Científica, vol. 25, núm. 1, pp. 01-11, enero-junio 2021, ISSN 2594-2921, Instituto Politécnico Nacional MÉXICO
DOI: https://doi.org/10.46842/ipn.cien.v25n1a09

Análisis comparativo de la modificación del parámetro de inercia para la mejora en el desempeño del algoritmo PSO

Comparative Analysis of the Modification of the Inertia Parameter for the Improvement of the PSO Algorithm Performance

Valeria **Álvarez-Garduño**[1], Natalia **Guadiana-Ramírez**[2], Álvaro **Anzueto-Ríos**[3]

[1]Instituto Politécnico Nacional, MÉXICO
https://orcid.org/0000-0003-2173-8947 | valvagar18@gmail.com
[2]Instituto Politécnico Nacional, MÉXICO
https://orcid.org/0000-0002-4123-0895 | nagura.bio@hotmail.com
[3]Instituto Politécnico Nacional, MÉXICO
https://orcid.org/0000-0003-1627-0323 | aanzuetor@ipn.mx

Recibido 30-07-2020, aceptado 26-08-2020.

Resumen

En este trabajo se presenta un desarrollo para mejorar el desempeño del algoritmo de optimización metaheurístico nombrado *Particle Swarm Optimization* (PSO). El algoritmo PSO está inspirado en el comportamiento que demuestran los grupos de individuos en la naturaleza, como ejemplo podemos mencionar las parvadas y los cardúmenes. Cada individuo o partícula, de forma análoga en un proceso matemático; es considerado como una posible solución y en ellos se contempla, como información relevante, su posición y la velocidad. La velocidad de cada partícula es modificada al multiplicarse por un parámetro nombrado factor de inercia y es este parámetro que proponemos modificar para mejorar el desempeño del algoritmo. La modificación del factor de inercia se desarrolla de dos maneras, decremento lineal y decremento caótico. Se han considerado las funciones de referencia Eggholder y Six-Hump Camelback, para determinar la mejora en el desempeño del algoritmo PSO. Los resultados presentados en este trabajo indican un mejor desempeño al aplicar el decremento de tipo caótico al factor de inercia.

Palabras clave: PSO, inercia, optimización, inercia caótica.

Abstract

In this work an improvement of the metaheuristic algorithm called Particle Swarm Optimization (PSO) is shown. The PSO algorithm is inspired in the behavior that groups of individuals from nature exhibit, it can be mentioned for example flocks and shoals. Each individual or particle, on a mathematical process in an analogue manner, is considered as a possible solution and from them it is contemplated, as relevant information, its position, and velocity. The velocity of each particle is modified as it is multiplied by a parameter named inertia weight and it is this parameter that we propose to modify for the improvement of the performance of the algorithm. The variation of the inertia weight develops as the following two manners, linear decreasing, and chaotic decreasing. The functions Eggholder and Six-Hump Calmelback were considered to determine the improvement of the performance in the PSO algorithm. The reported results in this work indicate a better performance in the application of chaotic decreasing to the inertia weight.

Index terms: PSO, inertia weight, optimization, chaotic inertia.

I. INTRODUCCIÓN

El método de optimización por enjambre de partículas (PSO, por sus siglas en inglés) es un algoritmo estocástico originalmente presentado por Kennedy y Eberhart en 1995 inspirándose en el comportamiento que manifiestan las agrupaciones de entes biológicos como es el caso de una parvada y un banco de peces en busca de comida [1]. Para este algoritmo, un pez o un ave representa una 'partícula', cada partícula está definida por 3 elementos: posición, velocidad y ajuste (fitness). Cada una de ellas viajará a través del espacio de búsqueda o solución, considerando su propia experiencia y ajustando su conocimiento de acuerdo con el mejor resultado de entre los más exitosos del resto de los miembros del enjambre; por lo que cada partícula representa una solución potencial [2].

En PSO, un enjambre está conformado de *N* partículas o individuos y, matemáticamente, puede ser representado por el siguiente vector:

$$X = \{x_1, x_2, \ldots, x_N\} \tag{1}$$

En donde la posición de la *i*-ésima partícula en la iteración (o generación) *t* es:

$$x_i(t) = [x_{i1}, x_{i2}, \ldots, x_{ij}] \tag{2}$$

siendo *j* un parámetro que representa una característica o variable en el espacio de solución. La posición es modificada utilizando el vector velocidad:

$$v_i(t) = [v_{i1}, v_{i2}, \ldots, v_{ij}] \tag{3}$$

para la siguiente ecuación de movimiento:

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1) \tag{4}$$

Donde *t* y *t* + 1 indican dos generaciones sucesivas del algoritmo. El vector de velocidad (Ec. 3) determina la forma en que las partículas se desplazan a través del espacio de búsqueda o solución. La exploración o desplazamiento de las partículas dentro del espacio de búsqueda está determinado por las siguientes 3 componentes:

1. Parámetro de inercia o momento, evita un cambio drástico de dirección al realizar un seguimiento del flujo anterior;
2. Componente cognitivo, explica la tendencia de las partículas a regresar a la mejor posición previamente registrada, si la nueva posición no representa un mejor valor; y
3. Componente social, identifica la predisposición de una partícula a moverse hacia la mejor posición o solución determinada por todo el enjambre.

