

Um estudo comparativo da inversão sísmica poço-a-poço utilizando PSO e GA seriais e paralelizados

Arthur Anthony da Cunha Romão E Silva
 Centro de Ensino Superior do Seridó
 Universidade Federal do Rio Grande do Norte
 Caicó, Rio Grande do Norte
 arthur.romao.700@ufrn.edu.br

Francisco Márcio Barboza
 Departamento de Computação e Tecnologia - UFRN
 Universidade do Rio Grande do Norte
 Caicó, Rio Grande do Norte
 marcio.barboza@ufrn.br

Resumo—A tomografia poço-a-poço de tempo de trânsito envolve encontrar um modelo de vagarosidade que minimiza a energia de erro entre os tempos de trânsitos medidos e teóricos. Nesse sentido, este trabalho apresenta um estudo comparativo de problemas de tomografia poço-a-poço utilizando os algoritmos globais *Particle Swarm Optimization* (PSO) e *Genetic Algorithm* (GA) seriais e paralelizados. Primeiramente, uma revisão dessas duas técnicas de otimização é apresentada. A solução do problema de tomografia poço-a-poço utilizando vínculo de suavidade é apresentada com o objetivo de ilustrar a eficiência dessas metodologias na solução de problemas geofísicos inversos. Os resultados mostram que as versões paralelizadas são mais eficientes.

Palavras-chave—Tomografia Poço-a-poço, *Genetic Algorithm*, *Particle Swarm Optimization*, Paralelização

I. INTRODUÇÃO

A tomografia sísmica poço-a-poço é uma das diversas técnicas para inferência de dados geológicos através do processo de inversão de dados sísmicos existentes no interior da Terra, sendo que estes dados podem ser obtidos a partir de ondas mecânicas ou eletromagnéticas. Este processo consiste em estimar um modelo de vagarosidade (inverso da velocidade) a partir dos dados observados, que podem ser os tempos de trânsito das ondas acústicas ou o formato de onda [1].

Os dados de tomografia sísmica fornecem informações entre poços através da medição do tempo que leva para um sinal viajar de uma fonte localizada em um poço para um receptor localizado em um poço vizinho. A vantagem dos dados de tomografia é que, devido às suas localizações no fundo do poço, os caminhos dos raios evitam cruzar as camadas rasas intemperizadas no topo da superfície da terra, tais camadas sendo uma fonte de interferência. Também pode haver muitas fontes e muitos receptores e estes podem estar próximos uns dos outros [2]. Este tipo de dado pode ser usado para detectar

as principais heterogeneidades (falhas, descontinuidades) entre os poços. A tomografia poço-a-poço também é usada no desenvolvimento de campo, por exemplo, para detectar a mudança nas propriedades dinâmicas do reservatório ao longo do tempo de produção (sísmica 4D) [3].

Vários métodos de otimização global foram propostos para inverter dados geofísicos, incluindo tempos de viagens sísmicas [4, 5]. As abordagens mais populares usam diferentes variantes de *Simulated Annealing* (SA) ou *Genetic Algorithm* (GA). Por exemplo, [6] e [7] usam abordagens baseadas em SA para inversão sísmica de reflexão e os tempos de trânsito poço-a-poço, respectivamente. Recentemente alguns trabalhos utilizam *Particle Swarm Optimization* (PSO) para a inversão dos tempos de trânsito poço-a-poço, por exemplo [8, 9, 10]. Esses estudos indicam que diferentes abordagens globais são ferramentas viáveis para inverter os dados sísmicos de tempo de trânsito.

Entretanto em muitos problemas de otimização é necessário avaliar um grande espaço para encontrar o mínimo global, então o processo de cálculo é muito demorado [11]. Algumas das grandes desvantagens dos métodos de Otimização Global além de lidar com grandes espaços de busca é possuir um custo computacional extremamente alto.

O uso de algoritmos globais com paralelismo no problema de Otimização é uma das soluções para alto custo nas inversões. Esses algoritmos permitem que o uso de um espaço maior seja pesquisado de forma síncrona; eles podem usar mais memória para lidar com um espaço e dimensões maiores. Algumas outras razões para justificar sua paralelização são que eles reduzem a probabilidade de encontrar soluções locais e podem cooperar em paralelo com outra técnica de pesquisa [12].

