ALGORITMO GENÉTICO DE CODIFICAÇÃO REAL APLICADO À OTIMIZAÇÃO DE FUNÇÕES DE *BENCHMARK*

Victor Honorato Cunha
Pontifícia Universidade Católica de Goiás
Departamento de Engenharia, Av. Universitária, 1440, Área III, Goiânia, Goiás - 74605010
victor.puc.engel@gmail.com

Emerson de Souza Campos
Pontifícia Universidade Católica de Goiás
MEPROS, Av. Universitária, 1440, Área III, Goiânia, Goiás - 74605010
emersoncampos.engel@gmail.com

Lucas de Carvalho Guimarães
Pontifícia Universidade Católica de Goiás
Departamento de Engenharia, Av. Universitária, 1440, Área III, Goiânia, Goiás - 74605010
lucascg1995@gmail.com

Maria José Pereira Dantas
MAF - Pontifícia Universidade Católica de Goiás
MEPROS, Av. Universitária, 1440, Área III, Goiânia, Goiás - 74605010
mjpdantas@gmail.com

RESUMO

O trabalho compara resultados obtidos por um algoritmo genético elitista de codificação real, com trabalhos da literatura que utilizam diferentes técnicas e operadores, na otimização das seguintes funções de *benchmark*: Goldstein-Price, Rosenbrock, Easom, Zakharov e Shubert. Dois parâmetros são utilizados na avaliação de eficiência: o número de avaliações da função objetivo e a taxa de sucesso de convergência. Para a visualização da complexidade das funções utilizadas é fornecido a sua representação matemática e gráfica, juntamente com um gráfico de convergência, que mostra a evolução da solução no decorrer das gerações do algoritmo genético. O algoritmo convergiu em cinco das sete funções utilizadas, com taxa de sucesso entre 47% e 90%. A comparação com outros trabalhos mostra a adequação dos algoritmos genéticos na solução de problemas de otimização, e também a necessidade do uso de técnicas específicas adjacentes para a solução de cada problema.

PALAVRAS CHAVE. Algoritmos Genéticos, Codificação Real, Funções de *Benchmark*.
TÓPICOS. MP (Modelos Probabilísticos); MH (Metaheuristicas).

ABSTRACT

This work compares the results obtained by a real-coded elitist genetic algorithm, with literature works that uses different techniques and operators, in the optimization of the following benchmark functions: Goldstein-Price, Rosenbrock, Easom, Zakharov and Shubert. Two parameters are used in the evaluation of efficiency: the number of objective function evaluations and convergence success rate. To the visualization of the complexity of the used functions is provided its mathematical and graphical representation, along with a convergence chart, which shows the evolution of the solution through the generations of the genetic algorithm. The algorithm converges in five of the seven functions used, with a success rate between 47% and 90%. Comparison with other studies shows the adequacy of genetic algorithms to solve optimization problems, and also the need to use specific adjacent techniques to the solution of each problem.

KEYWORDS. Genetic Algorithm, Real Coded, *Benchmark* Functions.
PAPER TOPICS. MP (Probabilistic Models); MH (Metaheuristics)

1. Introdução

A modelagem matemática de problemas reais de planejamento são utilizados modelos de otimização. Otimizar significa encontrar o equilíbrio entre o objetivo a ser alcançado e as restrições envolvidas, de forma que o resultado seja satisfatório.

Quando o espaço de busca onde estão contidas as soluções é relativamente pequeno é possível avaliar todas as soluções e apresentar a melhor depois dessa avaliação. Esses problemas são ditos como tratáveis, "em termos práticos, um problema é tratável se o seu limite superior de complexidade é polinomial" [Linden 2012]. Método da bisseção e método simplex são exemplos de modelos determinísticos utilizados na resolução de problemas tratáveis. Por outro lado, a grande maioria dos problemas tem o seu nível superior de complexidade de ordem exponencial, o que caracteriza um problema intratável. Um exemplo clássico é o do caixeiro viajante [Couëtoux *et al.*, 2010].

Na resolução de problemas intratáveis (NP-Completos), existem métodos heurísticos tendo base na natureza e conceitos sociais [Mollinetti 2013]. Não avaliam totalmente o espaço de busca, no entanto utiliza-se de uma informação anterior para chegar a uma boa solução. Esses algoritmos existem porque as resoluções de problemas de ordem exponencial por métodos determinísticos levariam um tempo muito grande para serem solucionados [Linden 2012], fora do limite aceitável. Os métodos heurísticos são utilizados quando as ferramentas determinísticas tornam-se inviáveis, isso geralmente ocorre quando o número de variáveis envolvidas é consideravelmente grande [Cirino et al. 2015]

Algoritmos Genéticos (AG's), objetivo de estudo deste trabalho, faz parte desses algoritmos supracitados. São dotados para encontrar soluções baseadas no processo biológico de evolução e seleção natural de Charles Robert Darwin [Linden 2012], onde, por meio de operadores genéticos, as características favoráveis são evidenciadas em gerações sucessivas e as características desfavoráveis tendem a ser minimizadas ou extinguir-se.