Conforme a lo anterior el vector de velocidad de la *i*-ésima partícula en una generación *t* + 1 se define como:

$$v_{ij}(t+1) = w(t+1)v_{ij}(t) + r_1c_1(p_{ij} - x_{ij}(t)) + r_2c_2(g_{ij} - x_{ij}(t)) \tag{5}$$

Para *j* = 1, 2, …, *dim*.

donde r_1, r_2 son números aleatoriamente generados dentro del rango $[0,1)$; c_1 y c_2 denotan los coeficientes de aceleración cognitivo y social, respectivamente, y el rango típico para ellos es $[0,4]$; $w(t+1)$ es la inercia en una generación $t+1$ con un rango de $[0,1]$, que para nuestro caso inicia con un valor igual a la unidad y decrece conforma avanza el proceso de optimización; p_{ij} es el mejor personal encontrado por la *i*-ésima partícula; y finalmente g_{ij} es el mejor global hasta el momento.

Lo que destaca a PSO de otros algoritmos es su fácil implementación, robustez en control de parámetros y eficiencia computacional, por lo que los campos de estudio y los problemas en los que se puede utilizar son numerosos. Inicialmente, PSO se aplicó para resolver múltiples problemas matemáticos con funciones no lineales, funciones multimodales con y sin restricciones [1]. Debido a su aceptación, esta técnica no sólo se puede encontrar en artículos para análisis matemático o estadístico, sino también hace su incursión en el diseño de antenas, optimización biomecánica, *clustering*, minería de datos, optimización combinatoria, solución de ecuaciones implicadas en leyes de control, entrenamiento en redes neuronales, entre otros [2], [3], [4], [5], [6], [7], [8], [9], [10], [11].

II. Optimización de funciones de referencia

Para las pruebas desempeño del algoritmo PSO se utilizaron dos funciones de referencia para ser evaluadas, las cuales serán descritas a continuación.

A. Función Eggholder

El espacio de solución en esta función esta acotado en un rango de $-512 \geq x_i \leq 512$ para las variables de solución, con un mínimo global ubicado en $f_{EH}(512, 404.2319) = -956.6407$ [12], y su representación, en una vista isométrica, se pude observar en Fig. 1. La función en su forma matemática está definida como:

$$f_{EH}(x_1, x_2) = -(x_2 + 47)\sin\left(\sqrt{\left|x_2 + \frac{x_1}{2} + 47\right|}\right) - x_1 \sin\left(\sqrt{|x_1 - (x_2 + 47)|}\right) \quad (6)$$

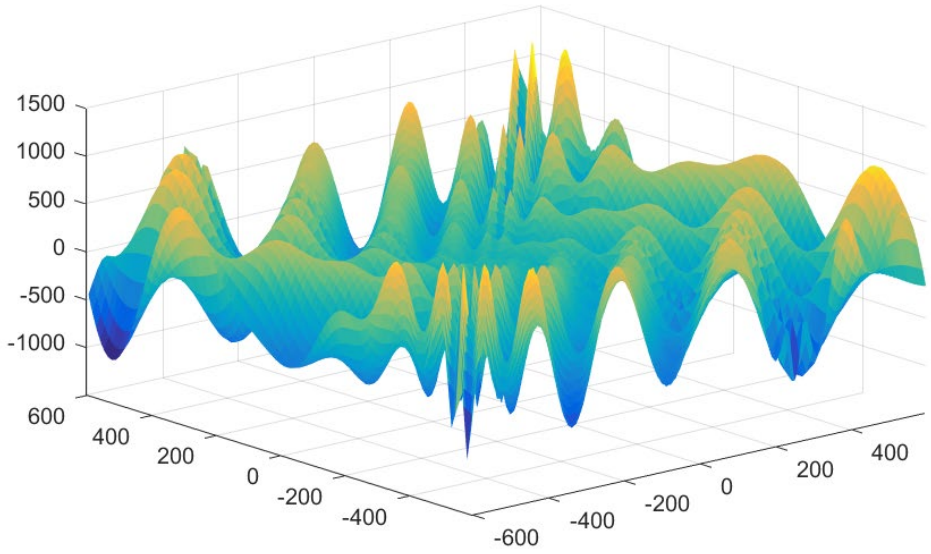

Fig. 1. Representación gráfica de la función Eggholder.

