ALGORITMO ALO MODIFICADO BASEADO EM PROJEÇÃO VETORIAL

RAIMUNDO N. D. COSTA FILHO

Coordenação do Curso de Engenharia Elétrica, Universidade Federal do Maranhão- Campus Balsas
MA 140, km 4, Balsas- MA, 65800-800, Brasil
E-mail: raimundo.diniz@ufma.br

Abstract – Currently, metaheuristics are algorithms applied to solve various optimization problems. In this context, the Ant Lion Optimizer (ALO) algorithm is proposed in 2015 based on the ant lion hunting mechanism, where its main prey is the ants. The ALO has basically four steps: trapping the ants, random walk of the ants, elitism and catching preys. However, ALO presents problems in diversifying its population, which ends up decreasing its performance in certain problems. Thus, this work presents an improved ALO algorithm based on vector projection and called IALO (Improvement ALO). For comparison between the proposed algorithm and ALO, simulations are performed on 19 benchmark functions. The results obtained indicate that IALO is more efficient and robust than ALO for most benchmark functions.

Resumo – Atualmente, as metaheurísticas são algoritmos aplicados na solução de vários problemas de otimização. Neste contexto, o algoritmo *Ant Lion Optimizer* (ALO) é proposto em 2015 tendo como princípio o mecanismo de caça da formiga-leão, onde sua principal presa são as formigas. O ALO possui basicamente quatro etapas: aprisionamento das formigas em armadilhas, passeio aleatório das formigas, elitismo e captura das presas. Porém, o ALO apresenta problemas em diversificar sua população, o que acaba diminuindo o seu desempenho em certos problemas. Desta maneira, este trabalho apresenta um algoritmo ALO melhorado baseado em projeção vetorial e denominado de IALO (*Improvement* ALO). Para comparação entre o algoritmo proposto e o ALO, simulações são realizadas em 19 funções de *benchmark*. Os resultados obtidos indicam que o IALO é mais eficiente e robusto do que o ALO para a maioria das funções de *benchmark*.

Keywords – ALO, IALO, Metaheuristic, Optimization.

Palavras- chaves – ALO, IALO, Metaheurística, Otimização.

1. Introdução

Nas últimas décadas, foi progressivamente se firmando a ideia de desenvolver algoritmos inspirados em mecanismos de adaptação dos seres vivos, conforme observados na natureza. A justificativa geral para tais ideias é que tais mecanismos de adaptação, na natureza, podem produzir respostas adequadas para problemas de grande complexidade (Gaspar-Cunha *et al.*, 2013). Neste contexto, podemos citar os algoritmos PSO (*Particle Swarm Optmization*) (Kennedy and Eberhart, 1995), ACO (*Ant Colony Optimization*) (Dorigo *et al.*, 1996), FPA (*Flower Pollination Algorithm*) (Yang, 2012), dentre outros. Na literatura especilizadas, algoritmos desta natureza estão dentro de um grupo de técnicas denominadas de metaheurísticas.

Neste contexto, em 2015, o pesquisador Seyedali Mirjalili descreveu e aplicou o algoritmo ALO (*Ant Lion Optimize*r) em 19 funções matemáticas e clássicos problemas de otimização (Mirjalili, 2015). O algoritmo ALO é inspirado na estratégia de forrageamento utilizado pelas formigas-leão para capturar suas pressas. As formigas-leão pertencem à família *Myrmeleontidae* (Neuroptera) e são predadores de artrópodes conhecidas por seu método de captura de presas no qual a larva esconde-se no fundo de um pequeno funil cônico construído na área ou poeira alimentando-se de formigas e outros insetos que ali caem (Maragno *et al.*, 2007).

Desde sua publicação até a presente data, várias aplicações do algoritmo ALO são reportadas na literatura. Aplicações quem englobam inúmeras áreas indo desde projeto de controladores em sistemas de energia (Costa Filho and Paucar, 2018) até otimização do desempenho de antenas em sistemas de comunicações (Subhashini and Satapathy, 2017).

