UM PEQUENO "TOOLBOX" DE ALGORITMO GENÉTICO PARA CONTROLADORES PID NO "SOFTWARE" SCILAB

Caio Cesar da Cunha Feitoza – caioc.feitoza@hotmail.com
Instituto Federal de Educação, Ciência e Tecnologia do Espírito Santo – São Mateus – Bolsista PIBIC – Júnior – CNPq
Rodovia BR 101 Norte – Km 58 – Bairro Litorâneo
CEP 29932-540 – São Mateus – ES – Brasil
Gledson Melotti – gledsonmelotti@yahoo.com.br
Instituto Federal de Educação, Ciência e Tecnologia do Espírito Santo – São Mateus
Rodovia BR 101 Norte – Km 58 – Bairro Litorâneo
CEP 29932-540 – São Mateus – ES – Brasil
Humberto Mendes Mazzini – hmazzini@ufsj.edu.br
Universidade Federal de São João del-Rei, Departamento de Engenharia Elétrica
Campus Santo Antônio, Praça Frei Orlando, Centro,
CEP 36.307-352 – São João del-Rei – MG – Brasil

Resumo: *Inúmeros problemas da engenharia são solucionados por meio de técnicas que pertencem à área de inteligência computacional. Tais técnicas englobam os algoritmos genéticos (AGs). Dentre os vários problemas solucionados por AGs pode-se destacar a determinação dos parâmetros de um controlador proporcional-integral-derivativo (PID), que são amplamente utilizados em ambiente industrial, a despeito do crescente desenvolvimento da tecnologia e da teoria de sistemas de controle. Nesse sentido, faz-se necessária a introdução de técnicas de algoritmos genéticos no ensino de cursos técnicos, pois muitos alunos desses cursos estão em contato direto com a teoria e prática de processos industriais, bem como com o estudo da sintonia dos parâmetros de um controlador PID. De maneira a suprir a falta de conteúdos relacionados com inteligência computacional no ensino técnico, o presente trabalho propõe o desenvolvimento de pequeno "toolbox", que utiliza um algoritmo genético (AG) na obtenção dos parâmetros de um controlador PID. O "tolboox" em questão foi desenvolvido no ambiente SCILAB, um "software" FOSS ("Free Open Source Software").*

Palavras-chave: *Controle PID, Algoritmo genético, SCILAB, "Software" FOSS, "Toolbox".*

1. INTRODUÇÃO

A teoria de controle automático está cada vez mais integrada à área da computação. Novos algoritmos e recursos computacionais têm sido utilizados a fim de realizar os mais diversos tipos de controle (MELOTTI *et al.*, 2008a; MELOTTI *et al.*, 2008b).

Os sistemas de controle automático são encontrados em abundância em setores da

indústria, tais como: controle de qualidade e fabricação de produtos, linha de montagem automática, controle de ferramentas, tecnologia espacial e de armamento, sistemas de transporte, sistemas de potência, controle de nível, controle de temperatura, robôs e muitos outros (GOMES *et al.*, 2011; DORF & BISHOP, 2009).

Dentre os vários tipos de controladores, destaca-se o controlador PID, que é indubitavelmente o mais utilizado em ambiente industrial, principalmente porque, apesar de sua simplicidade, podem assegurar resultados satisfatórios para uma grande variedade de processos (DORF & BISHOP, 2009; MELOTTI *et al.*, 2008b). A habilidade do PID em controlar grande parte dos processos do tipo industrial resulta em sua grande aceitação nesse meio. Fornecem, também, um desempenho robusto para inúmeras condições de operação. Além disso, são fáceis de programar e de simples entendimento (DORF & BISHOP, 2009; MELOTTI *et al.*, 2008a; MELOTTI *et al.*, 2008b).

O maior desafio relacionado aos controladores PID na indústria é a sintonia adequada de seus parâmetros, necessária para se produzir uma resposta adequada do sistema. Comumente, esta sintonia é realizada de forma manual através de gráficos de reposta do sistema, o que pode ocasionar lentidão no procedimento. Nesse sentido, programas computacionais simplificam e agilizam essa tarefa, como por exemplo, o MATLAB. Entretanto, tais programas às vezes possuem um alto custo de aquisição, o que inviabiliza sua utilização em grande escala. Porém, podem ser utilizados "softwares" computacionais FOSS ("Free Open Source Software") (GOMES *et al.*, 2011; NETO & GOMES, 2010), que estão em franca expansão, como o SCILAB, de boa eficiência e de fácil manipulação (MOROMISATO *et al.*, 2007) e que será usado neste trabalho.

