OTIMIZAÇÃO EVOLUTIVA DE UMA REDE NEURAL ASSOCIADA AO MÉTODO DE EVOLUÇÃO DIFERENCIAL APLICADA NO SISTEMA DE ANCORAGEM DE PLATAFORMAS FLUTUANTES

Thalita Mongarde Daer[a]
Edivaldo Ramos Delgado[b]
Bruno da Fonseca Monteiro[b]
Juliana Souza Baioco[a, b]

thalita_daer@hotmail.com

edivaldodelgado@poli.ufrj.br

bruno.monteiro@poli.ufrj.br

jsbaioco@id.uff.br

[a] TEQ/ E.Engenharia/UFF – Departamento de Engenharia Química e Petróleo - Escola de Engenharia – Universidade Federal Fluminense, Rua Passo da Pátria, 156, S 305, Bloco D, Campus Praia Vermelha, São Domingos, Niterói RJ, Brasil.

[b] POLI/UFRJ – Escola Politécnica, Universidade Federal do Rio de Janeiro, Cidade Universitária – Ilha do Fundão, Avenida Athos da Silveira Ramos, 149, CT – Bloco A, 2º andar, Cep 21941-909, Rio de Janeiro – RJ – Brasil.

Abstract. *As plataformas flutuantes estão sujeitas à ação de cargas ambientais, em diferentes direções e intensidades. O sistema de ancoragem é responsável pela integridade da plataforma, além de limitar os deslocamentos provocados pelas cargas ambientais a níveis convenientes. Uma maneira de analisar a eficiência desse sistema é feita por uma ferramenta numérica computacional baseada em elementos finitos, a qual define as tensões resultantes a partir de termos de séries temporais.*

Porém, essa ferramenta requer a execução de análises estáticas e dinâmicas que possuem alto custo computacional. Além disso, a matriz de casos de carregamento pode incluir centenas de combinações de carga. Com o intuito de reduzir os custos computacionais, este artigo sugere a aplicação de uma rede neural artificial (RNA) para estimar os offsets (passeios), e a tração aplicada sobre a linha de ancoragem, admitindo como entrada a combinação dos parâmetros que configuram um sistema de ancoragem (raios, ângulos, tensão aplicada, e o tipo de material da linha). Tomando como princípio uma RNA já treinada, o foco será encontrar um menor número de avaliações da função objetivo para treinar a rede, através do software Matlab, e, em seguida, aplicá-las ao algoritmo de evolução diferencial (Differential Evolution - DE), otimizando assim a função objetivo, diminuindo o tempo empregado para análise e os custos computacionais.

Keywords: *RNA, Sistema de Ancoragem, DE, Sistemas Flutuantes*

1 INTRODUÇÃO

1.1 Motivação e Objetivo

Com o avanço das atividades de exploração de petróleo em profundidades cada vez maiores, como nos campos do Pré-sal, alcançando profundidades de até 4000 m, o investimento em estudos e pesquisas em estruturas e sistemas *offshore* se tornou crucial nos dias atuais, a fim de minimizar os custos dessas estruturas complexas, além de manter a segurança das operações.

Diante desse cenário, viu-se a necessidade de desenvolver métodos que diminuam os custos e ao mesmo tempo proporcionem resultados adequados. Dessa forma, a utilização de métodos evolutivos (no caso, o método de otimização por Evolução Diferencial - DE), associado à evolução de uma rede neural, para estimar os deslocamentos da plataforma e as trações aplicadas nas linhas de ancoragem, podem levar à substituição de exaustivas análises paramétricas por projetos mais simples e eficientes.

Haja visto o dado contexto, este trabalho apresenta o desenvolvimento de uma RNA para estimar os deslocamentos da plataforma e as trações aplicadas nas linhas de ancoragem, usando parâmetros de configuração dos sistemas de ancoragem, como o raio, ângulo, e tipo de material usado na linha de ancoragem. Após treinar a RNA, iremos otimizá-la pelo método DE a fim de obter resultados com melhores propriedades de convergência, e, portanto, exigindo menos avaliações das soluções candidatas.

