada é decimado com a taxa
Tx2 que será usado na estimação dos parâmetros

y_n_decimado=dados(:,1); %coluna dois é os dados de saída
y_tx1=decimate(y_n_decimado,Tx1); %O sinal de saída é decimado com a taxa
Tx2 que será usado na seleção da estrutura

y_tx2=decimate(y_n_decimado,Tx2); %O sinal de entrada é decimado com a taxa
Tx2 que será usado na estimação dos parâmetros

m=size(y_tx1);
yest0=y_tx1; %Faz-se yest igual a y para poder pegar as 22 primeiras
amostras como estados inicial

%Montar o banco dos regressores

Regressores = ['1
                'u(k-1)
                'u(k-2)
                'u(k-3)
                'yest0(k-1)
                'yest0(k-2)
                'yest0(k-3)
                'u(k-1).^2
                'u(k-1)*u(k-2)
                'u(k-1)*u(k-3)
                'u(k-2).^2
                'u(k-2)*u(k-3)
                'u(k-3).^2
                'yest0(k-1)*u(k-1)
                'yest0(k-1)*u(k-2)
                'yest0(k-1)*u(k-3)
                'yest0(k-2)*u(k-1)
                'yest0(k-2)*u(k-2)
                'yest0(k-2)*u(k-3)
                'yest0(k-3)*u(k-1)
                'yest0(k-3)*u(k-2)
                'yest0(k-3)*u(k-3)
                '];

```

```

'yest0(k-1).^2';
'yest0(k-1)*yest0(k-2)';
'yest0(k-1)*yest0(k-3)';
'yest0(k-2).^2';
'yest0(k-2)*yest0(k-3)';
'yest0(k-3).^2';
'u(k-1).^3';
'u(k-1).^2*u(k-2)';
'u(k-1).^2*u(k-3)';
'u(k-1)*u(k-2).^2';
'u(k-1)*u(k-2)*u(k-3)';
'u(k-1)*u(k-3).^2';
'u(k-2).^3';
'u(k-2).^2*u(k-3)';
'u(k-2)*u(k-3).^2';
'u(k-3).^3';
'yest0(k-1)*u(k-1).^2';
'yest0(k-1)*u(k-1)*u(k-2)';
'yest0(k-1)*u(k-1)*u(k-3)';
'yest0(k-1)*u(k-2).^2';
'yest0(k-1)*u(k-2)*u(k-3)';
'yest0(k-1)*u(k-3).^2';
'yest0(k-2)*u(k-1).^2';
'yest0(k-2)*u(k-1)*u(k-2)';
'yest0(k-2)*u(k-1)*u(k-3)';
'yest0(k-2)*u(k-2).^2';
'yest0(k-2)*u(k-2)*u(k-3)';
'yest0(k-2)*u(k-3).^2';
'yest0(k-3)*u(k-1).^2';
'yest0(k-3)*u(k-1)*u(k-2)';
'yest0(k-3)*u(k-1)*u(k-3)';
'yest0(k-3)*u(k-2).^2';
'yest0(k-3)*u(k-2)*u(k-3)';
'yest0(k-3)*u(k-3).^2';
'yest0(k-1).^2*u(k-1)';
'yest0(k-1).^2*u(k-2)';
'yest0(k-1).^2*u(k-3)';
'yest0(k-1)*yest0(k-2)*u(k-1)';
'yest0(k-1)*yest0(k-2)*u(k-2)';
'yest0(k-1)*yest0(k-2)*u(k-3)';
'yest0(k-1)*yest0(k-3)*u(k-1)';
'yest0(k-1)*yest0(k-3)*u(k-2)';
'yest0(k-1)*yest0(k-3)*u(k-3)';
'yest0(k-2).^2*u(k-1)';
'yest0(k-2).^2*u(k-2)';
'yest0(k-2).^2*u(k-3)';
'yest0(k-2)*yest0(k-3)*u(k-1)';
'yest0(k-2)*yest0(k-3)*u(k-2)';
'yest0(k-2)*yest0(k-3)*u(k-3)';
'yest0(k-3).^2*u(k-1)';
'yest0(k-3).^2*u(k-2)';
'yest0(k-3).^2*u(k-3)';
'yest0(k-1).^3';
'yest0(k-1).^2*yest0(k-2)';
'yest0(k-1).^2*yest0(k-3)';
'yest0(k-1)*yest0(k-2).^2';
'yest0(k-1)*yest0(k-2)*yest0(k-3)';
'yest0(k-1)*yest0(k-3).^2';
'yest0(k-2).^3';
'yest0(k-2).^2*yest0(k-3)';
'yest0(k-2)*yest0(k-3).^2';

```

```

                                'yest0(k-3).^3                                '];