Ademais, várias variantes do referido algoritmo foram apresentadas na literatura, como por exemplo, com mapas caóticos ao quais os autores denominaram de CALO (*Chaotic* ALO) (Zawbaa *et al.*, 2016), usando o passeio aleatório (*random walk*) com voos Lèvy que foi denominado de LALO (Lèvy ALO) (Emary and Zawbaa, 2019), empregando um passeio aleatório dinâmico e a ideia de aprendizado oposto nomeado pelos autores de DALO (*Dynamic Opposite Learning* ALO) (Dong *et al.*, 2021) e (Guo *et al.*, 2020) aplicou oito tipos de modelos de espirais complexas em uma das etapas do ALO. As formas híbridas e suas aplicações podem ser encontradas em (Assiri *et al.*, 2020).

No algoritmo ALO, os passeios aleatórios das formigas garantem a convergência do processo de otimização e o método da roleta melhora a capacidade de pesquisa global até certo ponto, porém na forma padrão o supracitado algoritmo precisa ser melhorado em alguns aspectos: (1) com o aumento das iterações, a diversidade da população diminui gradualmente e (2) se a melhor as formigas (elite) e a roleta não estão na região global ideal, então toda a população pode cair em um ótimo local e a velocidade de convergência será afetada (Guo *et al.*, 2020).

Desta maneira, fundamentado nos itens (1) e (2) do parágrafo anterior e motivado pelo Teorema do *No Free Lunch* (Wolpert and Macready, 1997), o autor deste artigo foi encorajado a implementar uma nova variante do algoritmo ALO empregando a ideia de projeção vetorial. O algoritmo melhorado será

denominado de IALO (*Improvement* ALO) no decorrer do texto. O IALO é aplicado em 19 funções de *benchmark* e comparado com o algoritmo ALO na sua forma padrão

O trabalho está organizado da seguinte maneira: a Seção 2 apresenta a descrição da metaheurística ALO, a Seção 3 descreve o algoritmo proposto, a Seção 4 apresenta os resultados obtidos nas simulações e na Seção 5 são expostas as conclusões desta pesquisa.

2. ALO

A metaheurística ALO é inspirado no comportamento de caça das formigas-leão e no aprisionamento da sua principal presa, as formigas, em armadilhas em formato cônico. Há quatro operações principais no referido algoritmo: (1) aprisionamento das formigas em armadilhas, (2) passeio aleatório das formigas, (3) elitismo e (4) captura das presas (Mirjalili, 2015), (Chen *et al.*, 2017).

2.1 Aprisionamento das formigas em armadilhas

Para o modelo matemático do comportamento do aprisionamento das formigas em armadilhas, o ALO utiliza (1)-(2) (Mirjalili, 2015).

$$c^t = \frac{c^t}{I} \; e \; d^t = \frac{d^t}{I} \tag{1}$$

$$I = 10^w \frac{t}{T} \tag{2}$$

Onde t é a iteração corrente, T é o máximo número de iterações, w é uma constante que depende da iteração corrente ($w = 2$, quando $t > 0{,}1T$; $w = 3$, quando $t > 05T$; $w = 4$, quando $t > 0{,}75T$; $w = 5$, quando $t > 0{,}9T$ e $w = 6$, quando $t > 0{,}95T$).

2.2 Passeio Aleatório (Random Walk) das formigas

Como as formigas se movem de maneira estocástica na natureza para procurar comida, o modelo matemático para este passeio aleatório é por (3) (Mirjalili, 2015).

$$X(t) = [0, cs(2r(t_1) - 1), \cdots, cs(2r(t_N) - 1)] \tag{3}$$

Onde cs é a soma cumulativa, N é o número máximo de iterações e $r(t)$ é definido por (4).

$$r(t) = \begin{cases} 1, se\ rand > 0{,}5 \\ 0, se\ rand \leq 0{,}5 \end{cases} \tag{4}$$

Onde $rand$ é um número aleatório no intervalo [0,1].