Nesse sentido, tornam-se importantes a implementação e a utilização de "toolboxes" no ensino de engenharia de maneira a auxiliar no aprendizado dos alunos. A exemplificação em sala de aula de métodos de controle, bem como de simulação de algoritmos genéticos, por vezes torna-se difícil, uma vez que exigem um número elevado de dados e originam matrizes de ordem muito elevadas, que são de difícil manipulação sem a colaboração de programas computacionais.

O projeto de controle utilizando métodos clássicos é dependente do conhecimento do processo para a sua análise e posterior sintonia. Tal sintonia pode ser feita de várias maneiras, dentre elas pode-se destacar a otimização por meio de algoritmo genético (AG) (SANTOS *et al.*, 2011), que será abordada neste trabalho e que consiste em determinar os três parâmetros do controlador PID: K_P (ganho proporcional), K_I (ganho integrador) e K_D (ganho derivativo). A importância de se obter parâmetros adequados para o controle PID é fazer com que a saída de um processo industrial tenha o menor erro possível, comparado com o que se deseja obter na saída do sistema (SANTOS *et al.*, 2011; DORF & BISHOP, 2009; MELOTTI *et al.*, 2008a; MELOTTI *et al.*, 2008b; MOROMISATO *et al.*, 2007).

2. MATERIAL E MÉTODOS

2.1. Programa SCILAB

Desenvolvido desde 1990 por pesquisadores da INRIA (Institut National de Recherche en Informatique et em Automatique) e ENPC (École Nationale des Ponts et Chaussées), é agora mantido e desenvolvido pelo Consórcio SCILAB desde sua criação em maio de 2003. O programa SCILAB foi desenvolvido para aplicações em controle de sistemas e processamento de sinais. É distribuído gratuitamente em formato de código fonte. O SCILAB é composto de

três partes distintas: um interpretador, uma biblioteca de funções (rotinas do SCILAB) e uma biblioteca de rotinas de Fortran e C. Um aspecto interessante do SCILAB é a possibilidade do desenvolvimento de cálculo matricial: manipulações básicas de matrizes como concatenação, extração ou transposição são imediatamente processadas, bem como operações básicas como adição ou multiplicação. O SCILAB também manipula objetos mais complexos que matrizes numéricas, como matrizes rotacionais ou polinomiais, além de funções de transferência, comandos matemáticos para análise de sistemas dinâmicos e gráficos (MOROMISATO *et a*l., 2007).

O SCILAB possui uma programação aberta na qual a criação das funções e bibliotecas de funções está completamente nas mãos dos usuários. Funções são reconhecidas como objetos de dados no SCILAB e, então podem ser manipuladas ou criadas como outros objetos de dados. Por exemplo, funções podem ser definidas dentro do SCILAB e passadas como argumentos de entrada ou saída para outras funções. Além disso, o SCILAB suporta dados do tipo "character string". Matrizes de "character string" são também manipuladas com a mesma sintaxe de matrizes ordinárias (MOROMISATO *et a*l., 2007).

Finalmente, o SCILAB é facilmente ligado com subprogramas de Fortran e C. Isso permite o uso de pacotes padrões e bibliotecas no desenvolvimento do SCILAB.

A filosofia geral do SCILAB é fornecer a seguinte variedade de ambiente computacional:

- Ter tipos de dados que são variados e flexíveis com uma sintaxe que seja natural e fácil de usar.
- Fornecer uma variedade razoável de rotinas que servem como uma base para uma variedade ampla de cálculos.
- Ter um ambiente de programação aberto em que as novas rotinas sejam adicionadas facilmente.
- Suportar o desenvolvimento de biblioteca através dos "toolboxes" das funções para aplicações específicas (processamento de sinais, controle não linear, etc.).

O SCILAB fornece também um "desktop" fácil para desenvolver cálculos breves ou para um primeiro teste das funções, Figura 1a), e também possui um editor de texto integrado conhecido como SciNotes, conforme a Figura 1b).