% Montar matriz de regressores
reg1=ones((m-3),1); % constante
reg2=u_tx1(3:(m-1)); % u(k-1)
reg3=u_tx1(2:(m-2)); % u(k-2)
reg4=u_tx1(1:(m-3)); % u(k-3)
reg5=y_tx1(3:(m-1)); % y(k-1)
reg6=y_tx1(2:(m-2)); % y(k-2)
reg7=y_tx1(1:(m-3)); % y(k-3)
reg8=reg2.^2; % u(k-1).^2
reg9=reg2.*reg3; % u(k-1)*u(k-2)
reg10=reg2.*reg4; % u(k-1)*u(k-3)
reg11=reg3.^2; % u(k-2).^2
reg12=reg3.*reg4; % u(k-2)*u(k-3)
reg13=reg4.^2; % u(k-3).^2
reg14=reg5.*reg2; % y(k-1)*u(k-1)
reg15=reg5.*reg3; % y(k-1)*u(k-2)
reg16=reg5.*reg4; % y(k-1)*u(k-3)
reg17=reg6.*reg2; % y(k-2)*u(k-1)
reg18=reg6.*reg3; % y(k-2)*u(k-2)
reg19=reg6.*reg4; % y(k-2)*u(k-3)
reg20=reg7.*reg2; % y(k-3)*u(k-1)
reg21=reg7.*reg3; % y(k-3)*u(k-2)
reg22=reg7.*reg4; % y(k-3)*u(k-3)
reg23=reg5.^2; % y(k-1).^2
reg24=reg5.*reg6; % y(k-1)*y(k-2)
reg25=reg5.*reg7; % y(k-1)*y(k-3)
reg26=reg6.^2; % y(k-2).^2
reg27=reg6.*reg7; % y(k-2)*y(k-3)
reg28=reg7.^3; % y(k-3).^2
reg29=reg2.^2; % u(k-1).^3
reg30=(reg2.^2).*reg3; % u(k-1).^2*u(k-2)
reg31=(reg2.^2).*reg4; % u(k-1).^2*u(k-3)
reg32=reg2.*reg3.^2; % u(k-1)*u(k-2).^2
reg33=reg2.*reg3.*reg4; % u(k-1)*u(k-2)*u(k-3)
reg34=reg2.*reg4.^2; % u(k-1)*u(k-3).^2
reg35=reg3.^3; % u(k-2).^3
reg36=(reg3.^2).*reg4; % u(k-2).^2*u(k-3)
reg37=reg3.*reg4.^2; % u(k-2)*u(k-3).^2
reg38=reg4.^3; % u(k-3).^3
reg39=reg5.*reg2.^2; % y(k-1)*u(k-1).^2
reg40=reg5.*reg2.*reg3; % y(k-1)*u(k-1)*u(k-2)
reg41=reg5.*reg2.*reg4; % y(k-1)*u(k-1)*u(k-3)
reg42=reg5.*reg3.^2; % y(k-1)*u(k-2).^2
reg43=reg5.*reg3.*reg4; % y(k-1)*u(k-2)*u(k-3)
reg44=reg5.*reg4.^2; % y(k-1)*u(k-3).^2
reg45=reg6.*reg2.^2; % y(k-2)*u(k-1).^2
reg46=reg6.*reg2.*reg3; % y(k-2)*u(k-1)*u(k-2)
reg47=reg6.*reg2.*reg4; % y(k-2)*u(k-1)*u(k-3)
reg48=reg6.*reg3.^2; % y(k-2)*u(k-2).^2
reg49=reg6.*reg3.*reg4; % y(k-2)*u(k-2)*u(k-3)
reg50=reg6.*reg4.^2; % y(k-2)*u(k-3).^2
reg51=reg7.*reg2.^2; % y(k-3)*u(k-1).^2
reg52=reg7.*reg2.*reg3; % y(k-3)*u(k-1)*u(k-2)
reg53=reg7.*reg2.*reg4; % y(k-3)*u(k-1)*u(k-3)
reg54=reg7.*reg3.^2; % y(k-3)*u(k-2).^2
reg55=reg7.*reg3.*reg4; % y(k-3)*u(k-2)*u(k-3)
reg56=reg7.*reg4.^2; % y(k-3)*u(k-3).^2
reg57=(reg5.^2).*reg2; % y(k-1).^2*u(k-1)
reg58=(reg5.^2).*reg3; % y(k-1).^2*u(k-2)

```

```

reg59=(reg5.^2).*reg4; % y(k-1).^2*u(k-3)
reg60=reg5.*reg6.*reg2; % y(k-1)*y(k-2)*u(k-1)
reg61=reg5.*reg6.*reg3; % y(k-1)*y(k-2)*u(k-2)
reg62=reg5.*reg6.*reg4; % y(k-1)*y(k-2)*u(k-3)
reg63=reg5.*reg7.*reg2; % y(k-1)*y(k-3)*u(k-1)
reg64=reg5.*reg7.*reg3; % y(k-1)*y(k-3)*u(k-2)
reg65=reg5.*reg7.*reg4; % y(k-1)*y(k-3)*u(k-3)
reg66=(reg6.^2).*reg2; % y(k-2).^2*u(k-1)
reg67=(reg6.^2).*reg3; % y(k-2).^2*u(k-2)
reg68=(reg6.^2).*reg4; % y(k-2).^2*u(k-3)
reg69=reg6.*reg7.*reg2; % y(k-2)*y(k-3)*u(k-1)
reg70=reg6.*reg7.*reg3; % y(k-2)*y(k-3)*u(k-2)
reg71=reg6.*reg7.*reg4; % y(k-2)*y(k-3)*u(k-3)
reg72=(reg7.^2).*reg2; % y(k-3).^2*u(k-1)
reg73=(reg7.^2).*reg3; % y(k-3).^2*u(k-2)
reg74=(reg7.^2).*reg4; % y(k-3).^2*u(k-3)
reg75=(reg5.^3); % y(k-1).^3
reg76=(reg5.^2).*reg6; % y(k-1).^2*y(k-2)
reg77=(reg5.^2).*reg7; % y(k-1).^2*y(k-3)
reg78=reg5.*reg6.^2; % y(k-1)*y(k-2).^2
reg79=reg5.*reg6.*reg7; % y(k-1)*y(k-2)*y(k-3)
reg80=reg5.*reg7.^2; % y(k-1)*y(k-3).^2
reg81=reg6.^3; % y(k-2).^3
reg82=(reg6.^2).*reg7; % y(k-2).^2*y(k-3)
reg83=reg6.*reg7.^2; % y(k-2)*y(k-3).^2
reg84=reg7.^3; % y(k-3).^3