B. Función Six-Hump Camelback

Esta función de prueba tiene dos parámetros como variables de búsqueda, cuenta con seis mínimos locales, donde dos de ellos son globales. El área de prueba se encuentra normalmente en los rangos $-3 \geq x_1 \leq 3$ y $-2 \geq x_2 \leq 2$. Los mínimos globales se encuentran en $f_{SHC}(-0.0898, 0.7126) = -1.0316$ y $f_{SHC}(0.0898, -0.7126) = -1.0316$, su representación en vista isométrica se presenta en la Fig. 2 [13].

$$f_{SHC}(x_1, x_2) = \left(4 - 2 - 1x_1^2 + \frac{x_1^4}{3}\right)x_1^2 + x_1x_2 + (-4 + 4x_2^2)x_2^2 \quad (7)$$

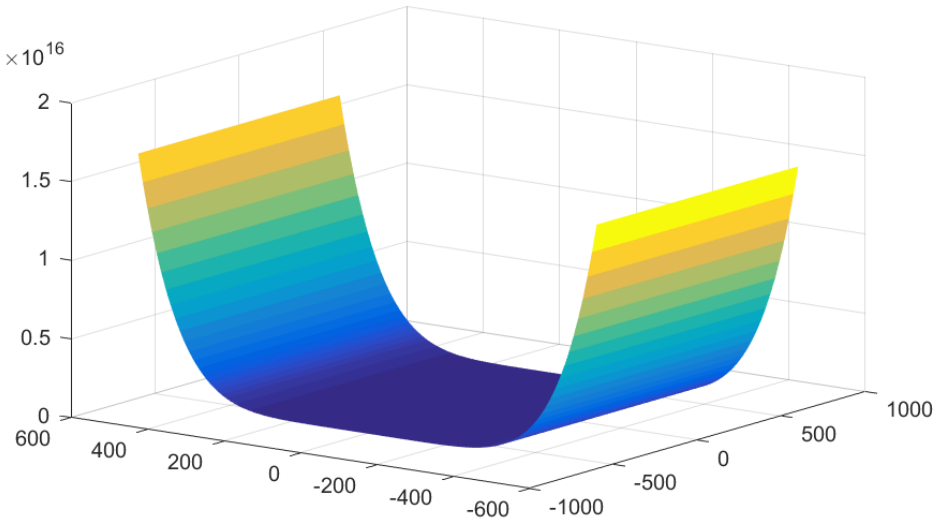

Fig. 2. Representación gráfica de la función Six-Hump Camelback.

III. Métodos para el cálculo del descenso del parámetro de inercia

El parámetro nombrado "inercia" se encarga de regular el vector de velocidad asociado a cada partícula. Disminuir este parámetro, durante la ejecución de algoritmo ha demostrado que mejora el desempeño del algortimo PSO [14], [15], aunque en el trabajo originalmente no se contemplaba [1]. Fue hasta 1998, que por primera vez Shi y Eberhart [16] introdujeron el concepto de masa inercial al aplicar un coeficiente de inercia para limitar la velocidad de las partículas en el algoritmo PSO; a su vez establecieron que dicho coeficiente facilitaba la búsqueda global cuando se trataba de un número grande, mientras que si se trataba de un número pequeño se facilitaba la búsqueda local.

Sin embargo, se ha demostrado en numerosos estudios que el parámetro de inercia, con ajuste dinámico en el tiempo, puede aumentar significativamente la convergencia de una solución, comparándolo con un valor constante [14], [17], [18].

Para las pruebas realizadas se consideraron dos métodos de descenso de la inercia: lineal y caótico.

A. Método de descenso inercial lineal

Este método ajusta w (Ecu. 5) con la siguiente fórmula:

$$w(t+1) = (w_{max} - w_{min})\left(\frac{t_{max} - t}{t_{max}}\right) + w_{min} \quad (8)$$

De donde w_{max} es el valor inicial del parámetro de inercia al iniciar el algoritmo y w_{min} representa el valor final. Se ha reportado que el rango idóneo para el parámetro w es de [0.9 - 0.4] [19]. El método de descenso lineal habitualmente presenta problemas de convergencia con funciones que tiene una mayor cantidad de variables en su solución.

B. Método de descenso inercial caótico

La inercia caótica fue propuesta por Feng *et. al* [15] en donde agrega al método de descenso lineal un factor que genera un mapeo caótico en el rango [0,1]:

$$z = \mu(z)(z-1) \tag{9}$$

donde $3.57 < \mu \leq 4$ y z es un número aleatorio de [0,1]. Sin embargo, cuando $\mu = 4$ se cubre el intervalo de [0,1]. Se ajusta entonces con la siguiente fórmula

$$w(t+1) = (w_{max} - w_{min})\left(\frac{t_{max}-t}{t_{max}}\right) + w_{min}(z) \tag{10}$$

IV. METODOLOGÍA

Se realizan dos estudios donde se modifica el método de descenso inercial, y que para cada uno de los estudios se analizarán los resultados para las dos funciones de prueba (Ec. 6, Ec. 7).

Para ambos estudios se establecen los parámetros mostrados en la Tabla 1.