Para manter o passeio aleatório dentro do espaço de busca, os mesmo são normalizados utilizando (5).

$$X_i^{t+1} = \frac{(X_i^t - a_i)(d_i^t - c_i^t)}{b_i - a_i} + c_i^t \tag{5}$$

Onde a_i e b_i são o passeio aleatório mínimo e máximo da i^{th} variável, respectivamente; c_i^t e d_i^t são o mínimo e máximo da variável i na t^{th} iteração, ao qual são definidos por (6).

$$\begin{cases} c_i^t = Antlion_j^t + c^t \\ d_i^t = Antlion_j^t + d^t \end{cases} \tag{6}$$

Onde c^t representa o menor valor de todas as variáveis na t^{th} iteração, d^t representa o maior valor de todas as variáveis na t^{th} iteração e $Antlion_j^t$ representa a posição da j^{th} formiga-leão selecionada na t^{th} iteração.

2.3 Elitismo

No decorrer das iterações, a melhor formiga-leão (alguns texto denominam de elite, ou seja, a formiga-leão elite é aquela que apresenta o melhor *fitness*) deve ser preservada para garantia da solução (pode ser o ótimo local ou global). A formiga-leão elite pode afetar o movimento de todas as formigas em cada iteração, assim como uma formiga-leão selecionada através de uma roleta. Desta maneira, a posição das formigas a cada iteração é atualizado por (7) (Mirjalili, 2015).

$$Ant_i^t = \frac{R_A^t + R_E^t}{2} \tag{7}$$

Onde R_A^t representa o passeio aleatório em torno da formiga-leão selecionado pela roleta na t iteração e R_E^t representa o passeio aleatório em torna da elite na t iteração.

2.4 Captura das Presas

O procedimento de captura das presas ocorre quando uma formiga tem o *fitness* maior do que a correspondente formiga-leão (Mirjalili, 2015):

$$Antlion_j^t = Ant_i^t \; se \; f(Ant_i^t) < f\left(Antlion_j^t\right) \tag{8}$$

Onde Ant_i^t indica a posição da i^{th} formiga na t^{th} iteração e $f(\blacksquare)$ denota a função que calcula o *fitness*. O procedimento simplificado do ALO é apresentado em Algoritmo (1).

3. IALO

A metaheurística ALO, na sua forma padrão, a posição das formigas em cada iteração é dada pela média aritmética de R_A e R_E. Porém, pode haver um problema de diversidade na população e consequentemente problemas de estagnação do algoritmo. Sendo assim, o IALO proposto pode evitar a estagnação aplicando a ideia de projeção vetorial de R_A em R_E ou o contrário.
A projeção vetorial de **a** sobre **b** é dada por (9).

$$proj_b a = \frac{(a.b)}{|b|^2} \times b \tag{9}$$

Baseado em (9), pode-se alterar a atualização da posição das formigas no algoritmo ALO através de (10). Dependendo do valor de u, R_A segue R_E ou R_E segue R_A, essa característica tende a melhorar a diversidade do algoritmo ALO.

$$\begin{cases} proj_{R_E} R_A \; se \; u > 0.5 \\ proj_{R_A} R_E \; se \; u \leq 0{,}5 \end{cases} \tag{10}$$

Onde u é um número aleatório no intervalo de [0, 1].

Para implementação do IALO, pode-se alterar a linha que atualiza a posição das formigas do algoritmo (1) substituindo a equação (7) pela (10). As outras etapas são similares ao ALO.

Algoritmo 1: Pseudocódigo do algoritmo ALO.

Inicializar a população de formigas (Ant_i) e formigas-leão ($Antlion_i$) de maneira aleatória.
Calcular $f(Ant_i)$ e $f(Antlion_i)$ com $i = 1, 2, \cdots, N$.
Encontrar a melhor formiga-leão e atribuí-la como a elite ($Antlion^*$).
Enquanto ($t \leq T$)
  Para cada formiga
    Selecionar uma formiga-leão usando a roleta.
    Atualizar c e d usando (1) e (2).
    Criar um passeio aleatório (3)-(6)
    Atualizar a posição das formigas usando (7).
  Fim Para
Calcular o *fitness* de todas as formigas.
Atualizar a posição das formigas-leão (8).
Atualizar a formiga-leão elite.
$t = t + 1$
Fim Enquanto
Retornar a formiga-leão elite ($Antlion^*$).