1.2 Plataformas Flutuantes de Produção de Petróleo *Offshore*

Os sistemas flutuantes são sistemas complacentes, pois caracterizam-se por manifestar grandes deslocamentos sob ação de cargas ambientais (ação das ondas, correntes e ventos); sendo assim, é crucial a aplicação de um sistema de ancoragem adequado, que limite tais deslocamentos a níveis convenientes (Lacerda, 2005).

Tais sistemas podem ser classificados de acordo com o tipo de casco, em navios ancorados ou plataformas semissubmersíveis, conectados a *risers* que transportam óleo, gás e outros fluidos.

Já o sistema de ancoragem abrange um conjunto de linhas de ancoragem confeccionadas por diferentes tipos de materiais, e estacas ou âncoras, que transmitem os esforços que atuam sobre a plataforma para o solo. As linhas de ancoragem devem garantir a integridade de todo o sistema flutuante, além de influenciarem diretamente o sistema de *risers* (afetados pelo passeio da embarcação), logo, a grande importância no estudo de atividades *offshore*.

Altamente suscetíveis às cargas ambientais, e também a efeitos não lineares severos devido aos grandes deslocamentos que sofrem, as linhas de ancoragem devem ser projetadas com base em parâmetros de sua resposta estrutural, como as tensões. Estes parâmetros são definidos em termos de séries temporais, geralmente obtidos a partir de ferramentas complexas como a de Elementos Finitos (FE), as quais requerem um elevado custo computacional.

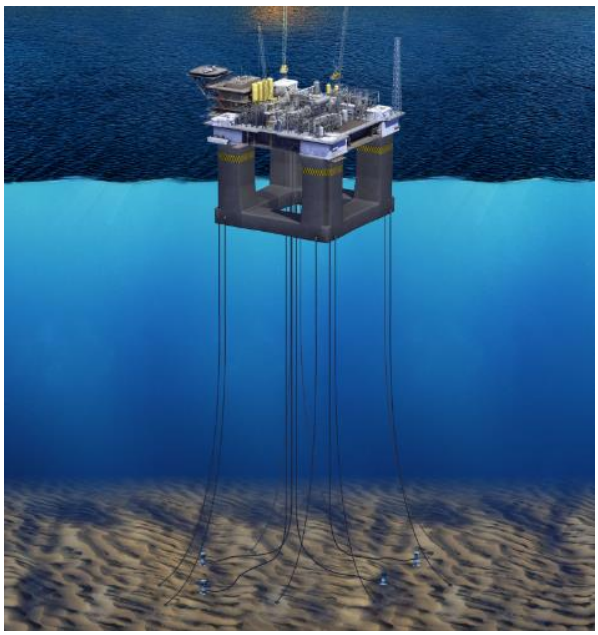

Figura 1. Plataforma Semissubmersível
Fonte: Repositório Digital da Petrobras

2 REDES NEURAIS ARTIFICIAIS (RNA)

Baseando no fato que o sistema de ancoragem e os *risers* sejam avaliados como um sistema integrado real, que engloba a interação entre o comportamento estrutural hidrodinâmico de todas as linhas (malhas FE) e o comportamento hidrodinâmico do casco, as análises acopladas se tornam mais precisas e revelam com maiores detalhes o comportamento global do sistema.

No entanto, a simulação acoplada tem alto custo computacional, o que leva à substituição por modelos como as RNA's (Redes Neurais Artificiais), que possuem um custo computacional mais baixo, menor tempo de análise, além de fornecer resultados compatíveis se bem treinadas.

2.1 Neurônios Artificiais e Função Ativação

Na indústria de petróleo as Redes Neurais Artificiais têm sido aplicadas em problemas de engenharia oceânica e offshore, incluindo a previsão das características do estado do mar (Yasseri et al., 2010), a estimativa dos passeios da plataforma sob cargas ambientais (Mazaheri; Downie, 2005), e muitas outras aplicações offshore (Ok et al., 2007).

Os RNA's são compostos por complexas redes de unidades simples (neurônios) interconectadas. Cada unidade recebe várias entradas e gera apenas uma saída, a qual pode se tornar uma das entradas de outra unidade. No sistema biológico, os neurônios são células responsáveis pela recepção, transmissão e processamento de sinais. Individualmente, realizam operações relativamente simples; contudo, as conexões entre eles geram uma enorme diversidade de tarefas. A Figura 2 ilustra um modelo artificial de neurônio – o neurônio de McCulloch-Pitts (McCulloch et al., 1943).