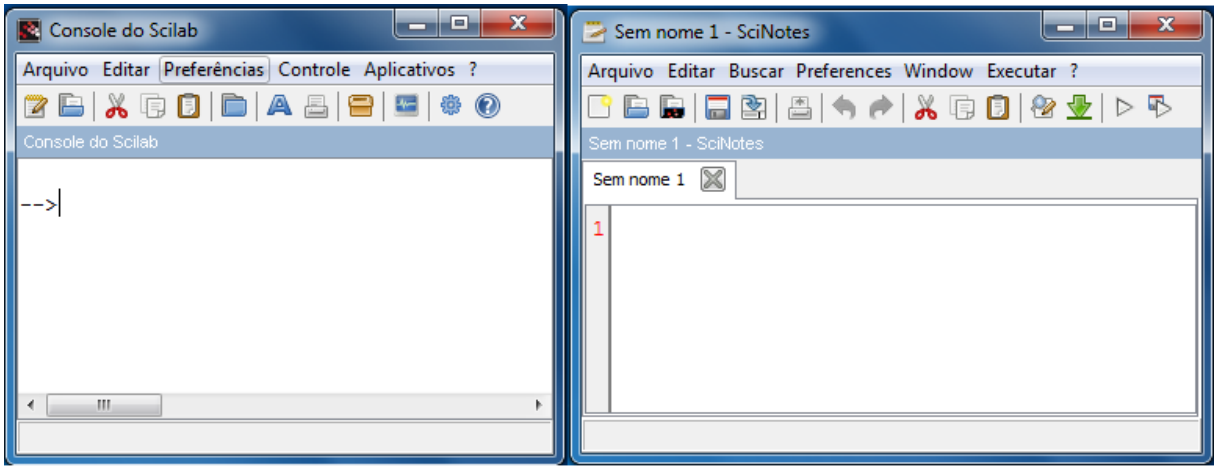

Figura 1: a) Tela principal do SCILAB; b) Tela do editor de texto.

2.2. Controlador PID

A Figura 2 representa uma estrutura genérica de uma malha de controle fechada com o algoritmo genético:

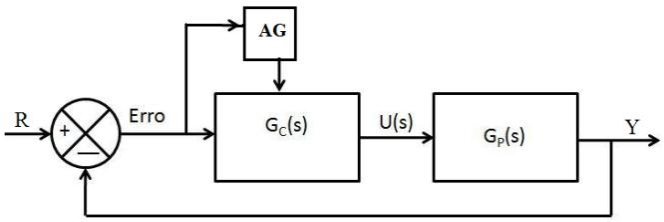

Figura 2: Diagrama de blocos de uma malha de controle realimentada com AG.

A entrada do sistema também é chamada de "set point". O controlador é representado por $G_C(s)$ e $G_P(s)$ é definida como o modelo da planta a ser controlada. A variável $U(s)$ é o sinal de saída do controlador, $R(s)$ a entrada do sistema, $Y(s)$ é a saída do sistema e AG é o algoritmo genético.

O controlador PID neste trabalho foi o de configuração ideal. Sua função de transferência é dada pela "Equação (1)" e a malha fechada do sistema pela "Equação (2)":

$$G_C(s) = \frac{K_D s^2 + K_P s + K_I}{s} \tag{1}$$

$$\frac{Y(s)}{R(s)} = \frac{G_P(s)G_C(s)}{1 + G_P(s)G_C(s)} \tag{2}$$

Em que K_P é o ganho proporcional, K_I é o ganho integral e K_D é o ganho derivativo. Tais parâmetros serão determinados pelo AG, a otimizar o desempenho do sistema. O erro que se origina é eliminado obtendo-se os valores adequados de K_P, K_I e K_D que minimize um índice de desempenho conhecido como ITAE[1] (integral do tempo multiplicado pelo valor absoluto do erro).

Nesse sentido, o projeto busca minimizar determinados índices de desempenho da resposta do sistema em malha fechada[2]. Alguns desses índices são conhecidos como tempo de subida, tempo de acomodação e o sobressinal. Ao minimizar tais índices, contribui-se para minimizar o erro do sistema (resposta da saída comparada com a entrada).