A estratégia utilizada neste trabalho foi a aplicação do *Toolbox* de Paralelização do *Matlab*, pois proporciona de modo simplificado o auxílio da paralelização nas estruturas

de repetição, favorecendo um melhor uso das *threads* do processador, com o intuito de melhorar a performance dos algoritmos globais [13, 14, 15].

Este artigo apresenta um estudo comparativo de inversão sísmica poço-a-poço usando *Particle Swarm Optimization* (PSO) e *Genetic Algorithm* (GA). Para o estudo, usamos a versão serial e paralela de cada algoritmo, para que possamos analisar se o uso do paralelismo traz algum benefício em termos de desempenho, além do aumento de velocidade computacional.

II. REFERENCIAL TEÓRICO

A. Tomografia Sísmica Poço-a-poço

A tomografia sísmica poço-a-poço é uma técnica de inferência de dados que explora as informações contidas em registros sísmicos para restringir modelos 2D ou 3D do interior da Terra. Geralmente requer a solução de um grande problema inverso para obter um modelo sísmico heterogêneo que seja consistente com as observações. Mais formalmente, desde que possamos estabelecer uma relação aproximada $t = G(\mathbf{s})$ entre os dados sísmicos t e para um determinado modelo \mathbf{s} possamos prever t – então o problema da tomografia sísmica equivale a encontrar \mathbf{s} tal que t explica os dados observados t_{obs} . Na maioria das vezes esse problema é mal-posto, tradicionalmente usa-se regularização. Neste trabalho é usada regularização de Tikhonov (Suavidade)[16].

1) *Formulação Linear*: Vamos inicialmente considerar o problema inverso linear, onde G é operador de modelagem de raios retos determinado pela discretização do problema, \mathbf{s} é o vetor de parâmetros (modelo) de tamanho n , e t é o vetor dos tempos de trânsito observados.

$$G(\mathbf{s}) = t \quad (1)$$

O problema inverso seria resolvido para um conjunto de dados t_{obs} , minimizando uma função objetivo da forma:

$$\Phi = \frac{\|G(\mathbf{s}) - t_{obs}\|_2^2}{N_{obs}} + \lambda^2 \|D\mathbf{s}\|_2^2 \quad (2)$$

onde D é um operador de diferenças finitas, N_{obs} é o número de observações e λ é um parâmetro de regularização. O primeiro termo na equação 2 mede a desajuste dos dados, enquanto o segundo mede a suavidade do modelo.

B. Algoritmos de Otimização Global

Os métodos de otimização local não linear do tipo gradiente convergem rapidamente para um mínimo local, próximo à estimativa inicial, mas pode não encontrar uma solução global. Entretanto, os algoritmos de otimização global são capazes de

buscar mais efetivamente dentro de um espaço de busca um mínimo global [17].

Este artigo compara dois algoritmos de inversão usando otimização global serializada e paralelizada. O GA é baseado em computação evolutiva e usa iteração estocástica sem gradientes. O PSO é uma técnica de otimização estocástica baseada em população inspirada no comportamento social de bando de pássaros ou cardume de peixes para encontrar uma solução ideal em espaços de busca complexos.

1) *Genetic Algorithm*: GA (ver Algoritmo 1), originalmente introduzido por [18], pertence ao grupo dos métodos de otimização global estocástica. Este é baseado na evolução biológica, onde o modelo mais apto tem maior probabilidade de sobrevivência e reprodução, enquanto outros (o mais fraco, valor de aptidão inferior) têm probabilidades mais baixas.

Algoritmo 1: *Genetic Algorithm*

Parâmetros de controle: tamanho da população (p_s), taxa de *crossover* (p_c), taxa de mutação (p_m), sigma (σ), número de iterações (k);

Gerar população inicial P aleatoriamente;