4. Simulações e resultados

Com o objetivo de avaliar o desempenho do algoritmo proposto, foram selecionadas 19 funções de *benchmark* para explorar a capacidade do algoritmo. Este conjunto de funções foi escolhido também em função da disponibilidade de resultados encontrados na literatura. As funções são do tipo unimodais (Tabela 1), multimodais (Tabela 2) e multimodais de dimensão fixa (Tabela 3). As funções da Tabela 1 e Tabela 2 são empregas em (Mirjalili, 2015). As Tabelas 1-3 apresentam as características gerais das referidas funções de *benchmark* utilizadas, todas elas associadas a problemas de minimização. A coluna *Dim* apresenta a dimensão do espaço de busca da função utilizada nas simulações. Os resultados obtidos pelo IALO são comparados com sua versão padrão (Mirjalili, 2015). Como índices de comparação foram utilizados o desvio padrão, o melhor, pior e a média dos valores encontrados pelos algoritmos. Seguindo a referência (Mirjalili, 2015), todos os algoritmos foram executados 30 vezes. O máximo número de iterações utilizadas para todas as 19 funções e ambos os algoritmos foi de 1000. Ademais, o tamanho da população de 30 foi utilizado nas simulações.

Tabela 1. Funções unimodais.

Função	*Dim*	Limites	F_{min}
$F_1(x) = \sum_{i=1}^{n} x_i^2$	10	[-100,100]	0
$F_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	[-10,10]	0
$F_3(x) = \sum_{i=1}^{n} \left(\sum_{j-1}^{i} x_j \right)^2$	30	[-100,100]	0
$F_4(x) = \max_i \{ \lvert x_i \rvert, 1 \leq i \leq n \}$	30	[-100,100]	0
$F_5(x) = \sum_{i=1}^{n} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30	[-30,30]	0
$F_6(x) = \sum_{i=1}^{n} ([x_i + 0.5])^2$	30	[-100,100]	0
$F_7(x) = \sum_{i=1}^{n} i x_i^4 + random[0,1]$	30	[-1.28, 1.28]	0

As Tabelas 4-6 apresentam os resultados obtidos em termos de média, melhor resultado, pior resultado e desvio padrão nas 30 execuções independentes dos algoritmos ALO e IALO. Os melhores valores destes índices estão destacados. Para as funções unimodais, o algoritmo IALO apresentou os melhores resultados em todos os índices de comparação. No caso das funções multimodais, geralmente F_8 é uma função benchmark difícil de obter a solução ótima, no caso a solução ótima é -12.569,487. Observe que o IALO chegou bem próximo a este valor no seu melhor resultado, enquanto que o valor obtido do ALO foi de -9.9835×10^3. Em contrapartida, para F_8, o desvio padrão do IALO ($3{,}0539 \times 10^3$) foi elevado comparado com ALO ($1{,}2794 \times 10^3$). Em F_{10} o algoritmo IALO obteve um desvio padrão nulo, mostrando uma maior diversidade na população equiparando-se ao ALO (0,7577).
Com relação às funções multimodais de dimensão fixa, os dois algoritmos tiveram praticamente o mesmo desempenho no quesito melhor resultado. Em geral, para esta classe de função benchmark, o desvio padrão e a média do IALO foram melhores do que os resultados do ALO. Pode-se observar que alguns resultados são iguais tanto para o ALO como para o IALO, significando que alcançaram o ótimo global da função objetivo analisada.

A Figura 1((a)-(f)) apresenta as curvas de convergência dos melhores resultados obtidos pelos algoritmos ALO e IALO para algumas funções. Observe a estagnação do algoritmo ALO na maioria das curvas de convergência. Mesmo com uma inicialização distante do ponto ótimo, o algoritmo IALO consegue chegar ao referido ponto.

Tabela 2. Funções multimodais.