2.3. Algoritmo genético

O algoritmo genético (AG) é uma ferramenta matemática usada para realizar uma otimização, que efetua um tipo de busca global aproximada. O AG segue o procedimento

[1] Existem outros índices de desempenhos para eliminar o tempo de subida, o tempo de acomodação e o sobressinal da resposta do sistema. Porém, neste trabalho usou-se o ITAE, por nenhum motivo em especial.
[2] Sistema em malha fechada é o sistema realimentado.

descrito a seguir (LINDEN, 2008; MELOTTI *et al.*, 2008a; TAKAHASHI & MARTINS, 2004):

- depende da informação obtida pela avaliação de diversos pontos no espaço de busca. Cada "ponto atual" é chamado de indivíduo, e o conjunto de "pontos atuais" é chamado população;
- a população converge para o ótimo de um problema através de operações sequenciais em cada iteração dos seguintes operadores: seleção, "crossover" (cruzamento), mutação e elitismo. Os operadores "crossover" e mutação são conhecidos como operadores genéticos. Segundo TAKAHASHI & MARTINS (2004) as definição de tais operadores são:

    1. seleção: operação que gera uma nova população a partir da população corrente, permitindo a transmissão à nova população dos indivíduos da população atual, com maior probabilidade para os indivíduos com melhor função-objetivo;
    2. cruzamento: operação que combina a informação de dois indivíduos, gerando novos indivíduos;
    3. mutação: operação que "perturba" um indivíduo, gerando um novo indivíduo com alguma semelhança com o indivíduo que o originou;
    4. elitismo: causa a seleção determinística de parte da população corrente, usualmente os melhores indivíduos, para integrarem a nova população.

Além dos operadores sequenciais, outras duas etapas muito importantes nos AGs são o tamanho da população e a função "fitness":

- tamanho da população: uma população pequena significa que o espaço de busca pode ter uma diversidade pequena, o que significa que o AG pode convergir prematuramente. Com uma população grande o AG pode ter uma diversidade maior, evitando assim uma convergência prematura. Dessa forma o AG pode oferecer uma solução mais eficiente, pois o espaço de busca é maior;
- função "fitness": é a função que determina a qualidade de um indivíduo. Nesse sentido, é possível comparar qual indivíduo é melhor que o outro. O melhor indivíduo, isto é, o melhor resultado é escolhido a partir do valor da função "fitness". Isso significa, em AG, que o melhor indivíduo possui o maior valor.

Cada operador genético pode ser implementado de diversas maneiras. Uma combinação de realizações específicas desses operadores constitui uma forma de algoritmo genético.

O AG básico pode ser formulado da seguinte forma:

Entrada: População inicial

Avalie a população por meio da função "fitness"

Repita (Início da evolução – gerações)

Passo 1: Seleção dos indivíduos pais por meio da "fitness"

Passo 2: Aplicar os operadores genéticos

- Aplicar "crossover" (reprodução) para cada par de pais
    1. Obter os novos indivíduos (filhos)
- Aplicar mutação nos indivíduos filhos

Passo 4: Obtenha a nova população de indivíduos

Passo 3: Calcular a nova "fitness" de cada indivíduo

- Aplicar elitismo

Final: Até que o critério de parada seja atendido
Saída: Obtenha o melhor indivíduo presente na última geração.

O AG desenvolvido neste trabalho, por meio do editor de texto SciNotes do SCILAB, é o binário. Desta forma, cada indivíduo é uma sequência de números binários, nos quais se aplicam os operadores genéticos ("crossover" e mutação). Os indivíduos representados por números binários obedecem a uma faixa de limite superior e inferior de valores reais. Tais números binários são convertidos para números reais dentro do AG para obter o valor da função "fitness". A expressão matemática que converte de um número binário para um número real dentro da uma faixa de limite de valores reais é representada pela "Equação (3)" (LINDEN, 2008):

$$real = inf + \left(\frac{sup - inf}{2^k - 1}\right) \times r \tag{3}$$

em que inf é o limite inferior de um valor real, sup é o limite superior de um valor real, k é o número de bits que compõe um indivíduo e r é o número real correspondente ao número binário.

2.4. Modelo para otimização

Foi formulado um modelo matemático para realizar a otimização do controle PID, a fim de determinar os ganhos K_P, K_Ie K_D. O modelo, M_1, considera o índice de desempenho ITAE.

$$M_1 = minimize\left(ITAE = \int_0^t t \times |e(t)|dt\right) \tag{4}$$

Nota-se que o modelo M_1, dentro do AG, foi formulado em termos de maximização, pois se trata de um algoritmo evolucionário:

- Cálculo do ITAE:

$$\begin{gathered} Erro = Entrada - Saída \\ abe = absoluto(Erro) \\ I = integral(t \times abe) \end{gathered} \tag{5}$$

- Transformando para um problema de maximização:

$$M_2 = \frac{1}{I} \tag{6}$$

em que I e M_2 são os resultados de um conjunto de indivíduos, isto é, uma população obtido a partir da função "fitness". Assim, deseja-se obter os valores do ganho para que se tenha o maior valor de M_2 para que se tenha o menor erro possível do sistema.