Avaliar aptidão da função $\Phi(\mathbf{s})$ para cada partícula contida em P;

while $k > 0$ **do**

for $i = 1$ **to** k **do**

 número de cruzamentos $\gamma_c = (p_s/2)$;

for $j = 1$ **to** γ_c **do**

 selecionar aleatoriamente duas soluções \mathbf{s}_a e \mathbf{s}_b da população P utilizando seleção proporcional de aptidão;

 gerar \mathbf{s}_c e \mathbf{s}_d por meio de um cruzamento uniforme de \mathbf{s}_a and \mathbf{s}_b ;

 salvar \mathbf{s}_c e \mathbf{s}_d na população Pop_1 ;

end

for $j = 1$ **to** γ_c **do**

 selecionar uma solução \mathbf{s}_j da população Pop_1 ;

 efetuar mutação em cada bit de \mathbf{s}_j sob a taxa p_m e do sigma (σ);

 gerar uma nova solução \mathbf{s}_j ;

 atualizar \mathbf{s}_j com \mathbf{s}_j da população Pop_1 ;

end

 atualizar população P $\leftarrow Pop_1$;

end

$k \leftarrow k - 1$

end

retornar o melhor \mathbf{s} da população P;

De modo geral o GA possui ideias chaves e simples, os parâmetros físicos (genes) são combinados como cromossomos, que representam um estado do modelo (possível solução). A coleção de modelos constitui uma população, que possui um número de indivíduos (modelos de vagariedade em nosso

caso) gerados aleatoriamente. Após esta etapa, o problema direto é resolvido e a discrepância entre os dados observados e calculados (função objetivo) é estimada. A adequação é então usada no processo de seleção e cruzamento sucessivo. Os melhores indivíduos (ou seja, os modelos de vagarosidade com o menor erro) são selecionados para gerar os próximos genes (descendentes). A aptidão dos indivíduos é então usada no processo de seleção sucessivo.

Uma série de indivíduos selecionados aleatoriamente (ou seja, o modelo de vagarosidade com o menor erro) tem o direito de passar seus genes para a próxima geração. A informação genética trocada entre indivíduos selecionados é realizada pelo operador de *crossover*. Durante a mutação, *bits* selecionados de descendentes com base na probabilidade de mutação são invertidos para permitir pequenas perturbações dos parâmetros do modelo. Para passar os indivíduos mais aptos de cada geração para a próxima, a estratégia do elitismo é usada evitando a possível perda de bons indivíduos [19, 20, 21]. As iterações desta teoria da evolução podem ser encerradas após um número fixo de gerações.

No presente trabalho utilizamos uma versão simplificada do GA, e partindo do pseudocódigo, paralelizamos o algoritmo global utilizando um *Toolbox* de paralelização do *Matlab*, realizando de maneira prática, apenas alguns ajustes, em algumas estruturas essenciais do GA.

2) *Particle Swarm Optimization*: O PSO (ver Algoritmo 2) foi introduzido por [22], é um método de otimização global com base na população, inspirado no comportamento coletivo dos pássaros e nas interações sociais humanas. Os cientistas descobriram que a sincronia do comportamento de agrupamento era através da manutenção de distâncias ideais entre os membros individuais e seus vizinhos. Assim, a velocidade desempenha o papel importante de ajustar uns aos outros para a distância ideal. Além disso, os cientistas simularam o cenário em que os pássaros procuram alimento e observaram seu comportamento social. Eles perceberam que, para encontrar comida, os membros individuais determinavam suas velocidades por dois fatores; sua melhor experiência anterior e a melhor experiência de todos os outros membros. Isso é semelhante ao comportamento humano na tomada de decisão quando as pessoas consideram suas melhores experiências anteriores e a melhor experiência de como as outras pessoas ao seu redor se saíram [22].

Este possui uma população de soluções, denominadas partículas. Um enxame dessas partículas se move pelo espaço de busca para encontrar uma posição ótima (aqui, presume-se que a função deve ser minimizada). O movimento de uma partícula é influenciada pelo histórico de pesquisa pessoal da partícula e pelas melhores posições que foram encontradas por outras partículas vizinhas no enxame.

