

DESENVOLVIMENTO DE UMA INTERFACE GRÁFICA PARA RESOLUÇÃO DE PROBLEMAS UTILIZANDO ALGORITMO DE EVOLUÇÃO DIFERENCIAL

Adjuto Vasconcelos adjuto@ifes.edu.br

Programa de Pós-Graduação em Engenharia de Controle e Automação

IFES-Campus Serra

ES-010 S/N, Km-6,5 – Manguinhos

CEP: 29173-087 – Serra – ES, Brasil

Raphael Rebello Haddad haddadrr@gmail.com

Programa de Pós-Graduação em Engenharia de Controle e Automação

IFES-Campus Serra

ES-010 S/N, Km-6,5 – Manguinhos

CEP: 29173-087 – Serra – ES, Brasil

Ben-Hur Prando bhprando@hotmail.com

Programa de Pós-Graduação em Engenharia de Controle e Automação

IFES-Campus Serra

ES-010 S/N, Km-6,5 – Manguinhos

CEP: 29173-087 – Serra – ES, Brasil

João Marcos dos Santos Souza jmarcosdss@yahoo.com.br

Programa de Pós-Graduação em Engenharia de Controle e Automação

IFES-Campus Serra

ES-010 S/N, Km-6,5 – Manguinhos

CEP: 29173-087 – Serra – ES, Brasil

Rodrigo Piol Capucho rodrigop@ifes.edu.br

Programa de Pós-Graduação em Engenharia de Controle e Automação

IFES-Campus Serra

ES-010 S/N, Km-6,5 – Manguinhos

CEP: 29173-087 – Serra – ES, Brasil

Dr. Wagner Teixeira da Costa wagnercosta@ifes.edu.br

Programa de Pós-Graduação em Engenharia de Controle e Automação

IFES-Campus Serra

ES-010 S/N, Km-6,5 – Manguinhos

CEP: 29173-087 – Serra – ES, Brasil

Resumo: Este trabalho propõe o desenvolvimento de uma interface gráfica de usuário para servir de auxílio a interação deste para auxiliar a entrada de dados e interpretação de resultados de saída na resolução de problemas de engenharia propostos utilizando algoritmo evolucionário diferencial (ED). A utilização da interface permite flexibilizar a resolução de problemas utilizando o ED, uma vez que será possível a utilização de funções de avaliação fornecida pelo usuário de forma interativa através da interface. Desta forma, será possível utilizar o programa para resolver diferentes problemas propostos com diferentes parâmetros de entrada definidos pelo usuário. O trabalho proposto também contribui para o ensino e difusão do ED uma vez que o entendimento do procedimento de execução do algoritmo é exposto de forma clara para o usuário, com fácil interpretação dos resultados.

Palavras-chave: computação evolucionária, evolução diferencial, interface gráfica.

1 INTRODUÇÃO

O projeto de sistemas de engenharia requer, ao mesmo tempo, conhecimentos relacionados com cada disciplina de engenharia, bem como técnicas capazes de tratar com eficiência o número imenso de possíveis soluções de projeto. Tais técnicas são conhecidas como métodos de otimização e, à medida em que avança a tecnologia, torna-se cada vez mais importante o uso delas, devido ao aumento da complexidade dos sistemas a serem projetados em todos os ramos da engenharia. Nos últimos anos, pesquisadores tem desenvolvido novas abordagens para problemas de otimização baseando-se em mecanismos biológicos de adaptação, tais como os observados na teoria da evolução das espécies, ou no comportamento de insetos sociais. Tais algoritmos, coletivamente conhecidos como técnicas de Computação Evolucionária, tem se mostrado altamente eficientes para resolver muitos problemas complexos de engenharia. Dentre os métodos de Computação Evolucionária abordamos o algoritmo Evolução Diferencial (ED).