3. RESULTADOS E DISCUSSÕES

O AG desenvolvido no "software" SCILAB foi um AG simples. Tal algoritmo usou a seguinte configuração: mutação bit a bit, seleção de pais para o "crossover" por meio do processo conhecido como roleta e um ponto de corte para o "crossover", além do elitismo. A planta de teste simulada para o controle PID foi uma planta de primeira ordem estável sem atraso, conforme a "Equação (7)". O modelo matemático ou função de transferência em malha fechada é representado pela "Equação (8)".

$$G_P(s) = \frac{3{,}8}{192{,}5s + 1} \tag{7}$$

$$\frac{Y(s)}{R(s)} = \frac{3{,}8K_D s^2 + 3{,}8K_P s + 3{,}8K_I}{s^2(192{,}5 + 3{,}8K_D) + s(1 + 3{,}8K_P) + 3{,}8K_I} \tag{8}$$

O primeiro passo do "toolbox" é a entrada do tempo de simulação, ilustrado na Figura 3.

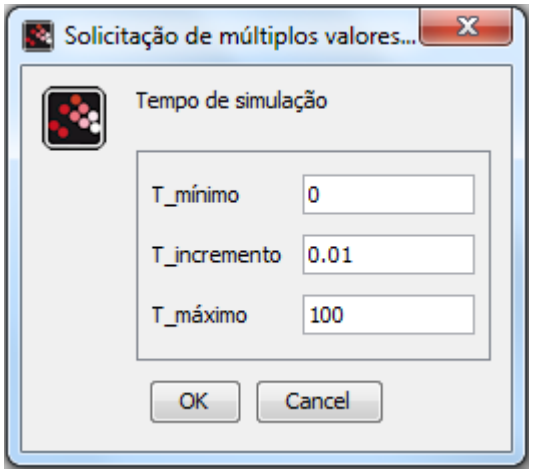

Figura 3: Primeiro passo do "toolbox". Entrada do tempo de simulação.

O numerador e denominador da função de transferência são incluídos no passo 2, conforme a Figura 4.

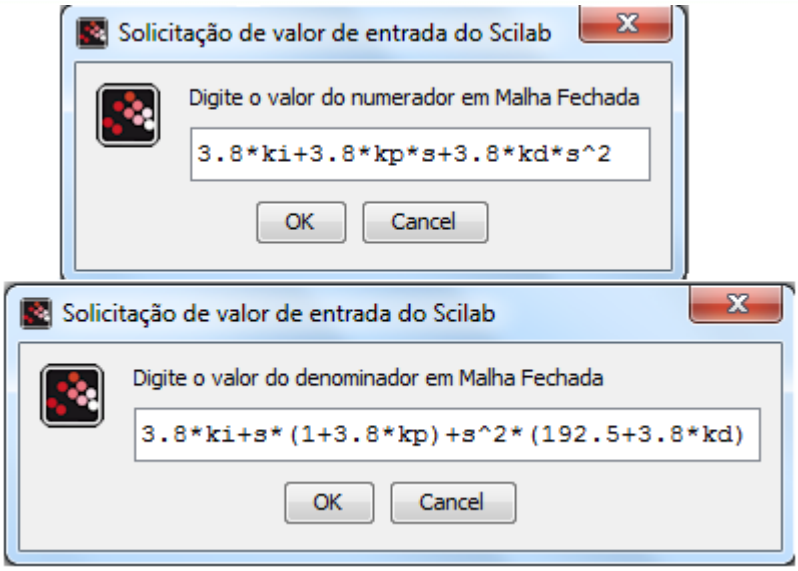

Figura 4: Segundo passo do "toolbox". Entrada da função de transferência em malha fechada.

Os parâmetros do AG podem ser vistos na Figura 5, que representa o terceiro passo do "toolbox".