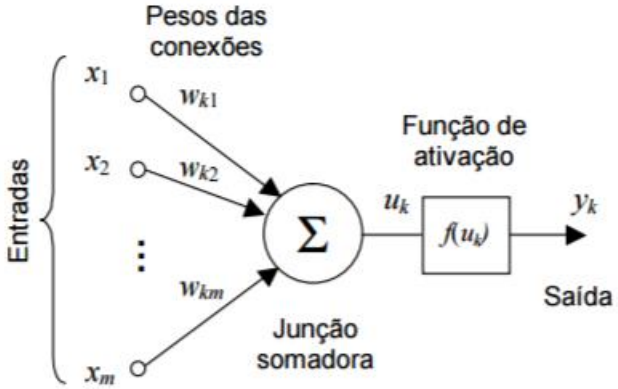

Figura 2. Neurônio de McCulloch-Pirrsk.
Fonte: Guo B., Song S. e Chacko J., 2005

Ao receber um dado número de entradas x_i, x = 1, N, primeiramente ele calcula uma combinação linear dessas entradas usando pesos sinápticos w_i (a cada entrada está associada a um peso w_i que corresponde a importância da entrada x_i) para gerar a entrada ponderada u_k, como visto na Eq. (1).

$$u_k = \sum_{i=1}^{N} w_i x_i\,(t) \qquad (1)$$

Em seguida, fornece uma saída y_k através de uma função de ativação - $f(u_k)$ que deve apresentar um comportamento monotônico crescente sobre uma determinada faixa de valores para u_k, e assumir um valor constante fora dessa faixa. Vários tipos de função ativação podem ser usadas (Haykin, 2001), incluindo a função logística definida pela Eq. (2) (com um parâmetro α que modifica a derivada da vizinhança de $u_k=0$, para ajustar a "velocidade" de transição):

$$y_k = f(u_k) = \frac{1}{1+ e^{-\alpha u_k}} \qquad (2)$$

2.2 Camadas

Em uma rede neural, os neurônios se organizam em camadas, onde todos os neurônios da mesma camada possuem o mesmo padrão de conexão para receber e transmitir sinais. As redes neurais podem ser simples ou multicamadas.

Embora mais complexas, as redes multicamadas apresentam melhores resultados. A maioria das aplicações de RNA utiliza redes de 2 camadas; apenas uma camada oculta, com um número adequado de neurônios, pode ser suficiente para aproximar funções contínuas; com duas camadas, até mesmo funções descontínuas podem ser representadas (Cybenk, 1989).

A disposição dos neurônios na rede pode ser feita em relação ao método de propagação da informação recebida, conforme mostrado na Figura 3. Pode-se diferenciar entre redes de propagação para frente (*feedforward*) e redes realimentadas (*recurrent*). No caso das redes de propagação para frente, o fluxo de informações é unidirecional. Os neurônios que recebem a informação conjuntamente organizam-se em camadas.

Já as redes realimentadas possuem ligações entre os neurônios sem restrições. Oposta às redes sem alimentação, o comportamento dinâmico desempenha o papel fundamental desse modelo. Em alguns casos, os valores de ativação da rede passam por um processo de relaxação até alcançarem um estado estável.

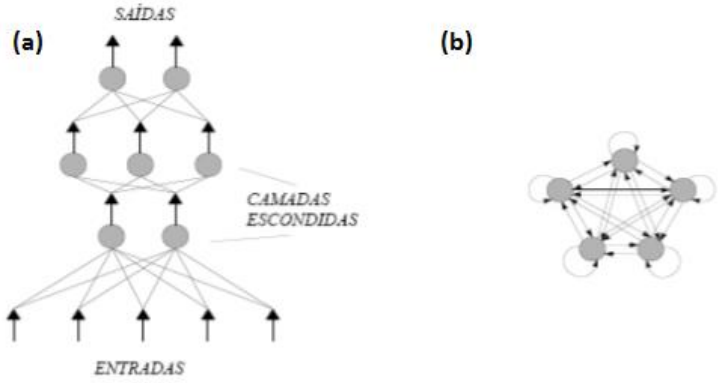

Figura 3. Tipos de Alimentação de RNA's: (a) propagação para frente; (b) redes realimentadas

Fonte: Guo B., Song S. e Chacko J., 2005

2.3 Treinamento

Após determinar a topologia da rede neural, a mesma tem que ser treinada, ou seja, os graus de liberdade, que a rede dispõe para solucionar a tarefa em questão, têm que achar um valor ótimo.