A Evolução Diferencial (ED) é um algoritmo evolutivo desenvolvido por Rainer Storn e Kenneth Price em 1995, na tentativa de resolver o problema de ajuste polinomial de *Chebyshev* (STORN; PRICE, 1995). Apesar da ED apresentar conceitos simples e ser um algoritmo de fácil implementação, esta técnica é robusta e eficiente na minimização de funções não-lineares e não-diferenciáveis no espaço contínuo (COELHO; MARIANI, 2004). Em (STORN, 1996) são disponibilizadas várias bibliotecas do algoritmo ED em diversas linguagens de programação.

Na ED, as populações de indivíduos são criadas e submetidas aos operadores genéticos: seleção, mutação e recombinação (*crossover*). Estes operadores utilizam uma caracterização da qualidade de cada indivíduo como solução do problema (avaliação) e vão gerar um processo de evolução natural destes indivíduos, que eventualmente deverá gerar um indivíduo que caracterizará uma boa solução, ou a melhor possível, para o problema (LINDEN, 2006).

A criação de uma interface gráfica utilizando a modelagem e simulação para resolução de problemas usando ED é uma medida importante para auxiliar nessa problemática. As interfaces gráficas facilitam a interação entre usuário e a máquina. Este trabalho tem como objetivo o desenvolvimento de uma interface gráfica para resolução de problemas usando ED, utilizando o programa MATLAB®.

O MATLAB® é um programa computacional muito empregado em ambientes acadêmicos e que facilita a resolução de problemas propostos em engenharia. Essa plataforma permite a utilização de controles de fluxo tradicionais, encontrados em diversas linguagens de programação, como também representação e manipulação matricial de dados, o que o torna

atrativo para trabalhar com problemas possuem caráter matemático, tais como o proposto usando ED.

Aliada a possibilidade de tratamento gráfico, o MATLAB® possui **IDE**, do inglês *Integrated Development Environment* ou **Ambiente de Desenvolvimento Integrado**, sendo um programa de computador que reúne características para ambiente e ferramentas de apoio ao desenvolvimento de software com o objetivo de agilizar o processo de criação de aplicações.

Geralmente os IDEs facilitam a técnica de RAD (de *Rapid Application Development*, ou "Desenvolvimento Rápido de Aplicativos"), que visa a maior produtividade dos desenvolvedores.

Dentro do IDE do MATLAB®, um dos recursos disponíveis para o programador é o GUIDE. O GUIDE (*Graphical User Interface Development Environment*) é o ambiente de criação de interfaces gráficas do MATLAB®. Esse ambiente fornece um conjunto de ferramentas para a criação de GUI (*Graphical User Interface*). Estas ferramentas simplificam o processo de diagramação e programação de interfaces gráficas. Quando se cria uma GUI no GUIDE, este se encarrega de criar dois arquivos: o arquivo de corpo da interface gráfica, que é salvo em uma extensão .fig e o arquivo que contém as funções que controlam a GUI, este arquivo é salvo na extensão .m do MATLAB®.

Para a construção de uma **interface com o usuário (IU)**, há várias técnicas para o desenho destas interfaces, mas é importante ter sempre em mente que uma boa interface deve proporcionar ao usuário uma boa usabilidade. Em Shneiderman(2010), recomenda-se oito princípios, também chamados “regras de ouro” que são aplicados nos mais diversos sistemas. Os princípios são: clareza, concisão, familiaridade, reatividade, consistência, atratividade, eficiência e capacidade de desfazer.

Este trabalho propõe o desenvolvimento de uma interface de usuário, usando o poder de processamento matemático do MATLAB® capaz facilitar a interação do usuário na parametrização de problemas genéricos que usam algoritmo ED. O usuário será capaz de parametrizar os parâmetros do ED, tais como população, fator de escala e *crossover* (WANG,2011). Ainda a interface facilitará a leitura e interpretação dos resultados de forma gráfica e amigável.

O uso de interface gráfica que proporcione ao usuário a parametrização e a visualização dos resultados gerados por algoritmos genéticos permite uma interação satisfatória entre o homem e máquina, o que favorece o processo de aprendizagem.