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Determinação da Estrutura %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

psi=[reg1 reg2 reg3 reg4 reg5 reg6 reg7 reg8 reg9 reg10 reg11 reg12 reg13
reg14 reg15 reg16 reg17 reg18 reg19 reg20 reg21 reg22 reg23 reg24 reg25
reg26 reg27 reg28 reg29 reg30 reg31 reg32 reg33 reg34 reg35 reg36 reg37
reg38 reg39 reg40 reg41 reg42 reg43 reg44 reg45 reg46 reg47 reg48 reg49
reg50 reg51 reg52 reg53 reg54 reg55 reg56 reg57 reg58 reg59 reg60 reg61
reg62 reg63 reg64 reg65 reg66 reg67 reg68 reg69 reg70 reg71 reg72 reg73
reg74 reg75 reg76 reg77 reg78 reg79 reg80 reg81 reg82 reg83 reg84];
vec=y_tx1(4:m); %define vec como as amostras que se deseja identificar do
vetor de medidas
Psi=[psi vec]; %De acordo com o algoritmo de Golub-Householder (Aguirre
página 430), monta-se a matriz PSI como sendo a matriz de regressores e
a ultima coluna com y

[A,err,piv]=myhouse(Psi,84); %Função que calcula a taxa de redução do
erro, define o vetor piv que contém quais
colunas dos regressores serão escolhidas
e faz a transformação de householder sobre
a matriz A (Obtida juntamente com o livro
Introdução a identificação de sistemas -
Luis A. Aguirre 2007).

m=size(y_tx2);

% Montar matriz de regressores
reg1=ones((m-3),1); % constante
reg2=u_tx2(3:(m-1)); % u(k-1)
reg3=u_tx2(2:(m-2)); % u(k-2)

```

```

reg4=u_tx2(1:(m-3));      % u(k-3)
reg5=y_tx2(3:(m-1));      % y(k-1)
reg6=y_tx2(2:(m-2));      % y(k-2)
reg7=y_tx2(1:(m-3));      % y(k-3)
reg8=reg2.^2;             % u(k-1).^2
reg9=reg2.*reg3;          % u(k-1)*u(k-2)
reg10=reg2.*reg4;         % u(k-1)*u(k-3)
reg11=reg3.^2;            % u(k-2).^2
reg12=reg3.*reg4;         % u(k-2)*u(k-3)
reg13=reg4.^2;            % u(k-3).^2
reg14=reg5.*reg2;         % y(k-1)*u(k-1)
reg15=reg5.*reg3;         % y(k-1)*u(k-2)
reg16=reg5.*reg4;         % y(k-1)*u(k-3)
reg17=reg6.*reg2;         % y(k-2)*u(k-1)
reg18=reg6.*reg3;         % y(k-2)*u(k-2)
reg19=reg6.*reg4;         % y(k-2)*u(k-3)
reg20=reg7.*reg2;         % y(k-3)*u(k-1)
reg21=reg7.*reg3;         % y(k-3)*u(k-2)
reg22=reg7.*reg4;         % y(k-3)*u(k-3)
reg23=reg5.^2;            % y(k-1).^2
reg24=reg5.*reg6;         % y(k-1)*y(k-2)
reg25=reg5.*reg7;         % y(k-1)*y(k-3)
reg26=reg6.^2;            % y(k-2).^2
reg27=reg6.*reg7;         % y(k-2)*y(k-3)
reg28=reg7.^3;            % y(k-3).^2
reg29=reg2.^2;            % u(k-1).^2
reg30=(reg2.^2).*reg3;    % u(k-1).^2*u(k-2)
reg31=(reg2.^2).*reg4;    % u(k-1).^2*u(k-3)
reg32=reg2.*reg3.^2;      % u(k-1)*u(k-2).^2
reg33=reg2.*reg3.*reg4;   % u(k-1)*u(k-2)*u(k-3)
reg34=reg2.*reg4.^2;      % u(k-1)*u(k-3).^2
reg35=reg3.^3;            % u(k-2).^3
reg36=(reg3.^2).*reg4;    % u(k-2).^2*u(k-3)
reg37=reg3.*reg4.^2;      % u(k-2)*u(k-3).^2
reg38=reg4.^3;            % u(k-3).^3
reg39=reg5.*reg2.^2;      % y(k-1)*u(k-1).^2
reg40=reg5.*reg2.*reg3;   % y(k-1)*u(k-1)*u(k-2)
reg41=reg5.*reg2.*reg4;   % y(k-1)*u(k-1)*u(k-3)
reg42=reg5.*reg3.^2;      % y(k-1)*u(k-2).^2
reg43=reg5.*reg3.*reg4;   % y(k-1)*u(k-2)*u(k-3)
reg44=reg5.*reg4.^2;      % y(k-1)*u(k-3).^2
reg45=reg6.*reg2.^2;      % y(k-2)*u(k-1).^2
reg46=reg6.*reg2.*reg3;   % y(k-2)*u(k-1)*u(k-2)
reg47=reg6.*reg2.*reg4;   % y(k-2)*u(k-1)*u(k-3)
reg48=reg6.*reg3.^2;      % y(k-2)*u(k-2).^2
reg49=reg6.*reg3.*reg4;   % y(k-2)*u(k-2)*u(k-3)
reg50=reg6.*reg4.^2;      % y(k-2)*u(k-3).^2
reg51=reg7.*reg2.^2;      % y(k-3)*u(k-1).^2
reg52=reg7.*reg2.*reg3;   % y(k-3)*u(k-1)*u(k-2)
reg53=reg7.*reg2.*reg4;   % y(k-3)*u(k-1)*u(k-3)
reg54=reg7.*reg3.^2;      % y(k-3)*u(k-2).^2
reg55=reg7.*reg3.*reg4;   % y(k-3)*u(k-2)*u(k-3)
reg56=reg7.*reg4.^2;      % y(k-3)*u(k-3).^2
reg57=(reg5.^2).*reg2;     % y(k-1).^2*u(k-1)
reg58=(reg5.^2).*reg3;     % y(k-1).^2*u(k-2)
reg59=(reg5.^2).*reg4;     % y(k-1).^2*u(k-3)
reg60=reg5.*reg6.*reg2;   % y(k-1)*y(k-2)*u(k-1)
reg61=reg5.*reg6.*reg3;   % y(k-1)*y(k-2)*u(k-2)
reg62=reg5.*reg6.*reg4;   % y(k-1)*y(k-2)*u(k-3)
reg63=reg5.*reg7.*reg2;   % y(k-1)*y(k-3)*u(k-1)
reg64=reg5.*reg7.*reg3;   % y(k-1)*y(k-3)*u(k-2)

```