Cada *i*-ésima partícula é composta por sua posição, sua

Algoritmo 2: Particle Swarm Optimization

Parâmetros de controle: Tamanho da população (p_s), fator de inércia (ω), coeficiente de aprendizagem individual (cognitiva) (c_1), coeficiente transmissão cultural (social) (c_2), número de iterações (k);

Saída: melhor solução s ;

Gerar população inicial P aleatoriamente;

Avaliar aptidão da função $\Phi(s)$ para cada partícula contida em P ;

foreach partícula s_i em P **do**

$\Phi_{pbest}^i \leftarrow \Phi(s_i)$;
 $s_i^{pbest} \leftarrow s_i$;

end

$gbest \leftarrow$ melhor s_i em P ;

$k \leftarrow 1$;

while Critério de parada não for satisfeito **do**

foreach partícula s_i em P **do**

 Atualizar velocidade utilizando $v_i^k \leftarrow \omega v_i^{k-1} + c_1 \cdot r_1 (s_i^{pbest} - s_i^{k-1}) + c_2 \cdot r_2 (gbest - s_i^{k-1})$;

 Atualizar posição da partícula utilizando

$s_i^k \leftarrow s_i^{k-1} + v_i^k$;

 Avaliar aptidão da partícula por meio da função $\Phi(s_i)$;

if $\Phi(s_i^k) < \Phi_{pbest}^i$ **then**

$\Phi_{pbest}^i \leftarrow \Phi(s_i^k)$;
 $s_i^{pbest} \leftarrow s_i^k$;

end

if $\min \Phi(s_i^k) < \Phi_{gbest}$ **then**

$\Phi_{gbest} \leftarrow \min \Phi(s_i^k)$;
 $gbest \leftarrow s_i^k$;

end

$k = k + 1$;

end

end

retornar o melhor $s = gbest$;

velocidade e sua melhor posição já encontrada ($gbest$). A posição desta é a sua solução, que é um ponto em um espaço n -dimensional. A velocidade é dada por $v_i = (v_{i1}, v_{i2}, \dots, v_{in})$, e sua a posição por $s_i = (s_{i1}, s_{i2}, \dots, s_{in})$. Esta se moverá com base em sua posição inicial restrito a um domínio de busca determinado pelo problema de otimização [23].

O mecanismo de evolução de uma única partícula no PSO canônico pode ser descrito pelas seguintes equações:

$$v_i^{k+1} = \omega v_i^k + c_1 \cdot r_1 \cdot (s_i^{pbest} - s_i^k) + c_2 \cdot r_2 \cdot (gbest - s_i^k) \quad (3)$$

$$s_i^{k+1} = s_i^k + v_i^k \quad (4)$$

onde s_i^k é a partícula i na iteração k , v_i^k é a velocidade da partícula i na iteração k e s_i^{pbest} é a melhor posição encontrada pela i -ésima partícula até o momento. ω é o fator de inércia, cuja função é balancear a busca global e a local, r_1 e r_2 são números aleatórios uniformemente distribuídos no intervalo de

0 a 1. Além disso temos os fatores de aceleração da partícula c_1 e c_2 , que é o coeficiente de aprendizagem individual e o coeficiente transmissão cultural respectivamente.

No presente trabalho é utilizamos uma versão canônica do PSO, com os ajustes necessários com fins de gerar soluções de inversões em tomografia poço a poço utilizando *Toolbox* de paralelização do *Matlab*.

III. METODOLOGIA

A. Paralelização do GA e PSO utilizando Toolbox do Matlab

A estrutura básica de paralelização usada nos algoritmos globais foi o *parfor* disponibilizadas pelo *Toolbox* de Paralelização do *Matlab*. O *loop parfor* é uma construção de fluxo de controle paralelo. É uma construção de compartilhamento de trabalho que executa o corpo do *loop* de maneira independente da ordem sobre um conjunto de *workers* disponíveis [24, 13, 25, 15].

A paralelização tem como objetivo uma melhor distribuição das tarefas por diferentes núcleos do processadores, sendo que a ideia básica é que cada núcleo do processador possa criar novos indivíduos, realizar as operações heurísticas de cada algoritmo global e calcular suas aptidões em paralelo. Um dos fatores para melhorar a eficiência na busca da solução ótima destes algoritmos é determinado pelo tamanho da população, implicando no aumento do tempo de execução para que o algoritmo convirja. Sendo que a população de indivíduos podem ser aumentada ou diminuída de acordo com a disponibilidade e velocidade dos processadores.

Os componentes heurísticos do PSO que foram selecionados para paralelização foram os da atualização da velocidade da partícula, a criação de uma nova solução a partir de uma nova velocidade e seus respectivos ajustes, como também a avaliação paralela das soluções.