2 ALGORITMO EVOLUTIVO DIFERENCIAL

Os algoritmos evolutivos constituem uma técnica de busca e otimização, altamente paralela, inspirada no princípio Darwiniano de seleção natural e reprodução genética (GOLDBERG,1999). São algoritmos probabilísticos que fornecem um mecanismo de busca paralela e adaptativa baseado no princípio de sobrevivência dos mais aptos e na reprodução (PACHECO,1999). Neste trabalho é utilizado o algoritmo evolutivo denominado Evolução Diferencial desenvolvido por Storn, que tem sido aplicado com sucesso em vários campos da ciência.

Na Evolução Diferencial as populações de indivíduos são criadas e submetidas aos operadores genéticos: seleção, diferenciação, recombinação (crossover) e mutação. Estes operadores utilizam uma caracterização da qualidade de cada indivíduo como solução do problema (avaliação) e vão gerar um processo de evolução natural destes indivíduos, que eventualmente deverá gerar um indivíduo que caracterizará uma boa solução, ou a melhor possível, para o problema (LINDEN,2006).

2.1 Parâmetros de Controle

Na Evolução Diferencial é necessário configurar cinco parâmetros de controle do algoritmo, são eles:

- D - número de dimensões (tamanho do cromossomo);
- N_P - tamanho da população;
- C_R - constante de crossover;
- F - peso aplicado ao vetor de diferenças (constante de mutação);
- gen_max - número máximo de gerações

2.2 Função de Avaliação:

A proposta do trabalho será de incluir uma função genérica de avaliação, sendo que cada função será passada pela interface através da função de custo apontada pelo arquivo em .m, disponível pela interface.

O usuário do programa poderá escolher a função de avaliação que melhor represente o problema que este deseja resolver. Assim a aplicação desenvolvida será bastante versátil, uma vez que poderá ser reutilizada para resolver vários problemas diferentes (LINDEN, R.,2006).

2.3 Algoritmo:

Um algoritmo genético pode ser descrito como um processo contínuo que repete ciclos de evolução controlados por um critério de parada, conforme apresentado em Pacheco(2009). Este processo pode ser resumido algoritmicamente da seguinte forma:

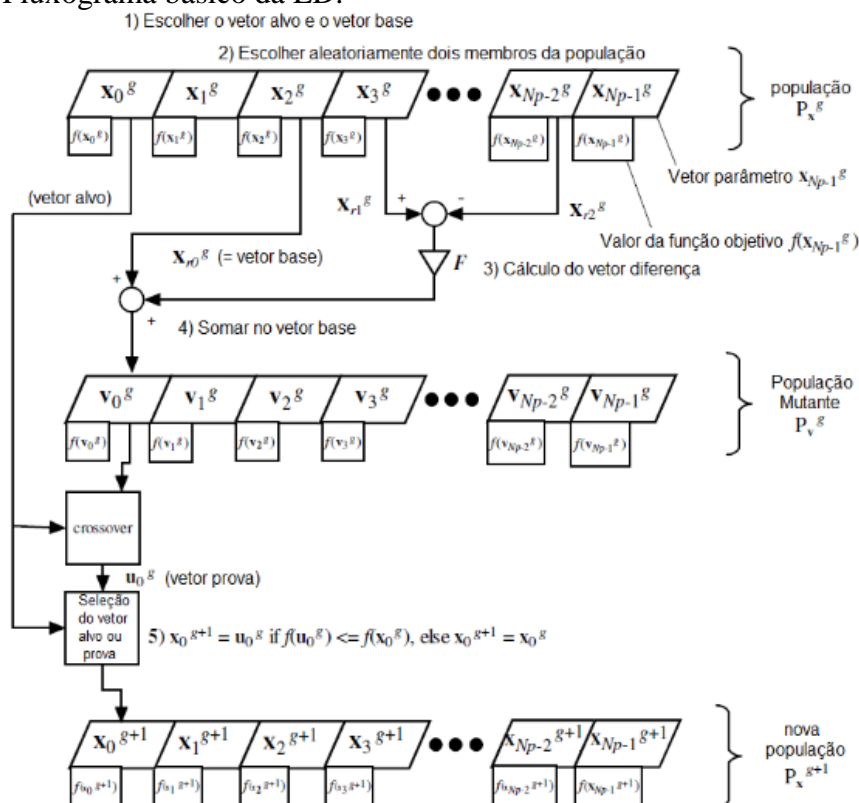
- Passo 1 - Inicializar D, NP, CR, F & gen_max;
- Passo 2 - Inicializar, randomicamente, a população inicial;
- Passo 3 - Avaliar cada indivíduo;
- Passo 4 - Repetir até que o critério de parada seja satisfeito;