TABLA 1
PARÁMETROS GENERALES PARA EL ALGORITMO PSO PARA LOS ESTUDIOS PROPUESTOS

Parámetros	**Valores**
$[c_1, c_2]$	$[2.5, 2.5]$
Máx. núm. iteraciones	1000
Límites eje *x*	$[-512, 512]$
Límites eje *y*	$[-512, 512]$
Partículas	250 - 1000
Corridas	200

A. Estudio 1. Método de descenso inercial lineal

Para este estudio se considera el uso de la Ec. 8 para su implementación dentro del algoritmo PSO, analizándose las funciones Eggholder (Ec. 6) y Six-Hump Camelback (Ec. 7), considerando que ambas dependen de dos variables.

B. Estudio 2. Método de descenso inercial caótico

Para este estudio se considera el uso de la Ec. 10 para su implementación dentro del algoritmo PSO, analizándose las funciones Eggholder (Ec. 6) y Six-Hump Camelback (Ec. 7), considerando que ambas dependen de dos variables.

La construcción del algoritmo, así como la realización de los estudios y obtención de resultados se realiza a través del software MATLAB® versión 14.0 en un sistema operativo Windows 10 con plataforma de 64-bits.

Los resultados se darán con base en una comparación entre las dos funciones de referencia. Para cada una de ellas se han tomado en cuenta tres factores:

1. Iteración: número de iteración en la que el sistema detecto el mínimo global.
2. Costo: valor obtenido al evaluar matemáticamente la función de referencia.
3. Gráficos asociados al registro de múltiples pruebas realizadas, empleando las funciones de referencia.

Para lo anterior se considerarán datos estadísticos.

Científica, vol. 25, núm. 1, pp. 01-11, enero-junio 2021, ISSN 2594-2921, Instituto Politécnico Nacional MÉXICO
DOI: https://doi.org/10.46842/ipn.cien.v25n1a09

V. Resultados

A. Estadísticas asociadas al número de iteraciones

En la Tabla 2 se presentan los datos estadísticos asociados a las iteraciones en las que se alcanzó el costo óptimo (este costo óptimo corresponde al valor numérico obtenido al evalúa la función de referencia con los datos que presenta como solución final el algoritmo PSO), para cada una de las funciones en el Estudio 1, mientras que en la Tabla 3 se muestran los datos para el Estudio 2.

Tabla 2
Estadística asociada al número de iteración en que se alcanza el costo óptimo para un método de descenso inercial lineal

	Función Eggholder			**Función Six-Hump Camelback**		
	Posición	**Costo**	**Iteración**	**Posición**	**Costo**	**Iteración**
Iteración Mínima	[512, 404.2318]	-959.6407	**438**	[-0.088 0.713]	-1.0314	**820**
Iteración Máxima	[512, 404.2318]	-959.6407	**972**	[-0.088 0.713]	-1.0314	**1000**
Mediana de las Iteraciones	[512, 404.2318]	-959.6407	**607**	[0.088 -0.713]	-1.0314	**969**

Tabla 3
Estadística asociada al número de iteración en que se alcanza el costo óptimo para un método de descenso inercial caótico

	Función Eggholder			**Función Six-Hump Camelback**		
	Posición	**Costo**	**Iteración**	**Posición**	**Costo**	**Iteración**
Iteración Mínima	[512, 404.2318]	-959.6407	**32**	[-0.088 0.713]	-1.0314	**68**
Iteración Máxima	[512, 404.2318]	-959.6407	**360**	[-0.088 0.713]	-1.0314	**455**
Mediana de las Iteraciones	[512, 404.2318]	-959.6407	**56**	[0.088 -0.713]	-1.0314	**154**

B. Estadísticas asociadas al costo

Para este caso se debe resaltar que para la función Six-Hump Camelback no se realiza esta comparación, ya que no hubo variación en el valor para todas las corridas realizadas en ambos estudios, por lo que sólo se mencionarán los datos asociados a los estudios realizados con la función Eggholder. En la Tabla 4 se muestran los resultados para ambos estudios.

Tabla 4
Estadística asociada al costo encontrado por el algoritmo para la función Eggholder usando un método de descenso inercial lineal y caótico

	Inercia Lineal			**Inercia Caótica**		
	Posición	**Costo**	**Media de Iteración**	**Posición**	**Costo**	**Media de Iteración**
Costo Mínimo	[512, 404.2318]	-959.6407	**584**	[512, 404.2318]	-959.6407	**59.8496**
Costo Máximo	[283.0759, -487.1257]	-718.1675	**808**	[283.0759, -487.1257]	-718.1675	**54**
Mediana del Costo	[512, 404.2318]	-959.6407	**584**	[482.353, 432.879]	-959.6407	**59.8496**

C. Historial de convergencia asociado a las estadísticas de las iteraciones

Para esta sección se mostrarán los gráficos del historial de convergencia, comparando para cada función el método de descenso lineal y el método de descenso caótico. Para ello se utilizará la corrida asociada a la iteración mínima, así como la media de todas las corridas realizadas.