O treinamento da rede neural refere-se ao processo de ajustes de peso – w_i da Eq. (1), de modo que, a escolha de um conjunto de entradas produz um conjunto desejado de saídas. Isso depende da disponibilidade de um conjunto de treinamento, isto é, um conjunto de pares entrada-saída conhecidos, que é apresentado à rede várias vezes (Pina et al., 2013).

O método de treinamento por retropropagação (*backpropagation*) é o mais comum para redes neurais de multicamadas. Consiste em usar os erros das saídas para atualizar todos os pesos, a partir do fim para o início da rede (de Pina et al., 2013).

Cada interação na retropropagação consiste em 2 etapas: no passo ascendente (ou para frente), uma entrada é aplicada à rede neural e o efeito é propagado através das conexões entre os neurônios, até o momento que uma resposta é produzida; no passo retrógrado, a resposta obtida é comparada com a saída desejada, o erro é calculado e retropropagado da saída para a entrada, ajustando os pesos de conexão. O procedimento é repetido até que um critério de paragem seja atingido.

3 OTIMIZAÇÃO PELO MÉTODO DE EVOLUÇÃO DIFERENCIAL

A evolução diferencial (DE) é um poderoso algoritmo de otimização global introduzido por Rainer Storn e Kenneth Price em 1995 (Storn e Price, 1995). DE surgiu a partir das tentativas de Price para resolver um problema de ajuste polinomial de Chebychev.

Algoritmos de pesquisa evolutiva, incluindo DE, tentam imitar a evolução natural de uma população de indivíduos. O DE é um dos melhores algoritmos de tipo genético para resolver problemas com variáveis representadas por um número real, além de ser eficaz para funções objetivo que não são diferenciáveis ou convexas, e grande facilidade na busca do ótimo com populações pequenas (Cheng e Hwang, 2001).

Devido à sua estrutura simples, facilidade de uso, velocidade e robustez, essa ferramenta é empregada em várias aplicações na engenharia, para descobrir soluções eficazes sem utilizar algoritmos de design complexos.

3.1 O Problema de Otimização

Similar a outros Algoritmos Evolutivos, a otimização pelo método DE se baseia na ideia da evolução de uma população de possíveis soluções candidatas. No decorrer das gerações, essas soluções candidatas sofrem operações de *mutação* e *cruzamento (crossover)*, onde são geradas novas soluções candidatas, e logo depois é realizada a *seleção*, e o ciclo se repete (Araujo, 2016). Parte dessas novas soluções são melhores que as soluções anteriores, e logo após um determinado número de iterações, um ótimo local ou global pode ser alcançado.

Mutação

Na operação de *mutação*, novos vetores de parâmetros são gerados, chamados de vetores doadores ou modificados, através da adição da diferença vetorial ponderada entre dois vetores aleatórios da população a um terceiro indivíduo, conforme representado abaixo pela Eq. (3).

$$v_{novo} = v_{base} + F\,(v_1 - v_2) \qquad (3)$$

Onde v_{novo} é o vetor doador, F é o fator de mutação e é responsável por manipular a amplitude da diferença entre dois indivíduos (quanto maior for esse fator, maior será o salto e mais rápido o algoritmo chega a um ótimo, contudo, as chances de não encontrar um máximo ou mínimo na função objetivo são maiores), e v_{base}, v_1 e v_2, são vetores escolhidos aleatoriamente, e mutuamente distintos.

Cruzamento (*Crossover)*

Após o processo de mutação, escolhe-se aleatoriamente outro vetor, vetor alvo (v_{alvo}), cujas componentes são misturadas com as componentes do vetor doador (gerado pelo processo de mutação), formando o vetor experimental. Essa operação é introduzida na população a fim de aumentar a diversidade dos indivíduos que sofreram mutação (Price, 1999).