I - Para cada indivíduo

- 1 - Selecionar randomicamente 3 indivíduos da população;
- 2 - Aplicar diferenciação, mutação, crossover;
- 3- Comparar indivíduo com a sua versão experimental e selecionar o de menor custo para a nova população;

- - Avaliar o custo do indivíduo selecionado;
- II - Fim Para cada indivíduo
- Passo 5 - Fim Repetir

Figura 1. Fluxograma básico da ED.



Fonte: (STORN; PRICE, 1996)

Em Chakraborty (2008) e Storn indicam algumas diretrizes gerais:

- A população inicial deve ser inicializada o mais próximo do espaço de busca (regiões promissoras ou soluções parciais);
 - Quando não houver convergência, escolher $CR \in [0,8,1]$;
 - Geralmente N_P pode assumir o valor igual a 10D;
 - Escolher $F \in [0,5,1]$;
 - Quanto maior o tamanho da população, menor o valor de F ;
 - Se o valor da função custo cair de forma inesperada, a minimização é um mínimo local;
- É fundamental a escolha adequada da função custo, pois quanto mais informações sobre o problema na função custo, maior será a possibilidade de se obter uma convergência suave e coerente.

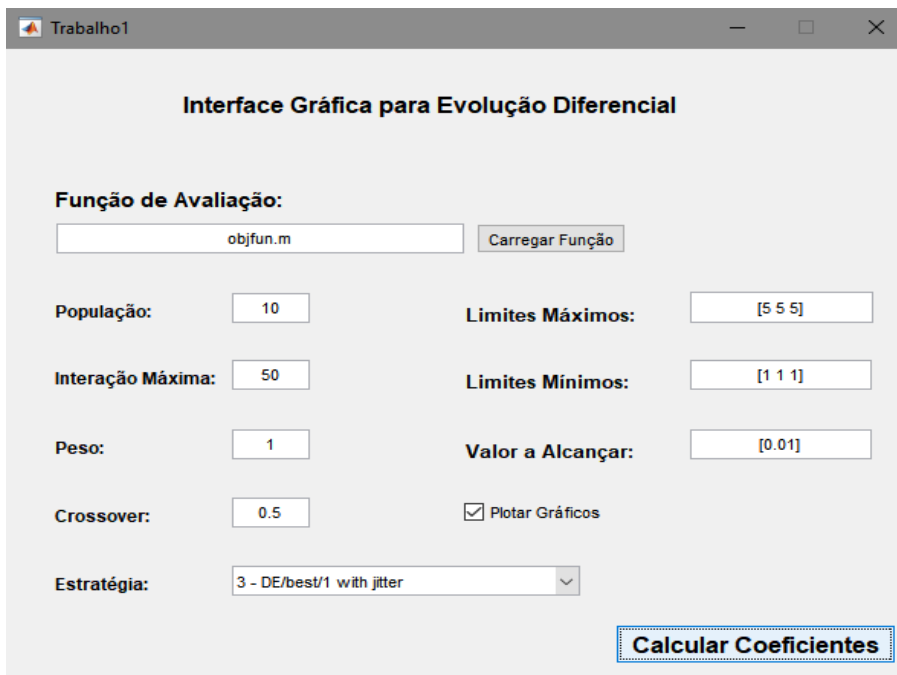
3 MATERIAS E MÉTODOS

3.1 Desenvolvendo o Layout da aplicação

Para fins didáticos, foi elaborada uma interface gráfica (conforme Figura 2) para facilitar a introdução de parâmetros do algoritmo e da função de avaliação a ser solucionada.