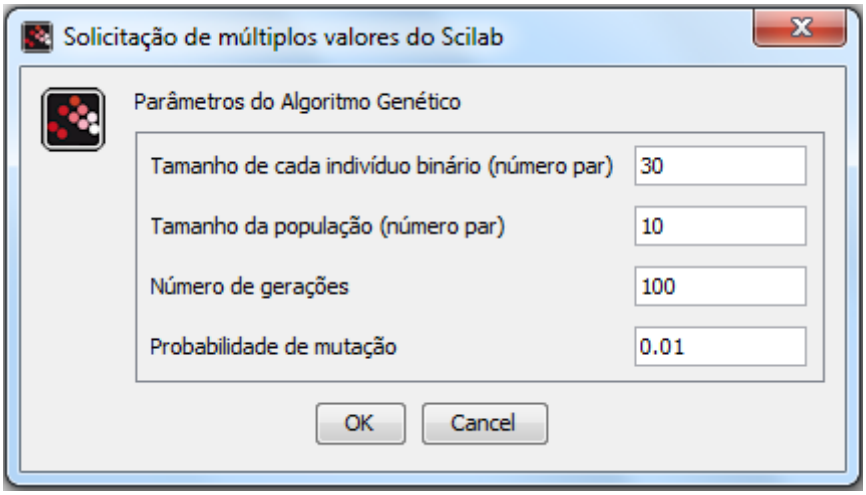

Figura 5: Terceiro passo do "toolbox". Entrada dos parâmetros do AG.

O último passo do "toolbox" são os valores de máximo e mínimo dos parâmetros do controlador PID, mostrados na Figura 6.

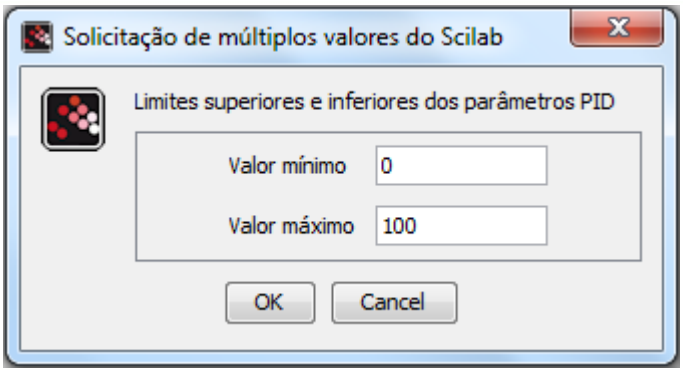

Figura 6: Quarto passo do "toolbox". Limites inferior e superior dos parâmetros do controlador PID.

Após a simulação, o AG desenvolvido no ambiente SCILAB fornece os parâmetros do controlador PID, como na Figura 7. Note que estes parâmetros fornecerem o melhor resultado para o intervalo de simulação, ou seja, o menor índice de desempenho ITAE.

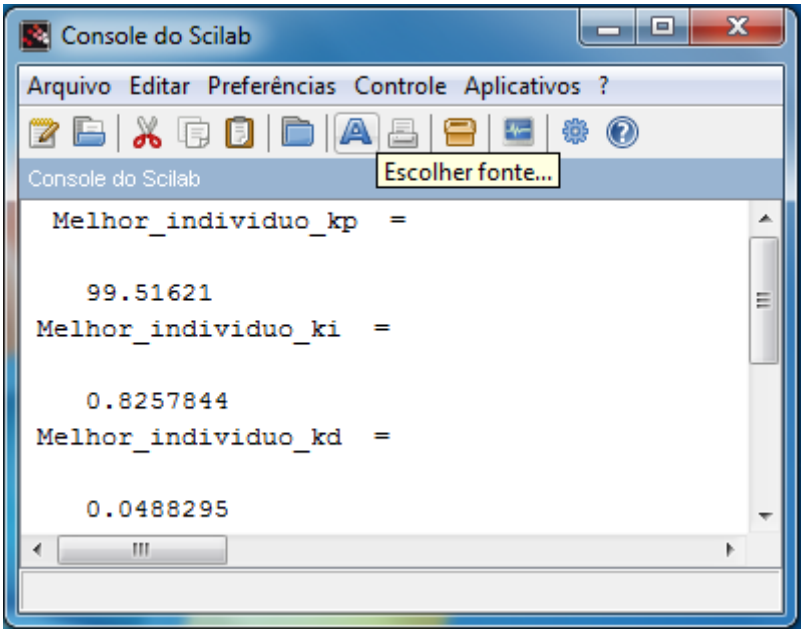

Figura 7: Resultado final dos ganhos do controlador PID.