En la Fig. 3 se muestra el historial para la corrida asociada a la iteración mínima de la función Eggholder.

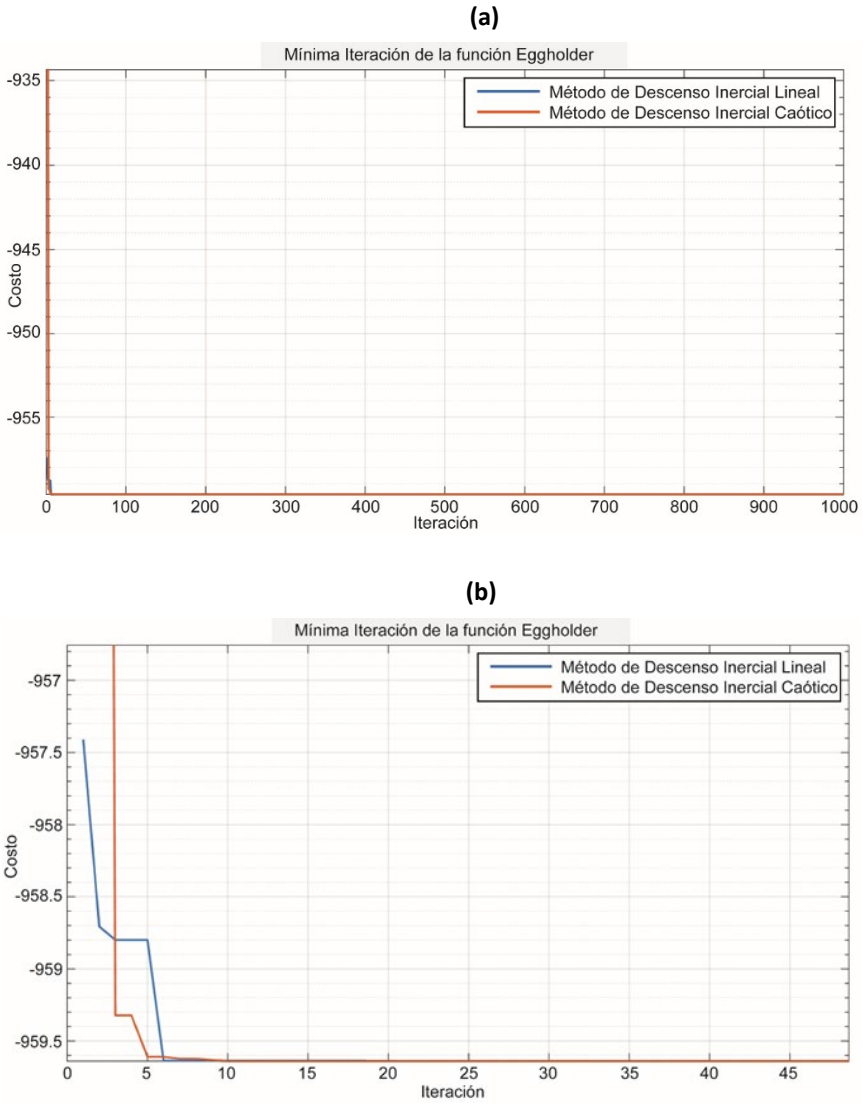

Fig. 1. Historial de convergencia de los dos métodos de variación del parámetro de inercia de la corrida con la mínima iteración de la función Eggholder. (a) Muestra el historial completo y (b) un acercamiento donde se observa de la iteración 0 a la 50, aproximadamente.

En la Fig. 4 se aprecia el historial de convergencia promedio para la función Eggholder.

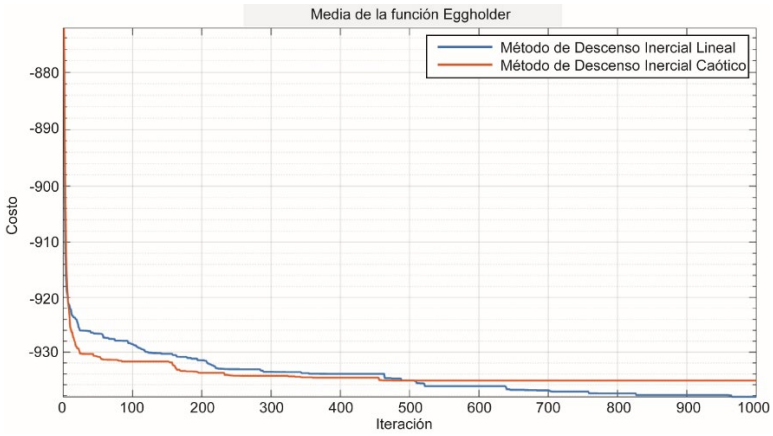

Fig. 2. Historial de convergencia promedio de los dos métodos de variación del parámetro de inercia de la función Eggholder. Se muestra el historial completo.