Já no GA, o paralelismo foi efetivado aos operadores genéticos cruzamento e mutação, que funcionam diretamente com indivíduos da população, visto que cada individuo sendo independente um do outro torna-se o fator principal quando se trata de paralelização de algoritmos, pois é desejável separar e criar estruturas e processamentos independentes.

B. Escolha dos parâmetros de controle e coeficientes da inversão

Ambos os algoritmos possuem parâmetros de controle em comum, pois são algoritmos bioinspirados, possuem características semelhantes, como a quantidade de indivíduos e a quantidade de iterações. Além dos mesmos coeficientes usados

na inversão, como o parâmetro de regularização e o números de parâmetros (vagarosidades).

O tamanho de enxame consiste na quantidade de indivíduos ou soluções que serão modificadas pelas heurísticas de cada algoritmo. Esta quantidade é um fator importante na busca do mínimo global no problema de otimização, pois um grande número de partículas infere uma maior amostra na busca deste mínimo global, sendo as soluções avaliadas a cada iteração.

Já o número iterações é a quantidade máxima de vezes que será realizado a busca do mínimo global, atualizando a posição das partículas com base em suas posições anteriores, e a eleição da solução ótima, caso a nova iteração gerar melhores posições, até uma melhor convergência.

Um modelo geológico sintético foi selecionado para realizar o experimento (ver Fig. 1), e este possui em 100 parâmetros. Para regularização do problema inverso foi estimado um multiplicador de Lagrange λ (parâmetro de regularização) com valor de 0.06 para o vínculo de suavidade. Estas informações são os coeficientes de inversão, e estão detalhadas juntamente com os parâmetros de controle na tabela I.

p_s	k	n	λ
1000	1000	100	0.06

Tabela I
COEFICIENTES DA INVERSÃO
E PARÂMETROS DE CONTROLE EM COMUM

1) *Parâmetros de controle GA*: O fator que está relacionado com a população é o mais significativo, pois proporcionará uma variedade genética (caracteres). A taxa de *crossover* (P_c) e mutação (P_m) e o sigma (σ) são importantes para gerar inovação por parte dos genes, existem muitas estratégias de cruzamento, como o elitismo, que agrega parte dos pais na nova geração. Os parâmetros de controle utilizados estão detalhados na tabela 2.

P_c	P_m	σ
1	0.1	0.1

Tabela II
PARÂMETROS DE CONTROLE GA

Vale salientar que a cada iteração há a avaliação das soluções, para serem separadas por exemplo por elitismo, como estratégia para se obter uma ótima solução e assim proporcionando gerações a cada cruzamento até que se chegue a convergência da solução ótima global. Os algoritmos de otimização utilizam um critério de parada, neste trabalho utilizamos apenas o número de iterações como critério de parada.

2) *Parâmetros de controle PSO*: Os parâmetros de controle que o compõe são: fator de inércia, coeficiente de aprendizagem individual, coeficiente transmissão cultural, tamanho e topologia do enxame.

Os fatores de coeficiente de aprendizagem individual e o coeficiente transmissão cultural, são conhecidos respectivamente como c_1 e c_2 , e selecionados geralmente nos intervalos entre [0...4]. Geralmente são iguais em seu uso, por exemplo $c_1 = 2.5$ e $c_2 = 2.5$. O fator de inércia geralmente utilizado é o $\omega = 0.9$ ou 0.72 e c_1 e c_2 sendo 2 [26]. Os parâmetros de controle utilizados no presente trabalho estão detalhados na tabela III.

c_1	c_2	ω
2	2	0.9

Tabela III
PARÂMETROS DE CONTROLE PSO

IV. RESULTADOS E DISCUSSÕES

Nesta seção apresentamos os resultados do experimento, que consistem nas figuras dos modelos estimados, curva de convergência da função objetivo, tabela dos tempos médios que algoritmos abordados levaram para encontrar uma solução ótima, e por fim a figura dos erros cumulativos relativos.

Os dados observados foram gerados a partir da modelagem de raios retos. Aplicamos sobre estes 5 % de ruído gaussiano pseudoaleatório, e o espaço de busca consistiu em um *range* de 50% a mais e 50% a menos dos parâmetros verdadeiros.

O modelo sintético escolhido (Fig. 1) possui uma estrutura não complexa, com certa consistência nas vagarosidades, contudo com uma anomalia central.