```

reg65=reg5.*reg7.*reg4; % y(k-1)*y(k-3)*u(k-3)
reg66=(reg6.^2).*reg2; % y(k-2).^2*u(k-1)
reg67=(reg6.^2).*reg3; % y(k-2).^2*u(k-2)
reg68=(reg6.^2).*reg4; % y(k-2).^2*u(k-3)
reg69=reg6.*reg7.*reg2; % y(k-2)*y(k-3)*u(k-1)
reg70=reg6.*reg7.*reg3; % y(k-2)*y(k-3)*u(k-2)
reg71=reg6.*reg7.*reg4; % y(k-2)*y(k-3)*u(k-3)
reg72=(reg7.^2).*reg2; % y(k-3).^2*u(k-1)
reg73=(reg7.^2).*reg3; % y(k-3).^2*u(k-2)
reg74=(reg7.^2).*reg4; % y(k-3).^2*u(k-3)
reg75=(reg5.^3); % y(k-1).^3
reg76=(reg5.^2).*reg6; % y(k-1).^2*y(k-2)
reg77=(reg5.^2).*reg7; % y(k-1).^2*y(k-3)
reg78=reg5.*reg6.^2; % y(k-1)*y(k-2).^2
reg79=reg5.*reg6.*reg7; % y(k-1)*y(k-2)*y(k-3)
reg80=reg5.*reg7.^2; % y(k-1)*y(k-3).^2
reg81=reg6.^3; % y(k-2).^3
reg82=(reg6.^2).*reg7; % y(k-2).^2*y(k-3)
reg83=reg6.*reg7.^2; % y(k-2)*y(k-3).^2
reg84=reg7.^3; % y(k-3).^3

%Monta a matriz psi com cada coluna sendo um regressor em particular

psi=[reg1 reg2 reg3 reg4 reg5 reg6 reg7 reg8 reg9 reg10 reg11 reg12 reg13
reg14 reg15 reg16 reg17 reg18 reg19 reg20 reg21 reg22 reg23 reg24 reg25
reg26 reg27 reg28 reg29 reg30 reg31 reg32 reg33 reg34 reg35 reg36 reg37
reg38 reg39 reg40 reg41 reg42 reg43 reg44 reg45 reg46 reg47 reg48 reg49
reg50 reg51 reg52 reg53 reg54 reg55 reg56 reg57 reg58 reg59 reg60 reg61
reg62 reg63 reg64 reg65 reg66 reg67 reg68 reg69 reg70 reg71 reg72 reg73
reg74 reg75 reg76 reg77 reg78 reg79 reg80 reg81 reg82 reg83 reg84];
vec=y_tx2(4:m); %define vec como as amostras que se deseja identificar do
vetor de medidas
Psi=[psi vec]; %De acordo com o algoritmo de Golub-Householder (Aguirre
página 430), monta-se a matriz PSI como sendo a matriz de regressores e
a ultima coluna com y

np = Nr_Termos; % numero de parametros no modelo final
Psit=Psi(:,piv(1:np)); %define a matriz Psit com base na matriz Psi, com
todas as linhas e somente as colunas que foram selecionadas pelo vetor piv

teta=(Psit'*Psit)\Psit'*vec; %Determina os parâmetros teta utilizando mínimos
quadrados.
teta_string=cellstr(num2str(teta(:)));