Dos campos da interface que foram criados para o auxílio didático pode-se citar os seguintes itens a serem preenchidos pelo usuário: **Função de Avaliação** é o campo que será carregado o arquivo em que contempla a função que será estudada; **População** é a quantidade de indivíduos que serão gerados para as interações; **Interação máxima** é o valor máximo de interações que o algoritmo executará em busca da solução (quanto mais interações, maior a chance de resolver o problema e ter um resultado ótimo); **Peso**, também chamado como **Fator de Mutação** é o valor pelo qual multiplica a diferença entre dois indivíduos escolhidos aleatoriamente numa população; **Crossover** tem por objetivo o aumento da diversidade que irá misturar o vetor mutação e o vetor alvo; **Estratégia** é a diretriz pela qual a função será avaliada, sabendo que diferentes estratégias podem ser obtidas alterando-se a forma de obtenção do vetor mutação, em estratégias mais convencionais a mutação pode variar de acordo com o vetor base e número de indivíduos, por exemplo; **Limites Máximos** são os valores máximos que são determinados para as variáveis de avaliação; **Limites Mínimos** são os valores mínimos que são determinados para as variáveis de avaliação; **Valor a Alcançar** é o valor limite que a sua função deverá atingir para satisfação do resultado, além dos campos citados acima foi disponibilizado um “*check box*” **Plotar gráficos** caso o usuário queira plotar e visualizar a evolução dos resultados da função avaliação até atingir seu objetivo.

Figura 2 Interface Gráfica do Algoritmo Evolucionário Diferencial



Fonte: Os autores deste trabalho

Ao executar o programa teremos uma interface didática e intuitiva em que o usuário apenas preenche os campos que estão livres, podendo optar pelo plote do gráfico ou não ao calcular os coeficientes. Assim, é proporcionado uma maior comodidade e diminuídas as possibilidades de erro quando se altera os parâmetros diretamente no código do algoritmo.

3.2 Desenvolvendo e rodando a aplicação

O programa elaborado foi desenvolvido para resolver diversos tipos de funções, ou seja, é um programa genérico que possibilita o usuário a escolher a função de avaliação em que se deseja resolver (tais função de segundo grau, PID...), não limitando assim o usuário a resolução de apenas um tipo de problema.

Para demonstrar uma aplicação, foi escolhido para exemplificar essa evolução diferencial um controlador PID aplicado a uma função de transferência.

A função de transferência escolhida para utilização como exemplo no programa foi:

$$F(s) = \frac{2s + 4}{1s^3 + 3s^2 + 5s + 1} \quad (1)$$

A solução proposta na função de avaliação é tal que tendo a função de transferência mais o controlador em malha fechada, a função de custo avaliada é o erro entre o set point e a saída do controlador. O limite deste parâmetro deve ser preenchido no campo “Valor a alcançar”. Esse é o valor mínimo do erro em que é satisfeita a necessidade do equacionamento. Foi escolhido um valor de 0,01 neste caso. Ou seja, após as interações, quando o controlador PID da função transferência em malha fechada atingir a condição de erro menor que 0,01, o programa irá parar e exibir os parâmetros dos coeficientes proporcional, derivativo e integral.