A resposta da saída do sistema em malha fechada pode ser ilustrada pela Figura 8a), bem como para os valores do índice de desempenho que decrescem (eixo na vertical) a cada geração (eixo horizontal), conforme ilustrado na Figura 8b).

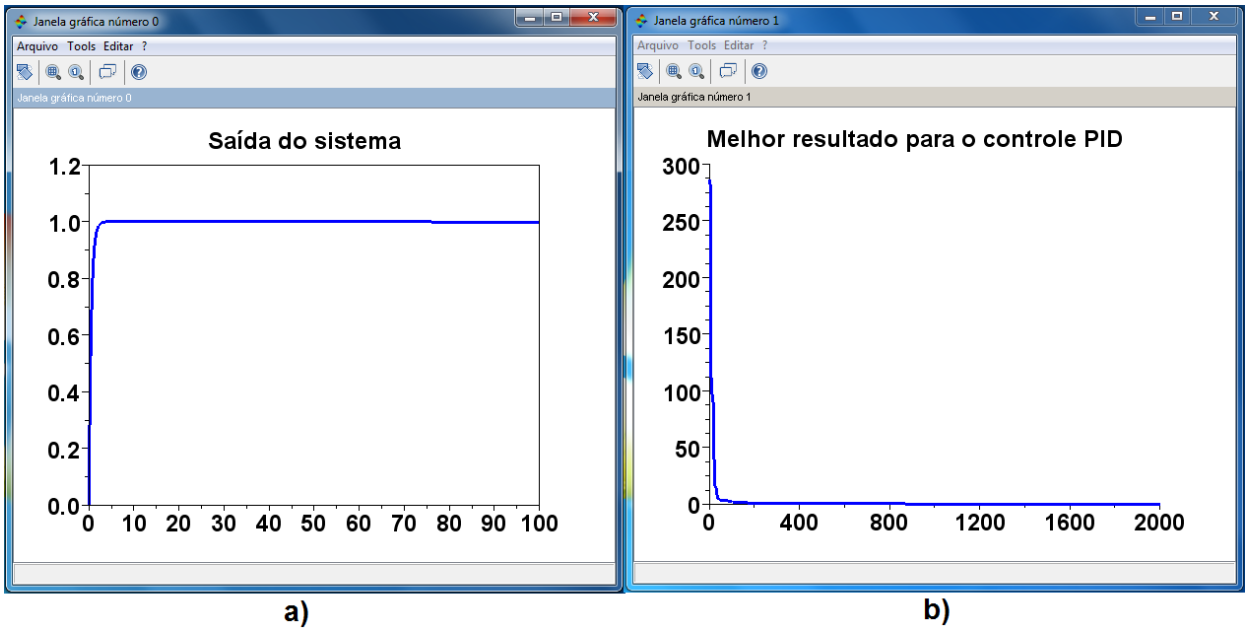

Figura 8: $a)$ Saída do sistema em malha fechada; $b)$ Índice de desempenho ITAE para cada geração.

Podem-se obter outras soluções mais viáveis modificando os valores dos parâmetros do AG, como o tamanho de cada indivíduo, o número de gerações, probabilidade de mutação e o tempo de simulação, bem como os valores máximos e mínimos dos parâmetros do controlador PID. Pode-se observar, entretanto, que pelo fato do AG possuir algumas partes aleatórias durante a simulação, alguns resultados podem ser inferiores ao resultado mostrado pela Figura 8 ao modificar os parâmetros do AG.

Agradecimentos

Os autores do projeto agradecem ao IFES – São Mateus, CNPq, CAPES, FAPEMIG pelo apoio financeiro. Os autores também agradecem a professora de português Adriana Pin por esclarecer algumas dúvidas de gramáticas.

4. CONCLUSÃO

O desenvolvimento deste trabalho teve como finalidade motivar tanto professores quanto alunos a adotarem o programa SCILAB, como ferramenta de auxílio nos estudos de simulações para os cursos técnicos e também os de engenharias.

O programa SCILAB pode ser de grande valia na simulação de computação evolucionária, como em otimização e controle PID, auxiliando professores e alunos em suas tarefas, além do fato de possuir um custo zero para as instituições de ensino, uma vez que o "software" pode ser adquirido livremente pelo site http://www.scilab.org.