%Monta a equação no formato de string
for i=1:np
    equacao=strcat(equacao,'+',teta_string(i,:),'*',Regressores(piv(i),:));
end
equacao_string=char(equacao);
equacao_string=strrep(equacao_string, ' ','');

%dados para validação
load validacao4.dat
dados2=validacao4;
u_n_decimado=dados2(:,2); %coluna um é os dados de entrada
u=decimate(u_n_decimado,100); %tx_decimacao;
y_n_decimado=dados2(:,1); %coluna dois é os dados de saída
y=decimate(y_n_decimado,100); %tx_decimacao;
yest0=y; %Inicializar as condições iniciais.

```



```

%Cálculo do erro quadrático e do BIC

for k=1:size(y)
    erro_quadratico=erro_quadratico+((y(k)-yest0(k))^2);
end
[tamanho tamanho2]=size(y);
fitness=tamanho*log(erro_quadratico/tamanho)+Nr_Termos*log(tamanho);

erro=erro_quadratico/tamanho; %(erro_quadratico/tamanho);
equation=cellstr(equacao);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Função responsável que calcula a taxa de redução do erro. %
%Fonte: Introdução a identificação de sistemas - Luis A. Aguirre - 2007. %
% %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [A,err,Piv]=myhouse(Psi,np);
% [A,err,Piv]=myhouse(Psi,np);
% Do livro Matrix Computations 2a Ed. pg 212
% Dada a matriz Psi (m,n), esta rotina acha Q de forma
% que Q'*Psi=V é triangular superior. A parte triangular
% superior de A é substituída pela parte triangular
% superior de V.
%
% Assume-se que a última coluna de Psi é o vetor de
% observacoes a ser explicado, y(k).
% np é o número de regressores escolhidos para compor o modelo
% err é um vetor de np valores que contem as taxas de redução de erro
% de cada um dos regressores escolhidos
% Piv é um vetor que contem os indices dos regressores escolhidos, ou
% seja são os indices das colunas usadas para pivotar a matriz
% Psi np vezes.

[m,n]=size(Psi);
A=Psi;
yy=Psi(:,n)'+Psi(:,n);
piv=1:n-1;

for j=1:np % Opera por colunas, ate o numero de termos final

    % Determina err para demais regressores e volta a escolher
    % o de maior valor

    for k=j:n-1 % ate completar o numero de termos candidatos
        c(k)=((A(j:m,k)'+A(j:m,n))^2)/((A(j:m,k)'+A(j:m,k))*yy); % err do re-
    gressor k
    end;

    [ans aux]=max(c(j:n-1));
    jm=j+aux-1;
    err(j)=ans;
    aux=A(:,jm); % coluna do regressor com o maior ERR
    A(:,jm)=A(:,j);
    A(:,j)=aux;

```

```

    aux=piv(jm); % indice do regressor com maior err
    piv(jm)=piv(j);
    piv(j)=aux;

    x=A(j:m,j);
    % v=house(x)
    % Do livro Matrix Computations 2a Ed. pg 196
    % Dado um vetor x, volta-se um vetor v de tal forma
    % que  $(I-2vv'/v'v)x$  é zero à excecao do primeiro elemento

    nx=length(x);
    u=norm(x,2);
    v=x;
    if u ~= 0
        b=x(1) + sign(x(1))*u;
        v(2:nx) = v(2:nx)/b;
    end;
    v(1)=1;
    % fim house(x)

    a=A(j:m,j:n);

    % a=rowhouse(a,v)
    % Do livro Matrix Computations 2a Ed. pg 197
    % Dada uma matriz A (m,n), e um vetor de comprimento m, v,
    % cujo primeiro elemento é 1, este algoritmo substitui
    % A por P*A onde  $P=I-2vv'/v'v$ 

    b=-2/(v'*v);
    w=b*a'*v;
    a=a+v*w';
    % fim rowhouse(a,v)

    A(j:m,j:n)=a;

end;
% fim myhouse(A)
Piv=piv(1:np);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                               Função que gera a população inicial                               %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function gera_populacao_pushbutton_Callback(hObject, eventdata, handles)
% hObject      handle to gera_populacao_pushbutton (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)
%Define as variáveis globais do sistema
global populacao rg_inf rg_sup pais nr_max_termos probabilidade_acumulada
probabilidade_individual;
global fitness_pais melhor melhor_fitness equation equation_string filhos
global geracao erro melhor_erro
cla(handles.axes2);
cla(handles.axes1);

geracao=0;

```

```

% Faz a leitura das variáveis definidas nos campos do software.
populacao =
str2double(get(findobj(gcf,'Tag','populacao_textbox'),'String'));
rg_inf = str2double(get(findobj(gcf,'Tag','rg1_textbox'),'String'));
rg_sup = str2double(get(findobj(gcf,'Tag','rg2_textbox'),'String'));
nr_max_termos =
str2double(get(findobj(gcf,'Tag','nr_term_textbox'),'String'));