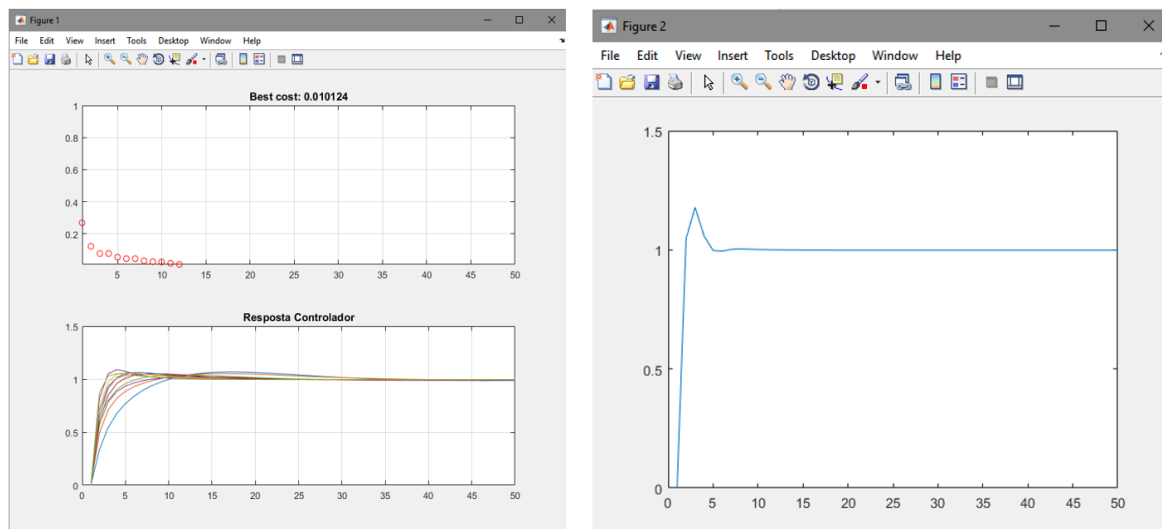
Ao preencher os limites máximos e mínimos na interface gráfica, pode-se definir o tipo de controlador que se deseja utilizar, se ambos os campos possuírem somente um valor, o controlador será definido como somente proporcional (P), caso hajam dois valores será um proporcional integral (PI) e se forem inseridos três valores o controlador será um proporcional integral derivativo (PID).

Pode-se avaliar que o programa exibe evolução das soluções do PID em malha fechada da função de transferência, representado pelas linhas em azul e do erro representado pelos círculos vermelhos, onde consegue se observar uma evolução em função do tempo tendendo (o erro) a 0,01 conforme definido na interface gráfica no campo “Valor a alcançar”.

Também é exibido pelo programa o controlador encontrado, limitado pelos parâmetros que foi imposto pelo usuário através da variável “FVr_x”. Como foram inseridos três valores, foram retornados todos os parâmetros de um controlador PID, neste caso $P = 7,9$, $I = 111$ e $D = 8,4$.

Na Figura 3 é exibido o resultado do sistema em malha fechada com controlador PID utilizando os coeficientes definidos pelo algoritmo evolutivo diferencial.

Figura 3 Resposta do controlador encontrado em relação a função transferência inicial



Fonte: Os autores deste trabalho

Na tabela 1 é possível verificar a evolução do algoritmo na busca de melhor solução para resultado dada a função de avaliação.

Tabela 1: Evolução da Solução da função de avaliação

Interação	Erro	P	I	D
1	0.122893	3.6635	5.44227	2.45094
2	0.077899	4.48934	7.88693	1.28246
3	0.077899	4.48934	7.88693	1.28246
4	0.055551	4.36198	9.2354	2.27349
5	0.046376	4.53652	13.3016	2.45095
6	0.046376	4.53652	13.3016	2.45095
7	0.032502	5.01873	18.7028	3.46077
8	0.026902	5.913	19.553	2.82263
9	0.025632	5.98319	25.2465	2.91803
10	0.016984	6.78042	39.8627	3.79873
11	0.010124	8.99059	39.8627	4.19823
12	0.008834	7.91943	111.779	8.43992

Fonte: Os autores deste trabalho

4 CONCLUSÕES

Neste trabalho foi apresentado o desenvolvimento de uma interface gráfica no ambiente GUIDE que, a partir de uma função de avaliação carregada pelo usuário, permite a parametrização do código no MATLAB® do algoritmo evolutivo diferencial, apresentando como resultado a melhor população gerada de coeficientes, além do gráfico correspondente.