Por meio da tela principal do SCILAB obteve-se o editor de texto SciNotes. Tal editor foi utilizado para desenvolver o AG simples, como dito em parágrafos anteriores. Pretende-se melhorar o AG por meio da inclusão de outras metodologias, como a ideia de "Nicho" e "torneio" para a seleção de pais para o "crossover". Além disso, pretende-se testar a rotina desenvolvida com modelos matemáticos de processos instáveis, integradores e funções de segunda ordem.

REFERÊNCIAS BIBLIOGRÁFICAS

DORF, Richard C.; BISHOP, Robert H. Sistemas de Controle Moderno. Rio de Janeiro-RJ: Editora LTC, 11ª ed. 2009.

SANTOS, F.G.; MAZZINI, H.M.; BAETA, B.S.; MELOTTI, G. Métodos de Sintonia de Controladores PI/PID Aplicados a um Sistema de Nível. **Anais:** X Simpósio Brasileiro de Automação Inteligente. São João del-Rei-MG: UFSJ, 2011.

GOMES, F.J.; QUEIROZ, F.P; GAMA, A.V.; BALDIOTI, H.R. Módulo Laboratorial de Baixo Custo, Baseado em FOSS, para Educação em Engenharia de Controle de Processos Industriais. **Anais:** XXXIX Congresso Brasileiro de Educação em Engenharia. Blumenau-SC: FURB, 2011.

LINDEN, Ricardo. Algoritmos Genéticos – Uma Importante Ferramenta da Inteligência Computacional. 2ª ed. Rio de Janeiro-RJ: Editora Brasport, 2008.

MELOTTI, G.; BAETA, B.S.; MAZZINI, H.M.; NETO, O.M. Sintonia de um Controle PID Para O Sistema Mola-Amortecedor da Suspensão de um Veículo Usando Otimização Multiobjetivo. **Anais:** XVII Congresso Brasileiro de Automática. Juiz de Fora-MG: UFJF, 2008a.

MELOTTI, G.; TEIXEIRA, D.A.; MENDES, E.M.A.M.; VASCONCELOS, J.A. Sintonia de Controlador PID Usando Rede Imunológica Artificial. **Anais:** XVII Congresso Brasileiro de Automática. Juiz de Fora: UFJF-MG, 2008b.

MOROMISATO, G.D.Y.; MELOTTI, G.; GARCIA, F.A.; MAZZINI, H.M.; NEPOMUCENO, E.G. A Utilização de Um "Software" Livre no Ensino de Sistemas de Controle. **Anais:** International Conference on Engineering and Computer Education. Monguaguá/Santos-SP: COPEC, 2007.

NETO, A.F.S.; GOMES, F.J. Controladores PID: Introduzindo Inteligência Computacional no Controle Industrial. **Anais:** XXXVIII Congresso Brasileiro de Educação em Engenharia. Fortaleza-CE: UFC/UNIFOR, 2010.

TAKAHASHI, R.H.C.; MARTINS, F.V.C. Avaliação e Interpretação do Operador mAG no Algoritmo Genético Real-Polarizado. **Anais:** XXXVI Simpósio Brasileiro de Pesquisa Operacional. São João del-Rei-MG: UFSJ, 2004.

A SMALL "TOOLBOX" FOR GENETIC ALGORITHM FOR THE PID CONTROLLERS IN THE "SOFTWARE" SCILAB

Abstract: *Many engineering problems are solved by techniques that belong to a group known as computational intelligence methodology. Such techniques include genetic algorithms (GAs). Among the many problems solved by GAs, we can highlight the determination of the*

parameters of a controller proportional-integral-derivative (PID), which are widely used in industrial environments, despite the increasing development of technology and theory of control systems. Thus, it is necessary to introduce the subject of genetic algorithm in the teaching of technical courses, as many students of technical courses are in direct contact with industrial processes, they need to determine the parameters of a PID controller. To address the lack of content related to computational intelligence in technical education, this paper proposes a project to develop a small "toolbox" that includes a genetic algorithm (GA) in order to get the PID controller parameters. The toolbox in question will be developed in SCILAB mathematical environment, a software FOSS ("Free Open Source Software").

Key-words: *PID controller, Genetic algorithms, SCILAB, Software FOSS, Toolbox.*