% Gerar População e Fitness %
pais=zeros(populacao,3);
filhos=zeros(populacao,3);
fitness_pais=zeros(populacao,1);
erro=zeros(populacao,1);
probabilidade_individual=zeros(populacao,1);
probabilidade_acumulada=zeros(populacao,1);
for k=1:populacao
    pais(k,1)=randi([rg_inf rg_sup]);
    pais(k,2)=randi([rg_inf rg_sup]);
    pais(k,3)=randi([4 nr_max_termos]);
    probabilidade_acumulada(k)=0;
    probabilidade_individual(k)=0;
    [fit-
ness_pais(k),erro(k)]=fitness_function(pais(k,1),pais(k,2),pais(k,3));
end
[melhor_fitness melhor]=min(fitness_pais);
[me-
lhor_fitness,melhor_erro,equation]=fitness_function(pais(melhor,1),pais(mel-
hor,2),pais(melhor,3));
equation_string=char(equation);
equation_string=strrep(equation_string,' ','');
set(findobj(gcf,'Tag','Equation_Edit_text_box'),'String',equation_string);
tx_decimacao=max([pais(melhor,1) pais(melhor,2)]);

% Carrega dados de validação
load validacao4.dat
dados2=validacao4;
u_n_decimado=dados2(:,2); %coluna um é os dados de entrada
u=decimate(u_n_decimado,100);%tx_decimacao);
y_n_decimado=dados2(:,1); %coluna dois é os dados de saída
y=decimate(y_n_decimado,100);%tx_decimacao);
yest0=y; %Inicializar as 22 primeiras condições iniciais.
m=size(y);

% Valida a equação
for k=5:size(y)
yest0(k)=eval(equation_string);
end
axes(handles.axes3);
cla();

% Plota os gráficos
plot(yest0(1:m,1),'r','DisplayName','yest(1:250,1)','YDataSource','yest(1:2
50,1)','LineWidth',2);
hold on
plot(y(1:m,1),'black','DisplayName','y(1:250,1)','YDataSource','yest(1:250,
1)','LineWidth',2);
axis([0 inf min(y) (max(y)*1.2)])

% Define os campos do software com os valores da população gerada
set(findobj(gcf,'Tag','melhor_tx1_textbox'),'String',pais(melhor,1));
set(findobj(gcf,'Tag','melhor_tx2_textbox'),'String',pais(melhor,2));

```

```
set(findobj(gcf,'Tag','melhor_ntr_textbox'),'String',pais(melhor,3));
set(findobj(gcf,'Tag','melhor_erro_textbox'),'String',melhor_erro);
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                               Função que gera a próxima geração                               %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
function proxima_geracao_pushbutton_Callback(hObject, eventdata, handles)
```

```
%Define as variáveis globais do software
```

```
global populacao rg_inf rg_sup pais nr_max_termos probabilidade_acumulada
probabilidade_individual;
global fitness_pais melhor melhor_fitness taxa_cruzamento taxa_mutacao soma
global modo_roleta modo_elitismo tx1melhor tx2melhor nrTerm_melhor
global segundo_melhor_fitness segundo_melhor tx1_segundo_melhor
tx2_segundo_melhor
global nrTerm_segundo_melhor filhos novapopulacao sorteio_roleta indi-
ce_sorteado
global pai1_Tx1_sorteado pai1_Tx2_sorteado pai1_NR_sorteado
tx_cruzamento_sorteada
global pai2_Tx1_sorteado pai2_Tx2_sorteado pai2_NR_sorteado pon-
to_de_cruzamento
global filho1_Tx1_binario filho1_Tx2_binario filho1_NR_binario ta-
xa_mutacao_sorteada
global filho2_Tx1_binario filho2_Tx2_binario filho2_NR_binario
global filho1_Tx1_filho1_Tx2 filho1_NR filho2_Tx1 filho2_Tx2 filho2_NR
global media_aptidao geracao erro melhor_erro
```

```
for l=1:1000
```

```
ta-
```

```
xa_cruzamento=str2double(get(findobj(gcf,'Tag','tx_cruzamento_textbox'),'St-
ring'));
```

```
ta-
```

```
xa_mutacao=str2double(get(findobj(gcf,'Tag','tx_mutacao_textbox'),'String')
);
```

```
geracao=geracao+1;
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                               Modo Roleta                               %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
if (modo_roleta ==1)
```

```
    soma= sum(fitness_pais(~isnan(fitness_pais))); %calcula a soma do fit-
ness da população
```

```
    %calcula da probabilidade individual
```

```
    for i=1:populacao
```

```
        probabilidade_individual(i)=(1/fitness_pais(i))/soma;
```

```
        if(isnan(probabilidade_individual(i)))
```

```
            probabilidade_individual(i)=0;
```

```
        end
```

```
    end
```

```
    soma =
```

```
(sum(probabilidade_individual(~isnan(probabilidade_individual))));
```

```
    for i=1:populacao
```

```
        probabilidade_individual(i)=probabilidade_individual(i)/soma;
```

```
    end
```

```

%calculo da probabilidade acumulada
probabilidade_acumulada(1)=probabilidade_individual(1);
for i=2:população
    probabilidade-
de_acumulada(i)=probabilidade_individual(i)+probabilidade_acumulada(i-1);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                               Elitismo                               %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if(modulo_elitismo==1)