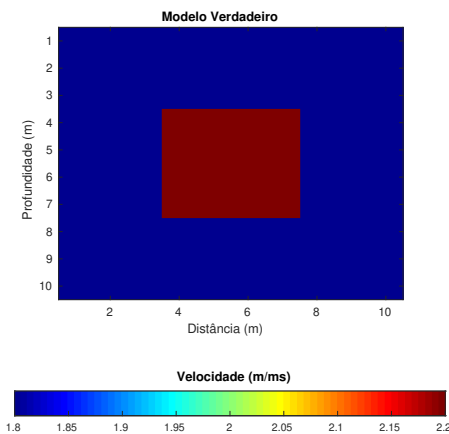


Fig. 1. Modelo sintético verdadeiro. Adaptado de [27]

A Fig. 2 mostra os resultados da média das 10 inversões para cada algoritmo global utilizado. Estes foram eficientes para estimar os parâmetros do modelo verdadeiros. É perceptível que o GA obteve visualmente uma estimativa em detrimento do PSO. E as versões paralelizadas foram melhores que as versões serializadas. Isto é ratificado a partir de uma análise estatística dos erros cumulativos relativos.

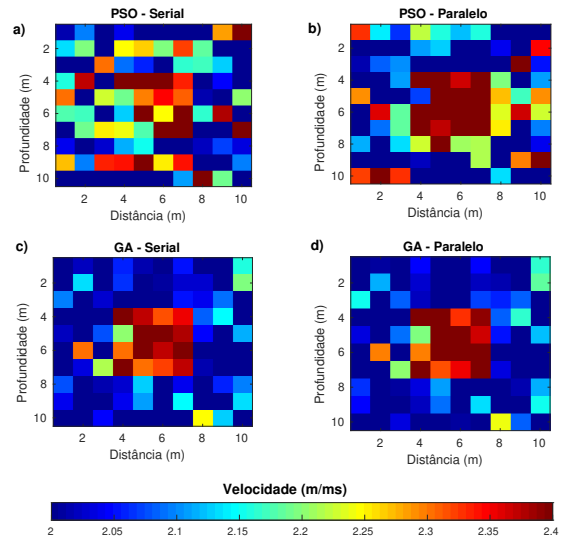


Fig. 2. Resultado das inversões para cada algoritmo global utilizado

Conseqüentemente as soluções obtiveram boa convergência para o critério de parada abordado. Isto é expressado na Fig. 3, que mostra a curva de convergência dos algoritmos para o problema de inversão sísmica.

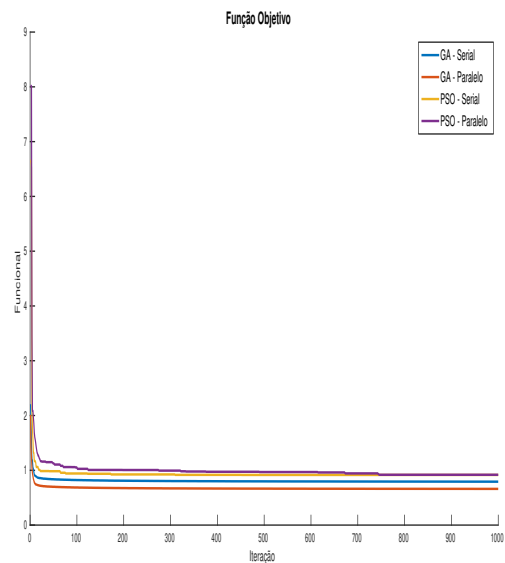


Fig. 3. Curva de convergência típica da função objetivo das inversões para cada algoritmo global utilizado

O experimento consistiu na obtenção do tempo de cada inversão para cada algoritmo de otimização, totalizando em um conjunto de 20 inversões. Desse modo foi realizado uma média dos tempos de execução e dos modelos estimados. A tabela IV mostra estes tempos médios de execução:

GA - Serial	GA - Paralelo	PSO - Serial	PSO - Paralelo
39.89 minutos	21.85 minutos	36.53 minutos	19.77 minutos

Tabela IV
MÉDIA DOS TEMPOS DE EXECUÇÃO

O algoritmo PSO foi mais rápido que o GA, isso se dá pelo fato que a versão abordada deste foi a canônica, e o GA possui uma maior quantidade de operadores que fornecem mais recursos para gerar as soluções. Ambos tiveram uma eficiência aproximada em relação a suas versões serializadas. A versão do PSO paralelizado obteve uma eficiência de aproximadamente 46% em relação a sua versão serial, enquanto o GA Paralelo obteve aproximadamente 45%.