Os AG's têm se mostrado eficientes na solução de muitos problemas como, por exemplo, na engenharia [Lienig e Thulasiraman 2007], [Peralta 2013], [Chang et al. 2010], ciência da computação [Gordon 1988], ciências sociais [Fisher 2008], medicina [Tosatti et al. 2008] e outras. O grande número de aplicações em diversas áreas se justifica por sua facilidade de implementação, por sua performance em sua busca por melhores soluções e pelo fato de não serem fundamentalmente limitados por restrições do espaço de busca [Goldberg 1989].

Os algoritmos genéticos são propostos de acordo com o problema que irão resolver [Doerr et al. 2012]. No ano de 1975 Jhon Holland criou os AG's de codificação binária. Seus trabalhos foram popularizados por seu aluno Goldberg [Goldberg 1989]. Mais tarde Kenneth A De Jong [Jong 1975] em sua tese de doutorado analisa o comportamento dos algoritmos genéticos em diversas situações utilizando funções reais, mostrando a eficiência dos AG's.

A fim de avaliar a eficiência de algoritmos, existem as chamadas funções de *Benchmark* [Molga 2005], que incluem também as funções utilizadas por De Jong [Jong 1975]. Elas oferecem um meio pelo qual esses algoritmos probabilísticos podem ser avaliados. O intuito deste trabalho é submeter um algoritmo genético a essas funções e comparar os resultados obtidos com outros trabalhos da literatura, contribuindo com novos resultados e eficiência de técnicas utilizadas; conteúdo indispensável para pesquisadores da área que desejam avaliar a eficiência de seus algoritmos.

Na seção 2 é apresentada uma breve revisão da teoria dos AG's. A descrição das escolhas feitas para o desenvolvimento do algoritmo é apresentada na seção 3. A seção 4 define como são realizados os experimentos. Os resultados são apresentados na seção 5. Por fim, na seção 6 é feita uma análise seguida de discussão do algoritmo genético desenvolvido.

2. Origem

O termo genético vem de Darwin e sua teoria da evolução das espécies descrita em sua obra "A Origem das Espécies". Onde é dito que a evolução dos indivíduos ocorre através da seleção natural, ou seja, o indivíduo que melhor se adapta ao meio em que vive tem mais chances de sobreviver e repassar suas características para a próxima geração; sendo que os indivíduos menos adaptados ainda podem se reproduzir, no entanto, com menor sucesso. A seleção é feita a partir de vários acontecimentos na vida de um indivíduo de uma determinada população, como, por exemplo, disputa por alimento.

Em 1975 o propósito de John Holland em utilizar a evolução natural como método de resolução de problemas de otimização, resultou nos algoritmos genéticos, que por sua vez são uma simulação computacional da evolução natural. Esses algoritmos são uma técnica heurística, baseados na experiência, utilizam-se desta informação para se guiar no conjunto das possíveis soluções de um problema. Através de operados genéticos, tais como seleção, cruzamento e mutação; realizam uma busca aleatório-direcionada no espaço de busca aleatória para encontrar soluções "otimizadas".

As possíveis soluções são representadas por cromossomos, isto é, população de indivíduos (pais) que submetida a certas modificações, geram novos candidatos (filhos) com a tendência de melhorar a desempenho de algo em direção a algum ponto ou pontos otimizados [Serrada 1996].

A estreita relação dos AG's e a genética faz com que muitos termos sejam utilizados na teoria dos algoritmos genéticos. A Tabela 1 faz uma analogia com a biologia, referente a tais termos.

Tabela 1 – Terminologia utilizada nos algoritmos genéticos

Biologia	**Algoritmo Genético**
Cromossomo	Indivíduo
Gene	Característica
Alelo	Valor de cada variável
Genótipo	Estrutura
Fenótipo	Conjunto de parâmetros
Indivíduo	Solução candidata de um problema
População	Conjunto de indivíduos de uma geração
Geração	Ciclo de criação e de transformação de um problema
Adequabilidade	Função de avaliação

Fonte: Adaptado de Calixto, 2008

2.1. Codificações Binária e Real (ou decimal)

Existem duas codificações tradicionais nos AG's, a binária e a real. A codificação binária foi utilizada nos trabalhos de John Holland, sendo, portanto, historicamente importante. Nesse tipo de codificação os dados são armazenados em cadeias de bits (cromossomos). Quando os dados são parâmetros discretos, a codificação binária é a mais indicada; sua simplicidade de análise também justifica o seu uso.

Quando as variáveis do problema são contínuas, é mais eficiente se utilizar a codificação real. No espaço de busca contínuo a precisão da solução é motivo de atenção; ao representarmos as soluções através de números reais, a precisão é aumentada facilmente pela simples adição de mais casas decimais. Já na codificação binária se quisermos aumentar a precisão em uma casa, devemos acrescentar 3,3 bits em cada indivíduo.

É de fundamental importância entender que o comportamento dos algoritmos genéticos é influenciado diretamente pelos seus operadores e parâmetros. Os operadores genéticos fundamentais são a seleção, o cruzamento e a mutação, enquanto que, o tamanho da população inicial, a taxa de cruzamento e a taxa de mutação são os parâmetros genéticos mais importantes. [Catarina e Bach 2003].