Função	*Dim*	Limites	F_{min}
$F_8(x) = \sum_{i=1}^{n} -x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)$	30	[-500, 500]	-418,9829 x *dim*
$F_9(x) = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$	30	[-5.12, 5.12]	0
$F_{10}(x) = -20exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$	30	[-32, 32]	0
$F_{11}(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	[-600, 600]	0
$F_{12}(x) = \frac{\pi}{n}\left\{10\sin(\pi y_i) + \sum_{i=1}^{n-1}(y_i - 1)^2[1 + 10\sin^2(\pi y_{i+1})] + (y_n - 1)^2\right\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4)$ $y_i = 1 + \frac{x_i + 1}{4}$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	30	[-50, 50]	0
$F_{13}(x) = 0.1\left\{\sin^2(3\pi x_1) + \sum_{i=1}^{n}(x_i - 1)^2[1 + \sin^2(3\pi x_i + 1)] + (x_n - 1)^2[1 + \sin^2(2\pi x_n)]\right\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	30	[-50, 50]	0

Tabela 3. Funções multimodais de dimensão fixa.

Função	*Dim*	Limites	F_{min}
$F_{14}(x) = \left(\frac{1}{500} + \sum_{j=1}^{25}\frac{1}{j + \sum_{i=1}^{2}(x_i - a_{ij})^6}\right)^{-1}$	2	[-65,65]	1
$F_{15}(x) = \sum_{i=1}^{11}\left[a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}\right]^2$	4	[-5,5]	0,00030
$F_{16}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	[-5,5]	-1,0316
$F_{17}(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10$	2	[-5,5]	0,398
$F_{18}(x) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \times (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)$	2	[-2,2]	3
$F_{19}(x) = -\sum_{i=1}^{4} c_i exp\left(-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2\right)$	3	[1,3]	-3,86

Tabela 4. Resultados para as funções unimodais.

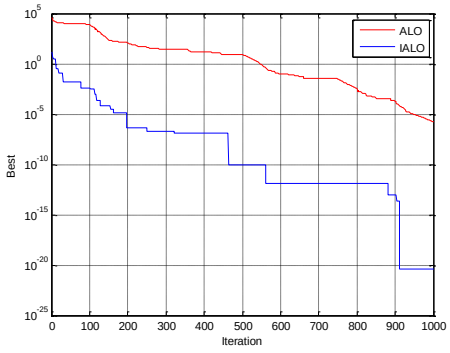
Função	**Algoritmo**	**Melhor**	**Pior**	**Média**	**Desvio**
F_1	ALO	1,7170E-6	6,2158E-5	1,1128E-5	1,1218E-5
	IALO	4,3777E-21	2,1623E-15	3,4552E-16	5,8082E-16
F_2	ALO	0,2291	124,6193	39,7892	48,2605
	IALO	3,1611E-9	1,0556E-7	2,7248E-8	2,3899E-8
F_3	ALO	218,4941	2,6964E3	1,172E3	653,2769
	IALO	1,5148E-18	1,3873E-13	1,9236E-14	3,2684E-14
F_4	ALO	3,9196	18,6473	11,8326	3,6584
	IALO	1,9881E-10	1,4610E-8	4,5252E-9	3,6137E-9
F_5	ALO	21,7646	1,0089E3	138,8906	237,3272
	IALO	2,2539E-15	1,1317E-5	1,2789E-6	2,8176E-6
F_6	ALO	5,4336E-7	3,4125E-5	8,7040E-6	7,0587E-6
	IALO	7,4019E-19	5,9179E-8	4,8903E-9	1,2355E-8
F_7	ALO	0,0528	0,2029	0,1092	0,0386
	IALO	6,4820E-7	2,2081E-4	5,8893E-5	5,1610E-5

Tabela 5. Resultados para as funções multimodais.

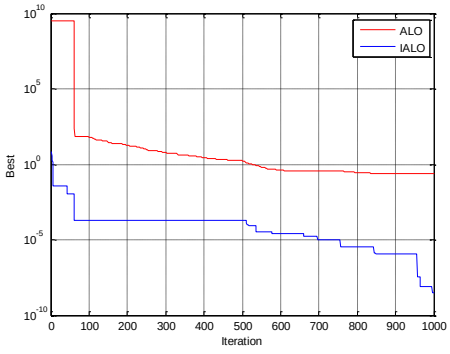
Função	Algoritmo	Melhor	Pior	Média	Desvio
F_8	ALO	-9,9835E3	-5,4177E3	-5,9811E3	1,2794E3
	IALO	-12,569E3	-5,4177E3	-10,064E3	3,0539E3
F_9	ALO	33,8286	139,2937	82,1171	22,4923
	IALO	0	0	0	0
F_{10}	ALO	4,0712E-4	4,3828	2,0168	0,7577
	IALO	8,8818E-16	8,8818E-16	8,8818E-16	0
F_{11}	ALO	4,1830E-4	0,0702	0,0117	0,0143
	IALO	0	9,6589E-15	5,9582E-16	1,7658E-15
F_{12}	ALO	6,0241	33,9944	10,8679	5,7290
	IALO	2,2489E-20	7,3199E-9	4,1639E-10	1,3175E-9
F_{13}	ALO	3,2233E-6	2,8516	0,1756	0,5799
	IALO	5,7302E-18	5,5391E-9	4,9327E-10	1,2495E-9