O vetor experimental u_i^j é formado pela seguinte Eq. (4):

$$u_i^j = \begin{cases} v^j + F(v_{r1}^j - v_{r2}^j) & \text{se } r_i \leq C \\ v_i^j, & \text{caso contrário.} \end{cases} \tag{4}$$

onde r_i é um número aleatório, *CR* (taxa de cruzamento) é um valor real dentro de um intervalo e é informado pelo usuário, v_i^j são as componentes do vetor alvo, o qual competirá com o novo vetor formado. Para assegurar a realização da mutação em pelo menos uma variável, seleciona-se aletoriamente à geração de um novo indivíduo, um dos componentes do mesmo, chamado *jRand*, no qual a mutação é feita independente da probabilidade *CR*.

Quanto maior for a taxa de cruzamento, maior a possibilidade de se obter mais dimensões do vetor doador no vetor experimental. Deve ser escolhida uma taxa alta para aumentar a variedade da população e assim obter melhores resultados.

Seleção

No processo de seleção há a escolha do melhor indivíduo entre o vetor pai x_i^t e o vetor experimental u_i^t para formar a nova população da geração $t + 1$. Se a aptidão determinada através do cálculo da função objetivo do indivíduo *i* da população corrente ($\vec{x}_{i,\,G+1}$) é maior do que a aptidão do indivíduo *i* da população de cruzamento ($\vec{u}_{i,\,G+1}$), esse indivíduo passa para próxima geração com os melhores entre as duas populações, conforme a Eq. (5).

$$\vec{x}_{i,\,G+1} = \begin{cases} \vec{u}_{i,\,G+1}, & \text{se } f(\vec{u}_{i,\,G+1}), \leq f(\vec{x}_{i,\,G+1}). \\ \vec{x}_{i,\,G+1} = \vec{x}_{i,\,G}, & \text{caso contrário.} \end{cases} \tag{5}$$

4 ESTUDOS DE CASO: OTIMIZAÇÃO DO SISTEMA DE ANCORAGEM

Visando demonstrar a potencialidade e permitir uma avaliação entre os métodos explicitados, iremos abordar inicialmente os resultados encontrados na RNA treinada e, posteriormente, a evolução através do método de evolução diferencial, que irá utilizar como dados iniciais os resultados encontrados na RNA. Ambos os algoritmos foram modelados no *software* MATLAB.

4.1 Janela Inicial para a RNA

Na aplicação considerada neste trabalho, a configuração básica implementada na RNA foi a rede de propagação para frente, com uma camada intermediária (ou camada oculta), e uma camada de saída, correspondente ao resultado desejado. Vale ressaltar que, a primeira camada com as unidades de entrada não é contabilizada, visto que não é realizado nenhum cálculo.

Os dados iniciais para o treinamento da RNA foram obtidos através de uma análise de movimento acoplado com o programa não comercial *SITUA-PROSIM* (2006), que gera, como resultado, uma matriz de 19 colunas e 8249 linhas, sendo as 10 colunas iniciais compostas pelo conjunto de *inputs* (parâmetros de configuração dos sistemas de ancoragem, como o raio, ângulo, e tipo de material usado na linha de ancoragem) como mostrado na Tabela 1, e as outras 9 colunas, os *offsets* (passeios) e a tração máxima aplicada à linha de ancoragem, representadas na Tabela 2.

Coluna	Parâmetro	Mínimo	Máximo
01 - 04	Ângulo por corner	-3°	3°
05 – 08	Raio por corner	-500 m	500 m
09	Tração média	1000 kN	3000 kN
10	Material (diâmetro)	0,122 m	0,262 m

Tabela 1. *Inputs* – Parâmetros de configuração do sistema de ancoragem

O software incorpora ferramentas numéricas para definir os dados de entrada: cargas ambientais, dados de base batimétrica, configurações das linhas de ancoragem e propriedades do casco.

Coluna	Parâmetro - Direção	Mínimo	Máximo
11	*Offset 1 – N*	0,122	0,262
12	*Offset 2 – NE*	92,7090	297,147
13	*Offset 3 – E*	90,32	259,924
14	*Offset 4 – SE*	87,066	204,363
15	*Offset 5 – S*	106,842	269,241
16	*Offset 6 – SW*	117,028	303,542
17	*Offset 7 - W*	96,864	194,09
18	*Offset 8 - NW*	106,121	163,849
19	*Tração Máxima*	0,164	0,963

Tabela 2. Conjunto de dados de treinamento da RNA

As simulações estáticas são realizadas sob a ação de cargas ambientais atuando em 8 direções de incidência (N, NE, E, SE, S, SW, W, NW), para fornecer as saídas (*offsets* e tração sob as linhas de ancoragem) que compõem o conjunto de dados de treinamento da RNA.