Com base nos erros cumulativos relativos expressos na Fig. 4, o algoritmo GA serial obteve aproximadamente 13% de erro acumulado em aproximadamente 96% dos dados, enquanto sua versão paralela obteve em aproximadamente 98% dos dados. Já o PSO serial obteve aproximadamente 24% do erro acumulado em aproximadamente 97% dos dados, e sua versão paralela obteve um pouco acima com aproximadamente 98% dos dados.

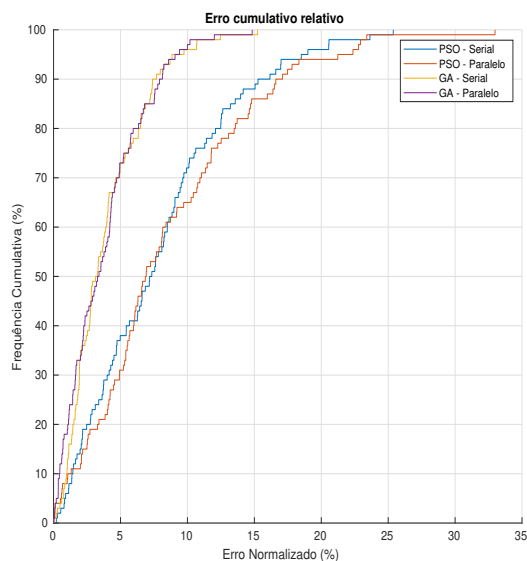


Fig. 4. Erro cumulativo relativo

Finalmente, também descobrimos que na maioria dos casos

o uso de paralelismo melhora a qualidade da soluções. Isso é algo interessante, pois constitui uma motivação adicional para paralelizar as algoritmos globais adotadas em tomografia sísmica.

V. CONCLUSÃO

O presente trabalho abordou discussões teóricas acerca do funcionamento dos dois algoritmos globais PSO e GA, de modo serial e paralelo, como também quais foram os aspectos de sua implementação. Também foi realizado um experimento de performance na resolução de um problema sintético de inversão sísmica poço-a-poço.

A linguagem de programação interpretada *Matlab* foi utilizada para produção dos algoritmos de inversão. Apesar de ser uma linguagem de alto nível, a eficiência para uma quantidade significativa de iterações mostrou-se eficaz, além da simplicidade de implementação da linguagem e sua legibilidade [28].

Os resultados obtidos nas inversões indicaram um desempenho melhorado das versões paralelizadas e boas características de convergência. Entretanto ambos os algoritmos de modo serial e paralelo se mostraram efetivos na busca pelo mínimo global.

Além disso, o uso do paralelismo também trouxe benefícios em termos de qualidade das soluções produzidas. No entanto, um estudo mais aprofundado do impacto do paralelismo em problemas inversos permanece como uma área de pesquisa aberta.

REFERÊNCIAS

- [1] G. F. Briao and J. P. Zubelli, *Tomografia sísmica por tempo de percurso: modelagem, métodos numéricos e implementação*. IMPA, 2005.
- [2] L. P. Geldart and R. E. Sheriff, *Problems in exploration seismology and their solutions*. Society of Exploration Geophysicists, 2004.
- [3] J. B. Ajo-Franklin, J. Urban, and J. M. Harris, "Temporal integration of seismic traveltimes tomography," in *SEG Technical Program Expanded Abstracts 2005*. Society of Exploration Geophysicists, 2005, pp. 2468–2471.
- [4] M. Sen and P. Stoffa, "Global optimization method in geophysical inversion: Elsevier sci," *Publ. Co., The Netherlands*, 1995.
- [5] M. Sambridge and K. Mosegaard, "Monte carlo methods in geophysical inverse problems," *Reviews of Geophysics*, vol. 40, no. 3, pp. 3–1, 2002.
- [6] S. K. Pullammanappallil and J. N. Louie, "Inversion of seismic reflection traveltimes using a nonlinear optimization scheme," *Geophysics*, vol. 58, no. 11, pp. 1607–1620, 1993.