A dificuldade de se utilizar cadeias binárias para representação de um espaço contínuo é principalmente o seu impacto no tempo de processamento. Os dados contínuos devem ser codificados e decodificados, demandando maior tempo de execução. E muitos dos problemas a serem resolvidos têm o seu domínio no espaço multidimensional, por exemplo: se utilizarmos 22 bits para cada variável de decisão do problema, os indivíduos terão $22 \cdot D$ bits, sendo *D* o número de variáveis; e novamente o desempenho do AG binário é reduzido.

3. Metodologia

3.1. Codificação e Funcionamento

Cada codificação possui os seus operadores genéticos. O algoritmo desenvolvido (AG_Real) neste trabalho para o teste de desempenho é de codificação real, sendo assim, os operadores a seguir apresentados, são mais comumente utilizados em codificação decimal. Porém seus princípios e conceitos podem ser utilizados para qualquer codificação.

O AG_Real foi implementado em MATLAB®, por ser uma linguagem de alto nível dedicada a cálculos matemáticos, e sua simplicidade de programação devido à utilização de matrizes, e segue o modelo inicial proposto por Holland, exceto pela inclusão da técnica de elitismo Figura 1. O algoritmo é dedicado à maximização de funções reais, cujo domínio esteja no R^n, ou seja, abrange funções multivariáveis [Jamil and Yang 2013].

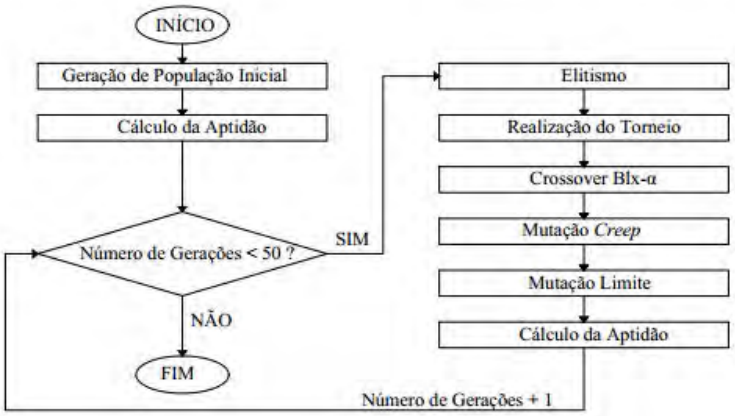


Figura 1 – Fluxograma AG_Real

3.2. Seleção

Utilizada em vários trabalhos [Miller and Goldberg 1995] a seleção por torneio foi escolhida como critério de seleção, ela determina, em parte, a pressão de seleção do AG_Real, pois os indivíduos melhores adaptados ao meio, dizemos os que possuem boas aptidões, têm maiores chances de serem escolhidos para o *crossover* (cruzamento). São escolhidos *n* cromossomos da população (conjunto das possíveis soluções do problema) aleatoriamente e com mesma probabilidade. O indivíduo com maior aptidão é então selecionado para o *crossover*.

3.3. *Crossover*

O *crossover* é o meio pelo qual os bons indivíduos, escolhidos na seleção, repassam sua característica para a próxima geração. Em uma analogia com o meio natural, o *crossover* pode ser comparado à reprodução sexuada dos seres. No AG_Real o cruzamento é feito pelo *crossover* BLX-α.

O *crossover* BLX-α tem uma interessante propriedade. A localização do novo indivíduo depende da diferença entre os pais. Se a diferença entre eles é pequena, a diferença entre pais e filhos é pequena também e vice-versa [Chen e Wang 2011].

Os cromossomos pais (**p1** e **p2**) possuem números decimais, cada qual relacionado a uma variável da função utilizada, e são submetidos à equação (1):

$$\boldsymbol{c = p1 + \beta * (p2 - p1)} \quad (1)$$

Tradicionalmente é definido $\alpha = 0{,}5$, ele estende em 0,5 unidades o intervalo entre os dois cromossomos pais, ou seja, os que irão participar do cruzamento. O parâmetro β aleatoriamente escolhido dentro do intervalo $[-\alpha; 1 + \alpha]$, esse processo ameniza a tendência de que os filhos sejam gerados próximos ao centro do intervalo entre os dois pais. Esse tipo de *crossover* explora melhor o espaço de busca das funções em comparação com outros operadores de cruzamento [Ariyarit e Kanazaki 2015].

Nem todos os indivíduos passam pelo operador de *crossover*. Existe uma probabilidade relacionada a esse parâmetro e o processo é repetido até formar todos os indivíduos (filhos) da próxima geração que ainda podem passar pelo operador de mutação.

3.4. Mutação

A mutação é responsável por evitar que a população permaneça apenas em uma pequena parte do espaço de busca, ou ainda, que a solução encontrada seja local e não global. O AG_Real utiliza duas mutações: *creep* e limite.

A mutação *creep* é uma mutação local onde cada gene do cromossomo (valor numérico relacionado a cada variável), é multiplicado por um número (aleatório) próximo de 1, no intervalo $[0{,}95; 1{,}05]$. De modo que, se o indivíduo está próximo do máximo global, ele irá rapidamente encontrar a melhor solução do problema. Por não ser muito destrutiva, a mutação *creep* pode ser associada a uma probabilidade relativamente alta. É adequada para representações reais dos genes [Soni e Kumar 2014].