Tabela 6. Resultados para as funções multimodais de dimensão fixa.

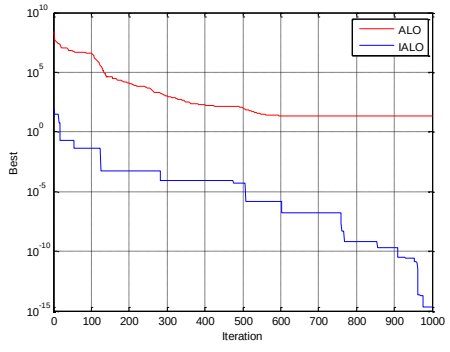
Função	Algoritmo	Melhor	Pior	Média	Desvio
F_{14}	ALO	0,9980	3,9683	1,7258	0,8840
	IALO	0,9980	1,9920	1,1305	0,3379
F_{15}	ALO	3,0749E-4	0,0204	0,0020	0,0049
	IALO	3,0749E-4	3,0810E-4	3,0761E-4	1,5173E-7
F_{16}	ALO	-1,0316	-1,0316	-1,0316	0
	IALO	-1,0316	-1,0316	-1,0316	2,2073E-8
F_{17}	ALO	0,3979	0,3979	0,3979	0
	IALO	0,3979	0,3979	0,3979	0
F_{18}	ALO	3,0000	3,0000	3,0000	0
	IALO	3,0000	3,0000	3,0000	0
F_{19}	ALO	-3,8628	-3,8628	-3,8628	8,4294E-8
	IALO	-3,8628	-3,8628	-3,8628	0



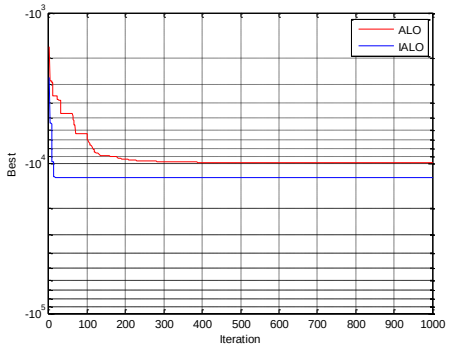
(a)



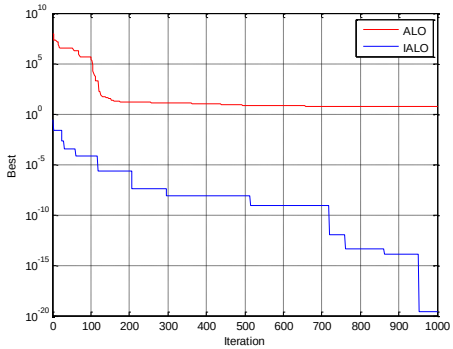
(b)



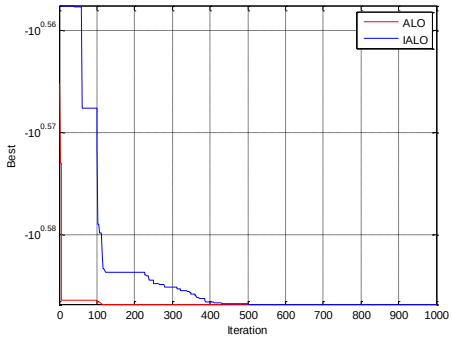
(c)



(d)



(e)



(f)

Figura 1. Melhor resultado da convergência de algumas funções: (a) F_1, (b) F_2, (c) F_5, (d) F_8, (e) F_{12} e (f) F_{19}.