- [7] Z. Weber, "Seismic traveltome tomography: a simulated annealing approach," *Physics of the Earth and Planetary Interiors*, vol. 119, no. 1-2, pp. 149–159, 2000.
- [8] J. L. Fernandez Martinez, T. Mukerji, E. Garcia Gonzalo, and A. Suman, "Reservoir characterization and inversion uncertainty via a family of particle swarm optimizers," *Geophysics*, vol. 77, no. 1, pp. M1–M16, 2012.
- [9] J. Tronicke, H. Paasche, and U. Böniger, "Crosshole traveltome tomography using particle swarm optimization: A near-surface field example," *Geophysics*, vol. 77, no. 1, pp. R19–R32, 2012.
- [10] M. Ebrahimi, M. Riahi, and M. S. Abadeh, "Nonlinear crosshole traveltome tomography using hybridizing particle swarm optimization with linearized least squares algorithm." *Bollettino di Geofisica Teorica ed Applicata*, vol. 59, no. 3, 2018.
- [11] P. B. Parker, "Genetic algorithms and their use in geophysical problems," Lawrence Berkeley National Lab.(LBNL), Berkeley, CA (United States), Tech. Rep., 1999.
- [12] E. Alba, J. M. Troya *et al.*, "A survey of parallel distributed genetic algorithms," *Complexity*, vol. 4, no. 4, pp. 31–52, 1999.
- [13] G. Sharma and J. Martin, "Matlab®: a language for parallel computing," *International Journal of Parallel Programming*, vol. 37, no. 1, pp. 3–36, 2009.
- [14] S. Kleiman, D. Shah, and B. Smaalders, *Programming with threads*. Sun Soft Press Mountain View, 1996.
- [15] P. Luszczek, "Parallel programming in matlab," *The International Journal of High Performance Computing Applications*, vol. 23, no. 3, pp. 277–283, 2009.
- [16] A. Tikhonov and V. Y. Arsenin, "Solutions of ill-posed problems. winston," *Washington, DC*, pp. 95–108, 1977.
- [17] M. G. O. Toolbox, "Matlab global optimization toolbox," *The MathWorks, Natick, MA, USA*, 2016.
- [18] J. H. Holland, "Adaptation in natural and artificial systems, university of michigan press," *Ann arbor, MI*, vol. 1, no. 97, p. 5, 1975.
- [19] D. E. Goldberg, J. Richardson *et al.*, "Genetic algorithms with sharing for multimodal function optimization," in *Genetic algorithms and their applications: Proceedings of the Second International Conference on Genetic Algorithms*. Hillsdale, NJ: Lawrence Erlbaum, 1987, pp. 41–49.
- [20] R. L. Haupt and S. E. Haupt, *Practical genetic algorithms with CD-Rom*. Wiley-Interscience, 2004.
- [21] K. Man, K. Tang, and S. Kwong, "Genetic algorithms: Concepts and designs, springer," 1999.
- [22] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95-International Conference on Neural Networks*, vol. 4. IEEE, 1995, pp. 1942–1948.
- [23] F. A. C. Viana and V. S. Júnior, "Particle swarm optimization," *Published in: 14° POSMEC-Simpósio do Programa de Pós-Graduação em Engenharia Mecânica. Universidade Federal de Uberlândia. Uberlândia-MG*, 2004.
- [24] W. Gao, Q. Kemao, H. Wang, F. Lin, and H. S. Seah, "Parallel computing for fringe pattern processing: A multicore cpu approach in matlab® environment," *Optics and lasers in engineering*, vol. 47, no. 11, pp. 1286–1292, 2009.
- [25] M. Szymczyk and P. Szymczyk, "Matlab and parallel computing," *Image Processing & Communications*, vol. 17, no. 4, p. 207, 2012.
- [26] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability, and convergence in a multidimensional complex space," *IEEE transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58–73, 2002.
- [27] M. A. Meju, *Geophysical data analysis: understanding inverse problem theory and practice*. Society of Exploration Geophysicists, 1994.
- [28] I. MathWorks, *MATLAB: the language of technical computing. Desktop tools and development environment, version R2018b*. MathWorks, 2018, vol. 9.