Já a mutação limite é altamente destrutiva, ela escolhe um indivíduo da população e move-o para o limite inferior ou superior do domínio considerado da função. Assim, a probabilidade da mutação limite deve ser baixa, para evitar que se tenha perda de progresso durante as gerações. As probabilidades são definidas de forma que o processo de busca não se torne totalmente aleatório.

3.5. Elitismo

O elitismo é um mecanismo que mantém bons indivíduos nas gerações futuras. É utilizado para assegurar que as soluções sejam iguais ou melhores que as alcançadas anteriormente, contribuindo para uma melhor convergência do AG_Real. Para isso, o melhor indivíduo é mantido na próxima geração sem sofrer alteração dos operadores genéticos na próxima geração [Herrera et al. 1998].

4. Experimentos

A avaliação do AG_Real é feita a partir de sua aplicação em funções *benchmark* através de comparação com outros trabalhos da literatura. As funções de *benchmark* são funções em que seus máximos ou mínimos já são conhecidos. Elas são um meio pelo qual o desempenho dos AG's pode ser avaliado. Outra vantagem de se utilizar essas funções, é que elas "desafiam" os algoritmos genéticos; pois oferecem uma variedade de topologias diferentes na sua superfície, de maneira que a robustez, ou seja, o balanço entre eficiência e eficácia para que ele obtenha bons resultados em diferentes "meio ambientes" também pode ser avaliada [Goldberg 1989].

Por ser um algoritmo probabilístico, mesmo encontrando boas soluções para um determinado problema, os AG's podem não encontrá-las novamente, ainda que as condições (parâmetros) de execução sejam mantidas. Dessa forma, realizar apenas um teste para cada função, não garante a robustez do algoritmo, já que pode-se obter diferentes soluções a cada execução.

Os parâmetros utilizados nas simulações são mostrados na Tabela 2. Esses valores foram escolhidos após a realização de vários testes [Bessaou e Siarry 2001].

Tabela 2 – Configuração dos parâmetros.

Parâmetros	Configurações
Tamanho da População	50
Número de Competidores no Torneio	3
Probabilidade de *Crossover* (Blx-0,5)	90%
Probabilidade de Mutação (Creep)	40%
Probabilidade de Mutação (Limite)	0,5%
Número de Gerações	50
Elitismo	1

Cada função é testada cem vezes. É considerada a taxa de sucesso do AG_Real, que fornece porcentagem de sucesso de convergência do algoritmo no universo dos cem testes realizados. É considerado convergência quando a lacuna entre o mínimo global analítico da função e o mínimo encontrado pelo algoritmo atinge uma precisão de 3 casas decimais. A precisão considerada depende das exigências do usuário e da grandeza com a qual se trabalha, a aplicação em funções é geral, logo essa escolha foi feita através da observação dos resultados apresentados por Chelouah e Siarry [2000]. É feita uma média do número de avaliações da função objetivo de todas as execuções que convergiram, uma dessas execuções é aleatoriamente utilizada para demonstrar através de gráfico com deu a convergência do AG_Real para o mínimo global da respectiva função [Chelouah e Siarry 2000].

5. Resultados

Primeiramente é mostrada a formulação matemática, depois uma visualização no espaço das funções utilizadas seguido dos resultados encontrados, o gráfico de convergência e comentários. São sete funções no total. Uma tabela de comparação é apresentada no fim desta seção.

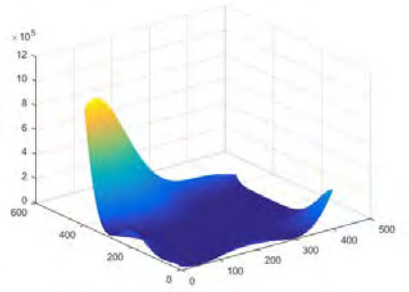
Função de Goldstein-Price (GP):

$$GP(x_1,x_2) = [1 + (x_1 + x_2 + 1)^2 \times (19 - 14x_1 + 13x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \times [30 + (2x_1 - 3x_2)^2 \times (18 - 32x_1 + 12x_1^2 - 48x_2 - 36x_1x_2 + 27x_2^2)] \quad (2)$$

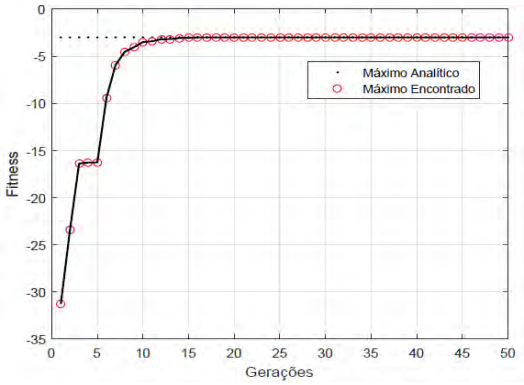
Domínio de busca: $-2 \leq x_j \leq 2,\ j = 1,2;$