4.2 Treinamento e Validação da Rede

O procedimento para os estágios de treinamento e validação da rede podem ser feitos após a janela inicial dos parâmetros desejados forem coletadas pelo método FE. Uma vez que a rede é treinada, para obter as saídas, a rede precisa de novos parâmetros. Esse mecanismo pode ser sintetizado na Tabela 3.

1. Execute análises estáticas pelo método FE (com diferentes quantidades de indivíduos), para obter um conjunto de *inputs* e um conjunto de *outputs* (*offsets* e tração na linha de ancoragem). 2. Use esse conjunto de dados para treinar e validar a rede. 3. Use a rede treinada para prever os *offsets* e a tração máxima aplicada à linha de ancoragem.

Tabela 3. Procedimentos para treinar a rede

Na 1ª etapa, treinamos diferentes quantidades de indivíduos (200, 1000, 1500 e 2000), todos com 10 neurônios, executados em 20 rodadas, nas quais obtivemos 4 conjuntos de dados (um para cada quantidade de indivíduo), com diferentes erros normalizados para a melhor rodada das 20 realizadas para cada *offset* e para a tração máxima.

O processo de treinamento fornece os pesos da RNA, que podem então ser aplicados para analisar novas configurações do sistema de ancoragem, além de formarem a função objetivo que será otimizada pelo DE.

4.3 Tratamento de Dados

A técnica da Evolução Diferencial faz uso de uma população de soluções candidatas para o equacionamento do problema a fim de explorar todo o espaço de busca. Uma vez gerada a população inicial, a função objetivo de cada indivíduo da população inicial é calculada e armazenada para futuras comparações com as novas gerações formadas através dos operadores de: mutação, cruzamento e seleção. A sequência de operações do algoritmo da Evolução Diferencial pode ser representada pela Tabela 4.

No problema em questão, a meta é minimizar a função objetivo, ou seja, o mínimo global para o erro obtido através da função *fitness*, e, para alcançarmos tal propósito, usaremos a técnica do DE sem restrições, onde o critério de parada adotado foi a de um determinado número máximo de gerações (no caso, 2000 gerações). Analisando a Tabela 4, ao final da 11ª etapa, o algoritmo retorna para a 4ª etapa até que todas as 2000 gerações sejam alcançadas, e por fim, obter um resultado com melhores padrões de otimização.

1. Início da rotina
2. Geração da população inicial
3. Geração da função objetivo para cada indivíduo da população inicial
4. Executa rotina de Otimização para população inicial até que o número de gerações proposto seja atingido
5. Seleciona indivíduos da população inicial aleatoriamente
6. Realiza operação de Mutação
7. Realiza operação de Crossover (cruzamento)
8. Geração de população teste a partir dos dados obtidos da mutação e crossover
9. Geração da função objetivo para cada indivíduo da população teste
10. Geração da população final a partir dos melhores indivíduos da comparação
11. Fim da rotina de Otimização da população corrente
12. Seleciona o melhor indivíduo da população final

Tabela 4. Esquema simplificado do algoritmo de Evolução Diferencial

Na aplicação do método DE apenas 3 parâmetros precisam ser ajustados: tamanho da população, fator de perturbação (F), e probabilidade de cruzamento (Cr). Segundo Storn (1996), a probabilidade de cruzamento Cr ϵ [0,1], se a convergência não for alcançada, utilizar um fator entre 0,8 e 1 o que tornará conveniente. Em relação ao número da população, quanto maior for, menor deve ser o fator de perturbação. O fator de perturbação F não deve ser menor que um certo valor para conter a convergência prematura, sendo este dependente da função objetivo, e que elevados valores de F conduzem a um maior risco de escapar de um ótimo local.

4.4 Resultados

Dentro dos intervalos recomendados pela literatura foram realizadas uma série de testes, variando os valores dos parâmetros e analisando os respectivos resultados da otimização. Porém, é importante que se adquira um maior conhecimento a respeito da escolha dos valores de ajustes dos parâmetros, pois para um determinado tipo de problema a estratégica pode funcionar perfeitamente, mas para outro não tão bem (Storn e Price, 1995).