    %Encontrar os dois melhores indivíduos e seus valores de x e y

    [melhor_fitness melhor]=min(fitness_pais);
    tx1melhor=pais(melhor,1);
    tx2melhor=pais(melhor,2);
    nrTerm_melhor=pais(melhor,3);

    segundo_melhor_fitness=
min(fitness_pais(fitness_pais>min(fitness_pais)));

    for i=1:populacao
        if (fitness_pais(i)==segundo_melhor_fitness)
            segundo_melhor=i;
        end
    end
    tx1_segundo_melhor=pais(segundo_melhor,1);
    tx2_segundo_melhor=pais(segundo_melhor,2);
    nrTerm_segundo_melhor=pais(segundo_melhor,3);

    %Os dois melhores pais passam para a próxima geração

    filhos(1,1)=tx1melhor;
    filhos(1,2)=tx2melhor;
    filhos(1,3)=nrTerm_melhor;
    filhos(2,1)=tx1_segundo_melhor;
    filhos(2,2)=tx2_segundo_melhor;
    filhos(2,3)=nrTerm_segundo_melhor;

    %sorteio dos pais e cruzamento

    novapopulacao=populacao/2;

    for i=3:novapopulacao+1

        %Sorteio pail dos TX1
        sorteio_roleta=rand();
        for j=1:populacao
            if(probabilidade_acumulada(j)>sorteio_roleta)
                indice_sorteado=(j);
                break;
            end
        end
    end

```

```

    pai1_Tx1_sorteado=pais(indice_sorteado,1);
    pai1_Tx2_sorteado=pais(indice_sorteado,2);
    pai1_NR_sorteado=pais(indice_sorteado,3);

    %Sorteio pai1 dos TX2
    sorteio_roleta=rand();
    for j=1:populacao
        if(probabilidade_acumulada(j)>sorteio_roleta)
            indice_sorteado=(j);
            break;
        end
    end
    pai2_Tx1_sorteado=pais(indice_sorteado,1);
    pai2_Tx2_sorteado=pais(indice_sorteado,2);
    pai2_NR_sorteado=pais(indice_sorteado,3);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                               Cruzamento em modo Elitismo                               %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    tx_cruzamento_sorteada=rand();
    if(tx_cruzamento_sorteada<taxa_cruzamento)

        pai1_Tx1_binario=de2bi(pai1_Tx1_sorteado,8,'left-msb');
        pai1_Tx2_binario=de2bi(pai1_Tx2_sorteado,8,'left-msb');
        pai1_NR_binario=de2bi(pai1_NR_sorteado,4,'left-msb');
        pai2_Tx1_binario=de2bi(pai2_Tx1_sorteado,8,'left-msb');
        pai2_Tx2_binario=de2bi(pai2_Tx2_sorteado,8,'left-msb');
        pai2_NR_binario=de2bi(pai2_NR_sorteado,4,'left-msb');

        %cruzamento
        ponto_de_cruzamento=randi(8,1);
        filho1_Tx1_binario=[pai1_Tx1_binario(1:ponto_de_cruzamento)
        pai2_Tx1_binario(ponto_de_cruzamento+1:end)];
        filho1_Tx2_binario=[pai1_Tx2_binario(1:ponto_de_cruzamento)
        pai2_Tx2_binario(ponto_de_cruzamento+1:end)];
        ponto_de_cruzamento=randi(4,1);
        filho1_NR_binario=[pai1_NR_binario(1:ponto_de_cruzamento)
        pai2_NR_binario(ponto_de_cruzamento+1:end)];

        filho2_Tx1_binario=[pai2_Tx1_binario(1:ponto_de_cruzamento)
        pai1_Tx1_binario(ponto_de_cruzamento+1:end)];
        filho2_Tx2_binario=[pai2_Tx2_binario(1:ponto_de_cruzamento)
        pai1_Tx2_binario(ponto_de_cruzamento+1:end)];
        ponto_de_cruzamento=randi(4,1);
        filho2_NR_binario=[pai2_NR_binario(1:ponto_de_cruzamento)
        pai1_NR_binario(ponto_de_cruzamento+1:end)];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                               Mutação                               %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    %Mutação do primeiro filho de x
    for j=1:8
        taxa_mutacao_sorteada=rand();
        if(taxa_mutacao_sorteada<taxa_mutacao)
            if(filho1_Tx1_binario(j)==1)

```