4 mínimos locais e um mínimo global $(x1,x2) = (-1,0);$

Mínimo Global: $GP[(x_1,x_2)] = 3$



Taxa de sucesso: 87%
Avaliações da função objetivo: 695



(a) Representação Gráfica (b) Convergência

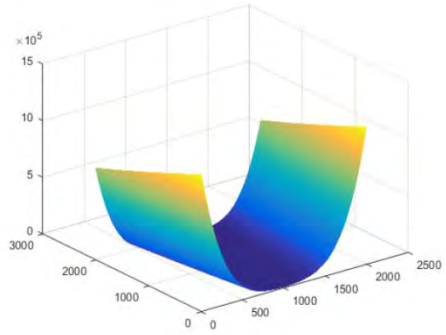
Figura 2 – Função Goldstein-Price

A função de Goldstein-Price (Eq. 2) (Figura 2a), tem duas variáveis e é uma função de teste de otimização global, pois possui apenas um mínimo global. O algoritmo apresenta bons resultados para esta função. A rápida convergência do algoritmo (Figura 2b) se deve ao fato de que o domínio considerado desta função é relativamente pequeno para o tamanho da população utilizada (cinquenta indivíduos), assim é possível varrer o espaço de busca como um todo. Encontrar o máximo desta função é simples, e pode-se utilizar técnicas como subida de encosta. Porém quando se busca sua minimização, ela apresenta um desafio para o algoritmo, pois possui 4 mínimos globais.

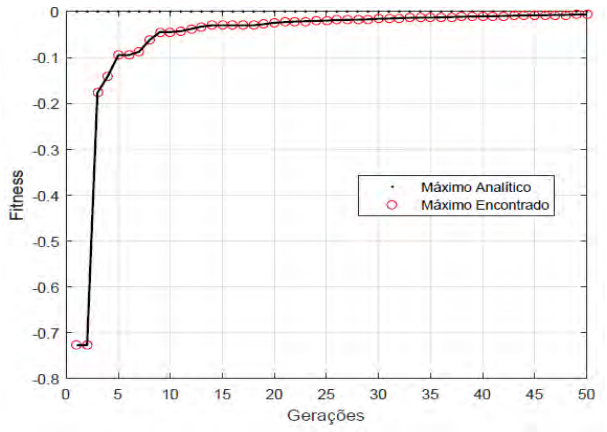
Função de Rosenbrock ($R_{n\ (variáveis)}$):

$$R_n(x) = \sum_{j=1}^{n-1} \left[100\left(x_j^2 - x_{j+1}\right)^2 + (x_j - 1)^2 \right] \qquad (3)$$

Utilizada nas três funções: R_2, R_5;
Domínio de busca: $-5 \leq x_j \leq 10,\ j = 1, \ldots n$;
1 *mínimo global*: $x = (1, \ldots, 1)$;
Mínimo Global: $R_n(x^*) = 0$.



Taxa de sucesso (n= 2): 47%
Avaliações função objetivo (n= 2): 610
Não convergiu para n= 5



(a) Representação Gráfica (b) Convergêcia

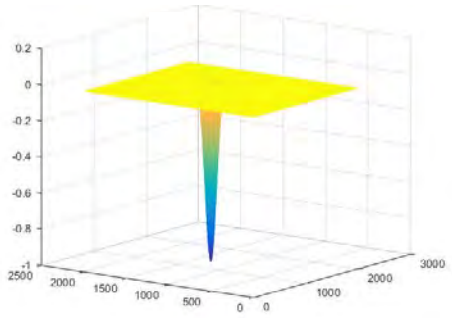
Figura 3 – Função Rosenbrock

A função de Rosenbrock (Eq. 3) (Figura 3a) é clássica e conhecida com função banana. O mínimo global encontra-se em um vale, longo, estreito, e de formato aplanado. Esta função de teste tem sido bastante utilizada para a avaliação de desempenho de algoritmos de otimização global [Cavalcanti 2004]. Para duas variáveis o AG_Real encontra o mínimo global em 47% dos casos, onde percebe-se uma dificuldade, e como esperado o algoritmo não convergiu para cinco variáveis pois quando se aumenta o número de variáveis aumenta também o espaço de busca, e como os parâmetros foram mantidos constantes para futura comparação, uma população de cinquenta indivíduos se torna pequena para varrer todo o espaço de busca. A convergência (Figura 3b) para o mínimo global é lenta devido á topologia da função, logo um aumento no número de gerações ajudaria no aumento da taxa de sucesso.

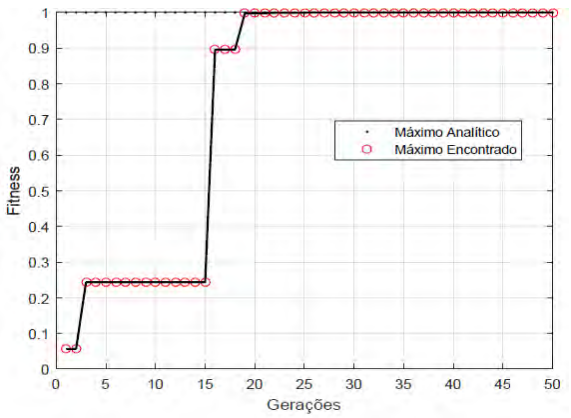
Função de Easom (ES):

$$ES(x_1,x_2) = -\cos(x_1) \times \cos(x_2) \times \exp\{-[(x_1-\pi)^2 + (x_2-\pi)^2]\} \quad (4)$$