Diante disso, diminuímos o valor de F e de Cr, alcançando assim um erro otimizado melhor que o encontrado pelo RNA, como mostra a Tabela 5.

A análise do erro normalizado pelo RNA, para um mesmo número de indivíduos (200 indivíduos) e do erro otimizado pelo método DE se baseou no estudo dos resultados dos *offset*s e da tração máxima, porém iremos abordar de forma detalhada apenas os dados do *offset 1*. Para a tração máxima o erro obtido pelo RNA na melhor rodada foi de $3{,}39 \times 10^{-3}$, enquanto que, para o DE, o melhor valor encontrado foi de $2{,}51 \times 10^{-3}$.

Nº de Partículas	Gerações	Média (DE)	Melhor Rodada (DE)	F	CR
25	2000	8,85	8,85	0,4717	0,8803
50	2000	4,17	4,16	0,4717	0,8803
75	2000	2,62	2,62	0,4717	0,8803
100	2000	3,28	3,17	0,4717	0,8803
200	2000	2,16	2,16	0,2	0,8
300	2000	2,16	2,16	0,2	0,8

Média (RNA)	Melhor Rodada (RNA)
3,06	2,178

Tabela 5. Erros obtidos pelo RNA e pela otimização do DE

Utilizamos um limite superior e um limite inferior, 10 e -10, criando um espaço de busca com soluções fisicamente possíveis, o qual permitiu encontrar o melhor resultado para uma análise de 200 partículas, como mostra a Figura 4, convergindo a um mesmo valor o erro da melhor rodada e o da média entre as rodadas, com erro igual a 2,16.

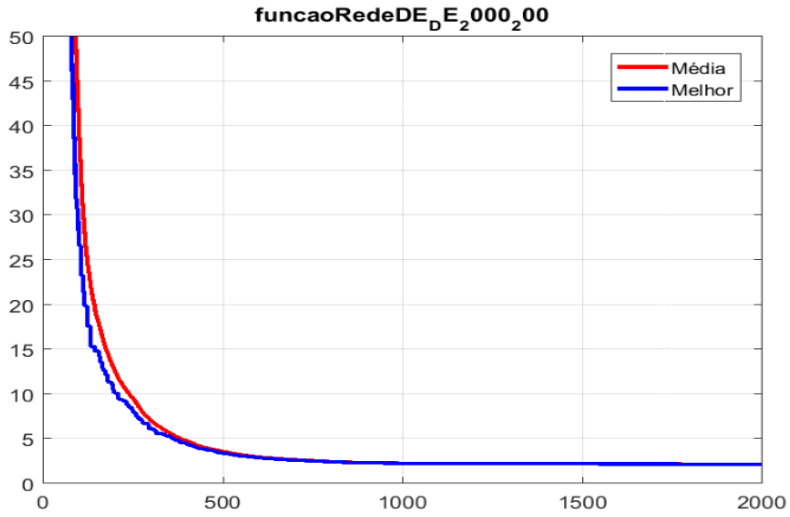

Figura 4. Erros otimizados obtidos através do método DE

4 CONCLUSÃO

Pela Figura 5 podemos observar que o gráfico em que os pontos mais se convergem, ou seja, obtém-se um menor desvio, é o gráfico com os pontos formados pela otimização através do algoritmo DE. Porém, analisando a Tabela 5, vimos que mesmo aumentando o número de indivíduos (200 para 300 indivíduos) e, mantendo os outros parâmetros, não obtivemos sucesso em relação à minimização da função objetivo, o que comprova que embora populações maiores possam conduzir a uma maior probabilidade de encontrar o mínimo global, elas implicam em um número maior de avaliações da função objetivo e assim a um maior custo computacional, que nem sempre é necessário.