```

        filho1_Tx1_binario(j)=0;
    else
        filho1_Tx1_binario(j)=1;
    end
end
end
for j=1:8
    taxa_mutacao_sorteada=rand();
    if(taxa_mutacao_sorteada<taxa_mutacao)
        if(filho1_Tx2_binario(j)==1)
            filho1_Tx2_binario(j)=0;
        else
            filho1_Tx2_binario(j)=1;
        end
    end
end
for j=1:4
    taxa_mutacao_sorteada=rand();
    if(taxa_mutacao_sorteada<taxa_mutacao)
        if(filho1_NR_binario(j)==1)
            filho1_NR_binario(j)=0;
        else
            filho1_NR_binario(j)=1;
        end
    end
end
end
%Mutação do segundo filho de x
for j=1:8
    taxa_mutacao_sorteada=rand();
    if(taxa_mutacao_sorteada<taxa_mutacao)
        if(filho2_Tx1_binario(j)==1)
            filho2_Tx1_binario(j)=0;
        else
            filho2_Tx1_binario(j)=1;
        end
    end
end
for j=1:8
    taxa_mutacao_sorteada=rand();
    if(taxa_mutacao_sorteada<taxa_mutacao)
        if(filho2_Tx2_binario(j)==1)
            filho2_Tx2_binario(j)=0;
        else
            filho2_Tx2_binario(j)=1;
        end
    end
end
for j=1:4
    taxa_mutacao_sorteada=rand();
    if(taxa_mutacao_sorteada<taxa_mutacao)
        if(filho2_NR_binario(j)==1)
            filho2_NR_binario(j)=0;
        else
            filho2_NR_binario(j)=1;
        end
    end
end
end

%Conversão dos filhos em decimal

filho1_Tx1=bi2de(filho1_Tx1_binario,'left-msb');

```

```

        filho1_Tx2=bi2de(filho1_Tx2_binario,'left-msb');
        filho1_NR=bi2de(filho1_NR_binario,'left-msb');
        if(filho1_NR==0)
            filho1_NR=1;
        end

        filho2_Tx1=bi2de(filho2_Tx1_binario,'left-msb');
        filho2_Tx2=bi2de(filho2_Tx2_binario,'left-msb');
        filho2_NR=bi2de(filho2_NR_binario,'left-msb');

        if(filho2_NR==0)
            filho2_NR=1;
        end

        %Atribuição dos novos filhos
        filhos(i,1)=filho1_Tx1;
        filhos(i,2)=filho1_Tx2;
        filhos(i,3)=filho1_NR;
        filhos((populacao+3)-i,1)=filho2_Tx1;
        filhos((populacao+3)-i,2)=filho2_Tx2;
        filhos((populacao+3)-i,3)=filho2_NR;

    else
        filhos(i,1)=pai1_Tx1_sorteado;
        filhos(i,2)=pai1_Tx2_sorteado;
        filhos(i,3)=pai1_NR_sorteado;
        filhos((populacao+3)-i,1)=pai2_Tx1_sorteado;
        filhos((populacao+3)-i,2)=pai2_Tx2_sorteado;
        filhos((populacao+3)-i,3)=pai2_NR_sorteado;
    end
end
pais=filhos;
for k=1:populacao
    for j=1:3
        if(pais(k,j)==0)
            pais(k,j)=1;
        end
    end
end
for k=1:populacao
    [fitness_pais(k)]=fitness_function(pais(k,1),pais(k,2),pais(k,3));
end
[melhor_fitness melhor]=min(fitness_pais);
[melhor_fitness melhor_erro equation]=fitness_function(pais(melhor,1),pais(melhor,2),pais(melhor,3));
equation_string=char(equation);
equation_string=strrep(equation_string,' ','');
set(findobj(gcf,'Tag','Equation_Edit_text_box'),'String',equation_string);
tx_decimacao=max([pais(melhor,1) pais(melhor,2)]);

%Carregar dados de validação
load validacao4.dat
dados2=validacao4;
u_n_decimado=dados2(:,2); %coluna um é os dados de entrada
u=decimate(u_n_decimado,100);%tx_decimacao);
y_n_decimado=dados2(:,1); %coluna dois é os dados de saída
y=decimate(y_n_decimado,100);%tx_decimacao);
yest0=y; %Inicializar as 22 primeiras condições iniciais.
m=size(y);

```



```

%Simulação Infinitos passos a frente usando dados de validação

for k=5:size(y)
yest0(k)=eval(equation_string);
end
axes(handles.axes3);
cla();

plot(yest0(1:m,1),'r','DisplayName','yest(1:250,1)','YDataSource','yest(1:250,1)','LineWidth',2);
hold on
plot(y(1:m,1),'black','DisplayName','y(1:250,1)','YDataSource','yest(1:250,1)','LineWidth',2);
axis([0 inf min(y) (max(y)*1.2)])

set(findobj(gcf,'Tag','melhor_tx1_textbox'),'String',pais(melhor,1));
set(findobj(gcf,'Tag','melhor_tx2_textbox'),'String',pais(melhor,2));
set(findobj(gcf,'Tag','melhor_ntr_textbox'),'String',pais(melhor,3));
set(findobj(gcf,'Tag','melhor_erro_textbox'),'String',melhor_erro);

%%%%%%%%%%Curva de Aptidão do melhor individuo%%%%%%%%%%
axes(handles.axes2);
hold on;
plot(geracao,melhor_fitness,'r*');

%%%%%%%%%%Curva de Aptidão da média dos individuos %%%%%%%%%%
axes(handles.axes1);
media_aptidao=(sum(fitness_pais(~isnan(fitness_pais)))/populacao);
hold on;
plot(geracao,media_aptidao,'b*');

end
end
end

```