Domínio de busca: $-100 \leq x_j \leq 100\ ,\ j = 1,2;$
1 mínimo global: $(x1,x2)^* = (\pi,\pi)$;
Mínimo Global: $ES[(x1,x2)^*] = -1.$



Taxa de sucesso: 73%
Avaliações função objetivo: 1040



(a) Representação Gráfica (b) Convergêcia

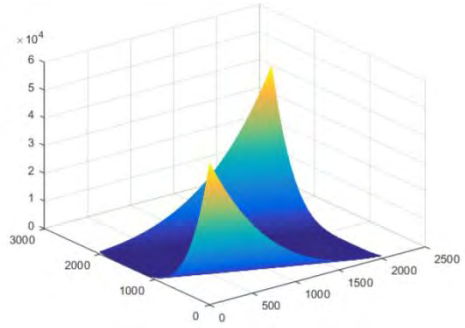
Figura 4 – Função Easom

A função Easom (Eq. 4) (Figura 4a) que é uma função unimodal, ou seja, possui apenas um mínimo global, pode parecer simples de otimizar, porém o seu mínimo se encontra apenas em uma pequena área com relação ao seu ao seu espaço de busca, o que dificulta a otimização. Quando alguns indivíduos da população encontram essa pequena região do domínio a convergência se dá de forma rápida como mostrado nas gerações de quinze a vinte do gráfico de convergência (Figura 4b).

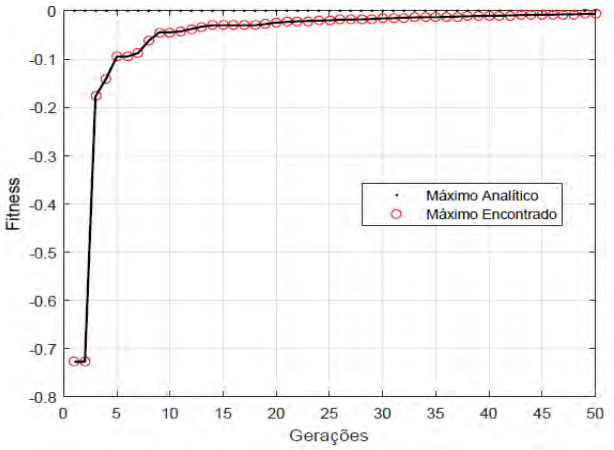
Função de Zakharov ($R_{n\ (variáveis)}$):

$$Z_{n(x)} = \left(\sum_{j=1}^{n} x_j^2\right) + \left(\sum_{j=1}^{n} 0{,}5j \times x_j\right)^2 + \left(\sum_{j=1}^{n} 0{,}5j \times x_j\right)^4 \quad (5)$$

Utilizada nas três funções: Z_2, Z_5;
Domínio de busca: $-5 \leq x_j \leq 10, \quad j = 1,\dots,n;$
1 mínimo global: $x^* = (0,\dots,0);$
$Z_n(x^*) = 0.$



Taxa de sucesso (n= 2): 90%
Avaliações função objetivo (n= 2): 540
Não convergiu para n= 5



(a) Representação Gráfica (b) Convergência

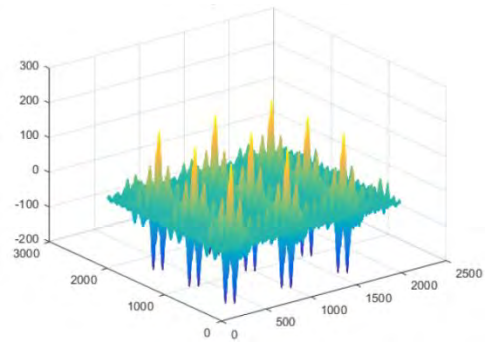
Figura 5 – Função Zakharov

A função de Zakharov (Eq. 5) (Figura 5a) é considerada relativamente simples para duas variáveis, como pode ser observado na Figura 5b porém a medida que aumentamos a quantidade de variáveis a topologia da função começa a ficar mais complexa assim também como o espaço de busca, dessa forma essa função com cinco variáveis representa um desafio multivariáveis para o algoritmo. Os melhores resultados do AG_Real foram obtidos nesta função com duas variáveis, porém a convergência não aconteceu para cinco variáveis devido a alta complexidade do problema.

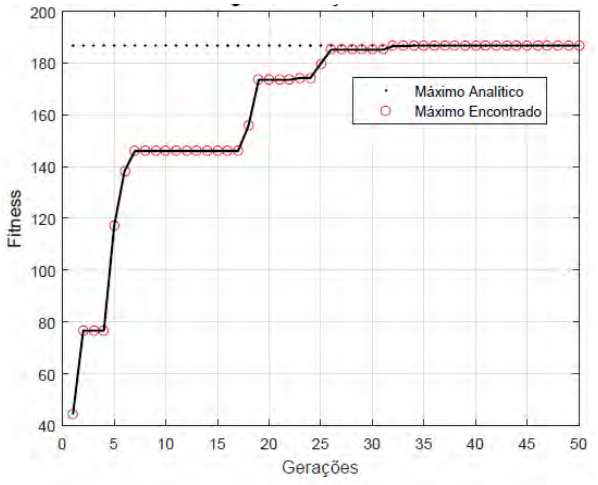
Função de Shubert (SH):

$$SH(x1,x2) = \left(\sum_{j=1}^{5} j \cdot \cos[(j+1)\cdot x1 + j]\right) \cdot \left(\sum_{j=1}^{5} j \cdot \cos[(j+1)\cdot x2 + j]\right) \quad (6)$$