5. Conclusão

Neste trabalho apresentou um algoritmo ALO melhorado denominado de IALO. O algoritmo proposto é baseado na aplicação da projeção vetorial na etapa de atualização das posições das formigas, a principal presa da formiga-leão. O algoritmo IALO foi aplicado em 19 funções de *benchmark* do tipo unimodais, multimodais e multimodais de dimensão fixa. Os resultados obtidos pelo IALO foram comparados com o ALO na sua forma padrão. Com índices de comparação foram utilizados a média, desvio padrão, melhor resultado e pior resultado. Os referidos resultados e as curvas de convergências das funções objetivas indicam que o IALO é mais eficiente e robusto do que o ALO para a maioria das funções de *benchmark*. Como futuras propostas é expandir o leque de aplicações, como por exemplo, despacho econômico (minimizar o custo de energia elétrica) e sintonia de controladores (maximizar o amortecimento, por exemplo).

Referências Bibliográficas

Assiri, A. S.; Hussien, A. G. and Amin, M. (2020). Ant lion optimization: variants, hybrids, and applications. *IEEE Acess*, Vol. 8, pp. 77746-77764.

Costa Filho, R. N. D. and Paucar, V. L. (2018). Robust and coordinated tuning of PSSs and FACTS-PODs of interconnected systems considering signal transmission delay using ant lion optimizer. *Journal of Control, Automation and Electrical Systems*, Vol. 29, pp. 625-639.

Chen, Z.; Yuan, X.; Yuan, Y.; Ho-Ching, H. and Fernando, T. (2017). Parameter identification of integrated model of hydraulic turbine regulating system with uncertainties using three different approaches. *IEEE Trans on Power Systems*, Vol. 32, No. 5, pp. 3482-3491.

Dong, H.;Xu, Y.; Li, X.; Yang, Z. and Zou, C. (2021). Na improved antlion optimizer with dynamic random walk and dynamic opposite learning. *Knowlede-Based Systems*, Vol. 216 pp. 106752.

Dorigo, M.; Maniezzo, V. and Colorni, A. (1996). The ant system: optimization by a colony of cooperating ants. *IEEE Trans Syst Man Cybern*, Vol. 26, pp. 29-42.

Emary, E. and Zawbaa, H. M. (2019). Feature selection via Lèvy antlion optimization. *Pattern Analysis and Applications*, Vol. 22, No. 3, pp. 857-876.

Gaspar-Cunha, A.; Takahashi, R. e Antunes, C. H. (2013). Manual de Computação Evolutiva e Metaheurística. Editora: UFMG.

Guo, M. W.; Wang, J. S.; Zhu, L. F.; Guo, S. S. and Xie, W. (2020). Improved ant lion optimizer based on spiral complex path searching patterns. *IEEE Access*, Vol. 8, pp. 22094-22126.

Kennedy, J. and Eberhart, R. (1995). Particle swarm optimization. *Proceedings IEEE International Conference on Neural Networks*, pp. 1942-1948.

Maragno, F. P.; Alves, L. A.; Barros, C. S. e Wolff, L. L. (2007). Forrageamento de larvas de formigas-leão (Neuroptera: Myrmeleontidae) no pantal do Miranda. *Anais do VIII Congresso de Ecologia do Brasil*, pp. 1-2.

Mirjalili, S. (2015). The ant lion optimizer. *Advances in Engineering Software*, Vol. 83, pp. 80-98.

Subhashini, K. R. and Satapathy, J. K. (2017). Development of an enhanced ant lion optimization algorithm and its application in antenna array synthesis. *Applied Soft Computing Computation*, Vol. 59, pp. 153-173.

Wolpert, D. H. and Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Trans on Evolutionar Computation*, Vol. 1, No. 1, pp. 67- 82.

Yang, X. S. (2012). Flower pollination algorithm for global optimization. *International Conference on Unconventional Computing and Natural Computation*, pp. 240-249.

Zawbaa, H. M.; Eary, E. and Grosan, C. (2016). Feature selection via chaotic antlion optimization. *PloS One*, Vol. 11, No. 3, pp. 1- 21.