O algoritmo ED foi eficiente para a função de avaliação aplicada. Além disto, o código do algoritmo ED foi implementado de forma genérica permitindo, assim, o uso de outras funções de avaliação, podendo apresentar resultados igualmente satisfatórios.

O exemplo de aplicação foi a busca por parâmetros de um controlador PID, que forneceu os melhores ganhos Proporcional, Integral e Derivativo para minimizar o erro em estado estacionário. O menor erro desejado foi fornecido pelo usuário ao algoritmo de busca, através da interface.

O desenvolvimento de uma interface gráfica amigável favorece o processo de aprendizagem do usuário relativo ao funcionamento do algoritmo ED, pois esta ferramenta facilita a análise cognitiva de diferentes parametrizações com diferentes estratégias de uma função de avaliação.

A utilização da ferramenta permite ao usuário uma flexibilidade muito grande na resolução de problemas. Como em qualquer algoritmo evolutivo, um ponto chave é a determinação da função custo ou função de avaliação que se deseja avaliar para propor a solução de determinado problema. Ao avaliar os resultados de forma gráfica, é bastante útil ao usuário a avaliação se os resultados apontados pelo algoritmo, realmente são capazes de serem utilizados como solução adequada ao problema proposto pelo próprio usuário.

Agradecimentos

Ao Instituto Federal do Espírito Santo – Campus Serra por todo apoio no desenvolvimento deste trabalho.

REFERÊNCIAS

CHAKRABORTY, U. K. **Advances in differential evolution**. St. Louis: Springer, 2008.

GOLDBERG, D. E. “**Genetic Algorithms in Search, Optimization and Machine Learning**”, Addison-Wesley 1989.

LINDEN, R. **Algoritmos genéticos**. Rio de Janeiro: Brasport, 2006.

LOPES, Heitor Silvério; TAKAHASHI, Ricardo Hiroshi Caldeira. **Computação evolucionária em problemas de engenharia**. OMNIPAX,, 2011.

PACHECO, M. A. C. “**Algoritmos Genéticos: Princípios e Aplicações**”. ICA – Laboratório de Inteligência Computacional Aplicada, Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro, 1999.

STORN, R.; PRICE, K. **Differential Evolution - a Simple and Efficient Adaptive Scheme for Global Optimization**, 1995

STORN, R.; PRICE, K. **Differential Evolution (DE) for Continuous Function Optimization**, 1996. Disponível em: <<http://www.icsi.berkeley.edu/~storn/code.html>>. Acesso em: 10 Abr. 2018

WANG, Yong; CAI, Zixing; Z., Qingfu. **Differential evolution with composite trial vector generation strategies and control parameters**. IEEE Transactions on Evolutionary Computation, v. 15, n. 1, p. 55-66, 2011.

DEVELOPING A GRAPHIC INTERFACE TO SOLVE PROBLEMS USING DIFFERENTIAL EVOLUTION ALGORITHM

Abstract: *This document proposes the development of a graphic user interface to get support to the user on entering data and interpreting output data results on solving engineering problems proposed using differential evolutionary algorithm (ED). The utilization of the interface allows flexibility on the resolution of the problems using the ED, once it would be possible using evaluation functions provided by the user on an interactive manner through the interface. Therefore would be possible using the user program by solving different proposed problems and making use on different input data parameters defined by the user. These work also contributes on teaching and making the ED well known, due to the clear understanding of the procedure of the execution of the algorithm that is presented to the user, making it clear the results analysis.*

Key-words: *Evolutionary Computing, differential evolution, graphic interface*