Domínio de busca: $-10 \leq x_j \leq 10, j = 1,2;$
760 mínimos globais e 18 mínimos globais
Mínimo Global: $SH[(x_1,x_2)] = -186.7309$



Taxa de sucesso: 51%
Avaliações função objetivo: 1050



(a) Representação Gráfica (b) Convergência

Figura 6 – Função Shubert

Shubert (Eq. 6) é uma função de teste multimodal (Figura 6a) de duas variáveis. Funções multimodais como essa são utilizadas em testes de desempenho de otimizadores, pois muitos problemas de planejamento possuem a característica de multimodalidade. O AG_Real convergiu em 51% das execuções; para fins de comparação os parâmetros genéticos foram mantidos os mesmos durante todos os testes. Porém no caso da função de Shubert, um aumento do número de gerações nos levaria a um aumento da taxa de sucesso, bem como no de avaliações da função objetivo. O gráfico de convergência mostra o algoritmo saindo de mínimos locais, dando esse formato de "escada" para a Figura 6b.

Comparação AG_Real, CGA e RCGA

O AG_Real é comparado com dois outros algoritmos: o RCGA (Real-Coded Genetic Algorithm) de Bessaou e Siarry [2001] e o CGA (Continous Genetic Algorithm) de Chelouah e Siarry [2000]. Na Tabela 3 são mostradas duas informações, primeiro o número médio de avaliações da função objetivo ($FOBJ_{aval}$) e depois a taxa de sucesso (TS).

Tabela 3 - Comparação da média do número de avaliações da função objetivo

AG	**AG_Real**		**RCGA**		**CGA**	
Função	**$FOBJ_{aval}$**	**TS (%)**	**$FOBJ_{aval}$**	**TS (%)**	**$FOBJ_{aval}$**	**TS (%)**
GP	695	87	270	100	410	100
R_2	610	47	596	100	960	100
ES	1040	73	642	100	1504	100
Z_2	540	90	437	100	620	100
SH	1050	51	946	100	575	100
R_5	-	0	4150	60	3990	100
Z_5	-	0	1115	100	1350	100

Quando comparado ao RCGA, o AG_Real não conseguiu melhores resultados de taxa de sucesso e número de avaliações da função objetivo. Em relação ao CGA, o número de avaliações da função objetivo do AG_Real foi menor para as seguintes funções: Rosenbrock de duas variáveis, Easom e Zakharov de duas variáveis.

Ambos os algoritmos utilizam técnicas adjacentes para melhoria da aptidão de cada geração, tais como: composição de subpopulações baseado na teoria de entropia, decrescimento da probabilidade de mutação e da quantidade de indivíduos na população, existe a proposição de um novo método de *crossover*, técnicas de diversificação e intensificação de busca, etc. O AG_Real utiliza operadores simples, onde apenas o elitismo entra como técnica de auxilio à convergência.

As técnicas empregadas no RCGA e CGA, se justificam pela análise das funções Zakharov e Rosenbrock de cinco variáveis, pois o AG_Real não convergiu com os parâmetros configurados, logo técnicas como decrescimento da probabilidade de mutação é importante no controle da pressão de seleção.

6. Conclusão

Após revisão de literatura percebeu-se a importância dos otimizadores probabilísticos. O algoritmo genético elitista de codificação real desenvolvido neste trabalho convergiu para cinco das sete funções de *benchmark* utilizadas na análise, com taxa de sucesso entre 47% e 90%. A implementação foi realizada com operadores simples, onde apenas o elitismo entra como técnica de auxilio à convergência.

Os experimentos de comparação serviram para a análise da adequação dos algoritmos à complexidade das funções e mostrar a necessidade do uso de outras técnicas de acordo com a complexidade de cada problema. Os algoritmos selecionados para a comparação utilizaram técnicas adjacentes tais como composição de subpopulações baseado na teoria de entropia,

decrescimento da probabilidade de mutação e da quantidade de indivíduos na população, novo operador de *crossover*, técnicas de diversificação e intensificação de busca.

Os resultados apresentados para sete funções de *benchmark,* através de uma formulação matemática, sua representação gráfica, gráfico de convergência, taxa de sucesso e número de avaliações da função objetivo, permitem uma melhor visualização dos tipos de problemas de otimização e, também, da adequação dos algoritmos genéticos para resolvê-los.

As funções de *benchmark* representam neste artigo a complexidade de problemas reais, e os três AG's comparados demonstram que os algoritmos genéticos podem ser escolhidos como ferramenta nas soluções de problemas complexos, que envolvem muitas variáveis e restrições, para os quais as ferramentas determinísticas tornam-se inviáveis.

Agradecimentos: Este trabalho teve apoio financeiro da Fundação de Amparo à Pesquisa do Estado de Goiás (FAPEG).