Para la función Six-Hump Camelback se muestra en la Fig. 5 el historial asociado a la corrida con la mínima iteración de convergencia. Mientras que en la Fig. 6 se observa el historial de convergencia promedio.

(a)

(b)

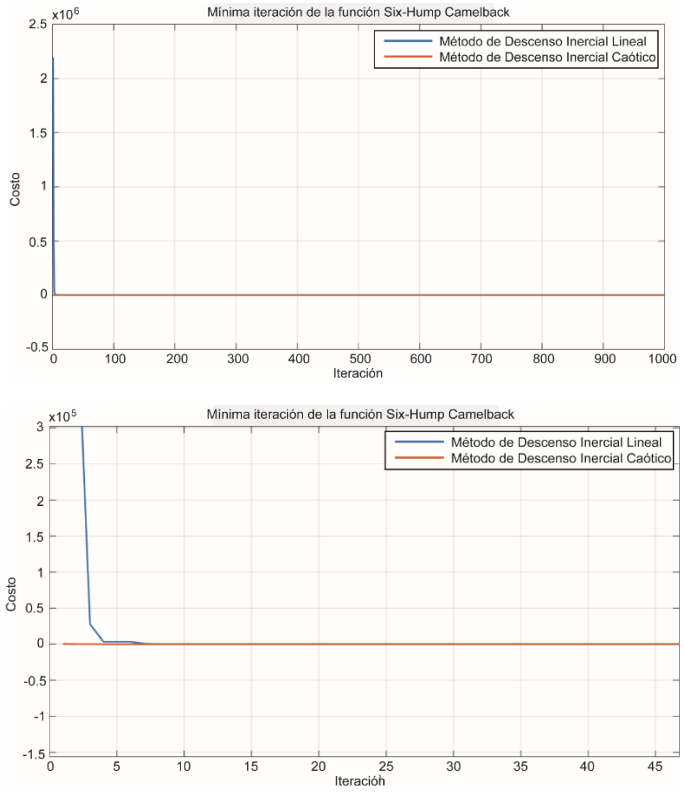

Fig. 3. Historial de convergencia de los dos métodos de variación del parámetro de inercia de la corrida con la mínima iteración de la función Six-Hump Camelback. **(a)** Muestra el historial completo y **(b)** un acercamiento donde se observa de la iteración 0 a la 45, aproximadamente.

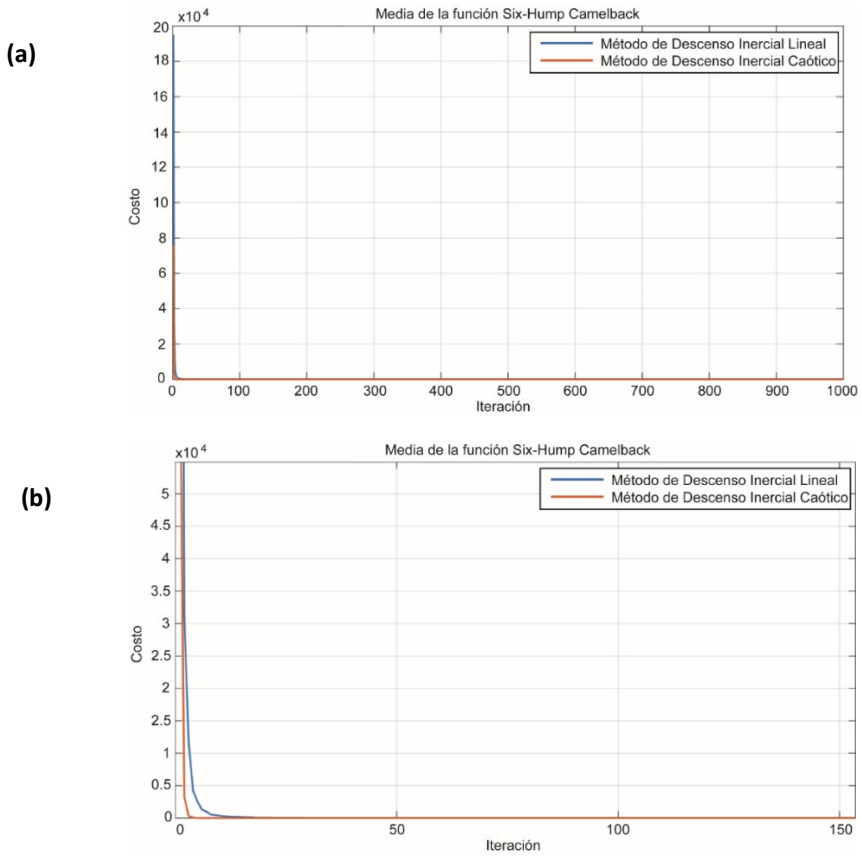

Fig. 4. Historial de convergencia promedio de los dos métodos de variación del parámetro de inercia de la función Six-Hump Camelback. (a) Muestra el historial completo y (b) un acercamiento donde se observa de la iteración 0 a la 150, aproximadamente.