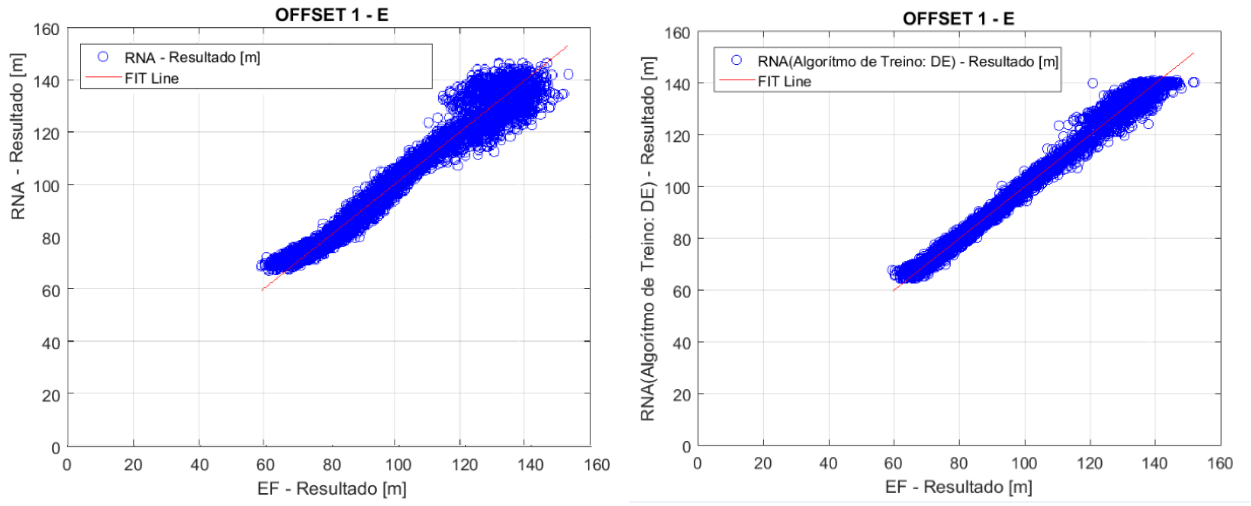

Figura 5. Resultados obtidos pela RNA e pela otimização pelo método DE, respectivamente

Portanto, o resultado do método desenvolvido e apresentado comprovou com sucesso a evidência na melhoria do erro obtido, alcançado pela execução de uma população de 200 indivíduos, com um tempo de execução satisfatório, tornando a aplicação viável.

Para estudos futuros, podem ser usados outros algoritmos de otimização para serem comparados com os resultados obtidos pelo método DE.

5 REFERÊNCIAS

ARAUJO, R. L., 2016. *Evolução Diferencial para Problemas de Otimização com Restrições Lineares*. Dissertação de mestrado, UFJF/MMC, Juiz de Fora, MG, Brasil.

CHENG, S.-L., HWANG, C., Nov 2001. *Optimal approximation of linear systems by a differential evolution algorithm*. Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on 31 (6), 698–707.

CYBENK, G., 1989. *Approximation by superpositions of a sigmoidal function. Mathematics of Control*. Signals and Systems; 2:303–14.

HAYKIN, S., 2001. *Neural networks – a comprehensive foundation*. New Jersey: Prentice Hall.

LACERDA, T. A.G.,2005. *Análise de Sistemas de Ancoragem de Plataformas Flutuantes*. Projeto Final de Graduação, EP/UFRJ, Rio de Janeiro, RJ, Brasil.

MAZAHERI, S., DOWNIE, M.J., 2005. *Response-based method for determining the extreme behaviour of floating offshore platforms*. Ocean Eng; 32:363–93.

MCCULLOCH, W.S., PITTS W., 1943. *A logical calculus of the ideas immanent in nervous activities*. Bull Math Biophys; 5:115–33.

OK D., PU Y., INCECIK A., 2007. *Artificial neural networks and their application to assessment of ultimate strength of plates with pitting corrosion*. Ocean Eng; 34:2222–30.

PINA, A.C., de PINA, A.A., ALBRECHT, C.H., de LIMA, B.S.L.P., JACOB, B.P., 2013. *ANN-based surrogate models for the analysis of mooring lines and risers,* Appl Ocean Research, 41:76-86, <http://dx.doi.org/10.1016/j.apor.2013.03.003>.

PRICE, K. V., 1999. An introduction to differential evolution. New Ideas in Optimization, 79–108.

STORN, R., PRICE, K. V., 1995. Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization 11, 341–359.

YASSERI, S.F., BAHAI, H., BAZARGAN, H., AMINZADEH, A., 2010. *Prediction of safe sea-state using finite element method and artificial neural networks*. Ocean Eng;37:200–7.