Referências:

Ariyarit, A. e Kanazaki, M. (2015). Multi-modal distribuition *crossover* base on two crossing segments bounded by selected parents applied to multi-objetive design optimization. Journal of Mechanical Science and Technology (29) 4 p. 1443-1448.

Bessaou, M. e Siarry, P. (2001). A genetic algorithm with real- value coding to optimize multimodal continuous functions. Struct Multidisc Optim, v. 23, n. 1, p. 63-74. ISSN 1615-147X.

Calixto, W. P. (2008). Aplicação do Mapeamento Conforme no Cálculo do Fator de Carter. Dissertação (Mestrado em Engenharia Elétrica) – Escola de Engenharia Elétrica e de Computação. Goiânia: Universidade de Goiás.

Catarina, A. S. e Bach, S. L. (2003). Estudo do efeito dos parâmetros genéticos sobre a solução otimizada e sobre o tempo de convergência em algoritmos genéticos com codificações binária e real. Acta Scientiarum Technology: Maringá, v. 25 n. 2, p. 147 -152.

Chang, L.-C. et. al. (2010). Constrained genetic algorithms for optimizing multi-use reservoir operation. Journal of Hydrology. Elsevier B.V, v. 390, p. 66-74.

Chelouah, R. e Siarry, P. (2000). A Continuous Genetic Algorithm Designed for the Global Optimization of Multimodal Functions. Journal of Heuristics, v. 6, n. 2, p. 191-213. ISSN 1381-1231.

Chen, Z.-Q. e Wang, R.-L. (2011). Two efficient real-coded genetic algorithms for real parameter optimization. International Journal of Innovative Computing, Information and Control – ICIC, vol. 7, n. 8, ago.

Cirino, R. B. Z. et al. (2015). Aplicação da metaheurística agc para o problema de alocação de aulas à salas. XLVII Simpósio Brasileiro de Pesquisa Operacional.

Couëtoux, B. et al. (2010). Labeled Traveling Salesman Problems: Complexity and approximation. Discrete Optimization, v. 7, n. 1-2, p. 74-85. ISSN 15725286.

Doerr, B. et al. (2012). More effective *crossover* operators for the all-pairs shortest path problem. Theoretical Computer Science. ISSN 0304-3975.

Fischer, I. (2008). "Using Genetic Algorithms for Simulation of Social Dilemmas" in New Issues and Paradigms in Research on Social Dilemmas, pp. 252-264.

Goldberg, D. E. (1989). Genetic Algorithms in Search, Optimization, and Machine Learning. Addison Wesley Publishing Company, Inc. Universidade do Alabama, Reading, MA, 412.

Gordon, M. (1988). Probabilistic and genetic algorithm in document retrieval. Communications of the ACM, v. 31 p. 1208-1218.

Herrera, F. et al. (1998). Tackling Real-Coded Genetic Algorithms: Operators and Tools for Behavioural Analysis. Artificial Intelligence Review, v. 12, n. 4, p. 265-319. ISSN 1573-7462. Available at: < http://dx.doi.org/10.1023/A:1006504901164 >.

Holland, J. H. (1975). Adaptation in Natural and Artificial Systems. MIT Press.

Jamil, M. e Yang, X.-S. (2013) A literature survey of *benchmark* functions for global optimisation problems. International Journal of Mathematical Modelling and Numerical Optimisation, v. 4, n. 2, p. 150-194. ISSN 2040-3607.

Jong, K. A. D. (1975). An analysis of the behavior of a class of genetic adaptive systems. 266 University of Michigan

Lienig, J. e Thulasiraman, K. (2007). A Genetic Algorithm for Channel Routing in VLSI Circuits. Evolutionary Computation, 1(4):293-311.

Linden, R. (2012). Algoritmos Genéticos. 3ª ed. Ciência Moderna.

Tosatti M. A. et al. (2008). Algoritmo híbrido genético-fuzzy aplicado em câncer de próstata, Hífen, 32 (62) (2008) 164-171.

Cavalcanti, A. M. (2004). Desenvolvimento e análise de algoritmos probabilísticos de otimização global.

Miller, B. L. e Goldberg, D. E. (1995). Genetic algorithms, tournament selection, and the effects of noise. Complex Systems, v. 9, n. 3, p. 193-212. ISSN 0891-2513.

Molga, M. e Smutnicki, C. (2005). Test functions for optimization needs.

Mollinetti, M. A. F. et. al. (2013). Análise de Perfomance do Imperialist Competitive Algorithm através de Funções de *Benchmark*. In: Encontro Anual de Tecnologia da Informação e Semana Acadêmica de Tecnologia da Informação – EATI, ano 3, n. 1. Frederico Westphalen: EATI, p. 13-21.

Peralta, R. et al. (2014). Multiobjective genetic algorithm conjunctive use optimization for production, cost, and energy with dynamic return flow. Journal of Hydrology, n. 511, p. 776-785.

Serrada, A. P. (1996). Una Introducción a la Computación Evolutiva. Universidade de Oviedo, Espanha.

Soni, N. e Kumar, T. (2014). Study of Various Mutation Operators in Genetic Algorithms.In: International Journal of Computer Science and Information Technologies. Universidade de Lingaya´s. Faridabad: vol. 5, n. 3, p. 4514-4521.