VI. Discusión de resultados

A partir de los resultados obtenidos, al considerar los criterios de iteración, costo y el registro de resultados en múltiples pruebas de convergencia aplicado en las funciones de referencia, se puede mencionar que, los valores numéricos al evaluar las funciones de referencia con los parámetros obtenidos al aplicar el algoritmo PSO, no presentaron diferencias significativas al considerar la variación del valor del parámetro de inercia de forma caótica o de forma lineal.

Para la función Six-Hump Camelback, se obtuvo un valor de −1.0314 al evaluar la función con los parámetros obtenidos al ejecutar el algoritmo PSO en 200 ocasiones, variando únicamente en la posición de los dos puntos considerados como mínimos globales, establecido en los antecedentes teóricos.

Para el caso de la función Eggholder, nuevamente se ejecutó el algoritmo PSO en 200 ocasiones, tanto para la variación del valor del parámetro de inercia de forma caótica y lineal. Para los dos tipos de variación, en ciertas ocasiones se alcanzó el valor de −718.1675 al evaluar la función; sin embargo, el mejor costo que se puede obtener con esta función de prueba es de −959.6407. Esta diferencia es debida a la múltiple cantidad de mínimos locales, que caracteriza a esta función y a los valores iniciales de las partículas [20].

Entonces, la diferencia entre ambos métodos de variación del parámetro de inercia se puede observar en las estadísticas asociadas al número de iteraciones en que se alcanza la convergencia al ser ejecutado el algoritmo PSO en una ocasión. Para la inercia lineal, es necesario un número mayor de iteraciones para alcanzar la convergencia en comparación a lo requerido en inercia caótica.

Además de las características estadísticas, se tienen los gráficos del historial de convergencia, de los cuales específicamente, los asociados a la media de las corridas muestran aspectos interesantes respecto a la velocidad de convergencia. Para la función Eggholder se observó una convergencia más rápida en el método de descenso caótico con respecto al método de descenso lineal, sin embargo, para este último se observa que se llegó al resultado óptimo en más ocasiones con respecto al descenso caótico (véase Fig. 4), es decir, que llega al valor esperado en un mayor número de ocasiones con respecto al método caótico.

Para el caso de la función Six-Hump Camelback se pueden percibir dos situaciones: Primero, el método de descenso caótico es más rápido y certero al llegar al costo óptimo. Segundo; desde la primera iteración, el valor del costo es más cercano al valor óptimo (véase Fig. 6).

VII. Conclusiones

En este trabajo se hace un comparativo de la modificación del parámetro de inercia del modelo PSO, entre dos métodos, el lineal y el caótico. De acuerdo con el análisis de resultados se puede mencionar principalmente que el hecho de que para el método caótico se requiera de menos iteraciones para alcanzar el costo óptimo en relación con el número de iteraciones para el método lineal, se puede asociar con la eficacia del método, es decir, que el método de descenso inercial caótico representa un mejor modelo para la obtención de mínimos sin requerir de una gran cantidad de iteraciones.

Esta razón permite establecer parámetros para futuros trabajos, recomendando el uso del método de descenso inercial caótico, así como la utilización de determinado número de iteraciones máximas, la cantidad de veces que hay que accionar el algoritmo, mejorando de esta manera la eficiencia computacional y asegurando la obtención de mejores resultados.

Referencias

[1] J. Kennedy, R. Eberhart, "Particle swarm optimization," Proceedings of ICNN'95 - International Conference on Neural Networks, Perth, WA, Australia, 1995, pp. 1942-1948 vol.4, doi: https://doi.org/10.1109/ICNN.1995.488968

[2] E. Ozcan, C. K. Mohan, "Particle swarm optimization: surfing the waves," Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406), Washington, DC, USA, 1999, pp. 1939-1944, vol. 3, doi: https://doi.org/10.1109/CEC.1999.785510

[3] R. Poli, "Analysis of the Publications on the Applications of Particle Swarm Optimisation", *Journal of Artificial Evolution and Applications*, 2008, pp. 1-10, vol. 2008, doi: https://doi.org/10.1155/2008/685175

[4] N. Jin, Y. Rahmat-Samii, "Advances in Particle Swarm Optimization for Antenna Designs: Real-Number, Binary, Single-Objective and Multiobjective Implementations," en *IEEE Transactions on Antennas and Propagation*, vol. 55, no. 3, pp. 556-567, Mar. 2007, doi: https://doi.org/10.1109/TAP.2007.891552

[5] M. P. Wachowiak, R. Smolikova, Yufeng Zheng, J. M. Zurada, A. S. Elmaghraby, "An approach to multimodal biomedical image registration utilizing particle swarm optimization," en *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 289-301, Jun. 2004, doi: https://doi.org/10.1109/TEVC.2004.826068

[6] M. Houcine, S. Amin, C. Mondher, M. Nouri, "Improved Particle Swarm Optimization for the Mutual Inductance of an Implantable Biomedical Application," *2019 International Conference on Wireless Technologies, Embedded and Intelligent Systems (WITS)*, Fez, Morocco, 2019, pp. 1-6, doi: https://doi.org/10.1109/WITS.2019.8723760

[7] S. Fong, R. Wong, A. V. Vasilakos, "Accelerated PSO Swarm Search Feature Selection for Data Stream Mining Big Data," en *IEEE Transactions on Services Computing*, vol. 9, no. 1, pp. 33-45, 1 Ene.-Feb. 2016, doi: https://doi.org/10.1109/TSC.2015.2439695

[8] S. Easter Selvan, S Subramanian, S. Theban Solomon, "Novel technique for PID tuning by particle swarm optimization", *Seventh Annual Swarm Users/Researchers Conference,* 2003.

[9] J. Salerno, "Using the particle swarm optimization technique to train a recurrent neural model," *Proceedings Ninth IEEE International Conference on Tools with Artificial Intelligence*, Newport Beach, CA, USA, 1997, pp. 45-49, doi: https://doi.org/10.1109/TAI.1997.632235

[10] A. I. El-Gallas, M. El-Hawary, A. A. Sallam, A. Kalas, "Swarm-intelligently trained neural network for power transformer protection," *Canadian Conference on Electrical and Computer Engineering 2001. Conference Proceedings (Cat. No.01TH8555)*, Toronto, Ontario, Canada, 2001, pp. 265-269 vol.1, doi: https://doi.org/10.1109/CCECE.2001.933694

[11] A. Chatterjee, K. Pulasinghe, K. Watanabe, K. Izumi, "A particle-swarm-optimized fuzzy-neural network for voice-controlled robot systems," in *IEEE Transactions on Industrial Electronics*, vol. 52, no. 6, pp. 1478-1489, Dec. 2005, doi: https://doi.org/10.1109/TIE.2005.858737

[12] R. Ernesto, L. Ernesto, B. Rafael, G. Yolanda, "Perfiles de comportamiento numérico de los métodos de búsqueda immune network algorithm y bacterial foraging optimization algorithm en funciones benchmark", *Ingeniería, Investigación y Tecnología*, vol. 17, no. 4, pp. 479-490, 2016, doi: https://doi.org/10.1016/j.riit.2016.11.007

[13] X. Yang, *Engineering Optimization: An Introduction with Metaheuristic Applications*, 1a ed. UK: John Wiley & Sons, 2010, p. 264.

[14] R. C. Eberhart, Yuhui Shi, "Tracking and optimizing dynamic systems with particle swarms," *Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No.01TH8546)*, Seoul, South Korea, 2001, pp. 94-100 vol. 1, doi: https://doi.org/10.1109/CEC.2001.934376

[15] Y. Feng, G. Teng, A. Wang, Y. Yao, "Chaotic Inertia Weight in Particle Swarm Optimization," *Second International Conference on Innovative Computing, Information and Control (ICICIC 2007)*, Kumamoto, 2007, pp. 475-475, doi: https://doi.org/10.1109/ICICIC.2007.209

[16] Y. Shi, R. Eberhart, "A modified particle swarm optimizer," *1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360)*, Anchorage, AK, USA, 1998, pp. 69-73, doi: https://doi.org/10.1109/ICEC.1998.699146

[17] J. Xin, G. Chen, Y. Hai, "A Particle Swarm Optimizer with Multi-stage Linearly-Decreasing Inertia Weight," *2009 International Joint Conference on Computational Sciences and Optimization*, Sanya, Hainan, 2009, pp. 505-508, doi: https://doi.org/10.1109/CSO.2009.420

[18] M. Arumugam, M. Rao, "On the performance of the particle swarm optimization algorithm with various inertia weight variants for computing optimal control of a class of hybrid systems", *Discrete Dynamics in Nature and Society*, vol. 2006, pp. 1-17, 2006, doi: https://doi.org/10.1155/ddns/2006/79295

[19] R. C. Eberhart, Y. Shi, "Comparing inertia weights and constriction factors in particle swarm optimization," *Proceedings of the 2000 Congress on Evolutionary Computation. CEC00 (Cat. No.00TH8512)*, La Jolla, CA, USA, 2000, pp. 84-88 vol.1, doi: https://doi.org/10.1109/CEC.2000.870279

[20] J. Czerniak, D. Ewald, H. Zarzycki, P. Augustyn, "Application of the New FAAO Metaheuristics in Modeling and Simulation of the Search for the Optimum of a Function with Many Extremes", *Advances in Intelligent Systems and Computing*, pp. 301-309, 2020, doi: https://doi.org/10.1007/978-3-030-47024-1_30

Científica, vol. 25, núm. 1, pp. 01-11, enero-junio 2021, ISSN 2594-2921, Instituto Politécnico Nacional MÉXICO
DOI: https://doi.org/10.46842/ipn.cien.v25n1a09