

# VALIDACIÓN DE UN ALGORITMO HÍBRIDO DEL PSO CON EL MÉTODO SIMPLEX Y DE TOPOLOGÍA DE EVOLUCIÓN PARAMÉTRICA

## VALIDATION OF A PSO-SIMPLEX HYBRID ALGORITHM OF PARAMETRIC EVOLUTION TOPOLOGY

RODRIGO CORREA

*Escuela de Ingenierías Eléctrica, Electrónica y de Telecomunicaciones, Universidad Industrial de Santander, rcorrea@uis.edu.co*

OSCAR BEGAMBRE

*Escuela de Ingeniería Civil, Universidad Industrial de Santander, ojbegam@uis.edu.co*

JULIO C. CARRILLO E.

*Escuela de Matemáticas, Universidad Industrial de Santander, jccarril@uis.edu.co*

Recibido para revisar enero 17 de 2009, aceptado marzo 2 de 2010, versión final abril 12 de 2010

**RESUMEN:** Este artículo describe algunos de los aspectos más importantes relacionados con la experimentación numérica de un híbrido del algoritmo PSO (*Particle Swarm Optimization*) con el tradicional método simplex modificado de Nelder-Mead. El híbrido de estas dos técnicas de optimización sin restricciones se realizó con una topología que permite optimizar en cada iteración los parámetros del algoritmo PSO utilizando el método simplex modificado. Se realizaron experimentos numéricos con este algoritmo híbrido aplicados a varias funciones de prueba típicas para establecer su efectividad. Los resultados obtenidos se compararon con los del método simplex y el método cuadrático, los cuales resultaron ser muy satisfactorios desde el punto de vista de su repetibilidad y reproducibilidad, aunque el tiempo de cómputo fue considerablemente mayor. Se resalta, sin embargo, que la precisión del algoritmo híbrido fue del cien por ciento en todos los ensayos para las funciones de prueba seleccionadas.

**PALABRAS CLAVE:** Optimización sin restricciones, métodos heurísticos, métodos estocásticos.

**ABSTRACT :** This paper describes some of the most important aspects related to the numerical experimentation of a hybrid of the algorithm PSO (Particle Swarm Optimization) with the traditional modified simplex method of Nelder-Mead. The hybridization of these two techniques of optimization without restrictions was carried out with a topology that allows to optimize in each iteration the parameters of the algorithm PSO using the modified simplex method. Numerical experiments with this hybrid algorithm were carried out and applied to several of typical test functions to establish its effectiveness. The results obtained were compared with the simplex and the quadratic methods, which turned out to be very satisfactory since the point of view of their repeatability and reproducibility, although the time of computation was considerably longer. It stands out itself, nevertheless, that the precision of the hybrid algorithm was a hundred percent in all the trials for the test functions selected.

**KEYWORDS:** Optimization without restriction, heuristic methods, stochastic methods.

### 1. INTRODUCCIÓN

En la actualidad se disponen de algoritmos capaces de resolver problemas de optimización cada vez más complicados. Una buena parte de

estos algoritmos se fundamenta en teorías heurísticas que plantean novedosas formas de buscar la solución de tales problemas mediante métodos no rigurosos, como por ejemplo, por tanteo, reglas empíricas y analogías naturales,

entre otras. La contrapartida formal en computación de este tipo de métodos es el algoritmo, por lo que reciben el nombre de *algoritmos heurísticos*. Un algoritmo heurístico comprende un conjunto de pasos que se deben realizar para identificar en el menor tiempo posible una solución de alta calidad para un determinado problema de optimización. De otro lado, un *algoritmo metaheurístico* es una estrategia de alto nivel que fusiona sinérgicamente dos o más heurísticas para buscar soluciones factibles en dominios donde la tarea es compleja. Entre las técnicas (los algoritmos) metaheurísticas más conocidas se encuentran la de colonia de hormigas, la búsqueda Tabú, de búsqueda local, la simulación del templado, los algoritmos genéticos y el enjambre de partículas. Las ventajas del uso de los algoritmos heurísticos se hacen evidentes cuando no existe una solución matemática simple a las ecuaciones que definen el problema tratado, cuando hay incertidumbre de los datos y cuando existen múltiples puntos óptimos en problemas de optimización no lineal. Además, también se deben considerar las facilidades de formulación y programación de este tipo de algoritmos, y tener en cuenta las capacidades de uso más eficiente de un procesador. Es de notar que los algoritmos heurísticos son independientes de la estimación del punto de partida, lo cual permite garantizar de cierta manera su convergencia, y que además están adaptados para encontrar soluciones globales. En comparación con los métodos basados en gradientes o derivadas de segundo orden, la principal desventaja de estos algoritmos radica en que necesitan un mayor número de evaluaciones de la función objetivo para encontrar una solución óptima, aspecto que incrementa notablemente su tiempo de ejecución. De otro lado, los métodos clásicos son mucho más susceptibles a converger a óptimos locales que los algoritmos heurísticos, siendo no recomendable emplearlos en problemas no lineales con múltiples puntos óptimos. Adicionalmente, los métodos clásicos son altamente sensibles a la presencia de ruido, debido a que los gradientes usados dependen de cualquier error en la medición o en el modelado. En la Tabla 1 se presenta una comparación entre estos métodos (ver [1]-[3]). Con el híbrido

propuesto en este artículo se espera darle una mayor robustez y precisión al PSO convencional.

**Tabla 1.** Comparación entre los métodos de optimización heurísticos y los clásicos

**Table 1.** Comparison among heuristic and classic optimization methods

Heurísticos	Clásicos
Soluciones globales	Soluciones locales
No requieren cálculo de derivadas	Es necesario el cálculo de derivadas
No dependen del punto inicial	Dependen del punto inicial
Sin restricciones de convexidad y/o continuidad	Con restricciones de convexidad y/o continuidad
Menor propagación de error	Mayor sensibilidad al error
Mayor número de evaluaciones de la función objetivo	Menor número de evaluaciones de la función objetivo
Programación basada en operaciones lógicas sencillas	La complejidad de la programación depende del problema
Basado en reglas heurísticas	Basado en pensamiento matemático
Literatura fragmentada	Libros, publicaciones, etc.

## 2. FUNDAMENTOS

El método simplex es uno de los métodos clásicos más comunes para resolver problemas de optimización sin restricciones (ver [4] para una descripción detallada). A continuación se describen algunos de los aspectos más relevantes de la versión convencional del algoritmo de enjambre de partículas PSO y de su híbrido con el método simplex, el algoritmo PSOSX.

### 2.1. El algoritmo PSO convencional

Esta técnica metaheurística evolutiva, inspirada en el comportamiento social de las bandadas de pájaros, fue propuesta por primera vez por Kennedy y Eberhart en 1995 (ver [5]) y hace parte de una categoría más amplia de métodos, conocidos como inteligencia de enjambre (*swarm intelligence*), utilizados para resolver problemas de programación no lineal. El algoritmo parte de la hipótesis de que cada

partícula (o punto dentro del dominio de la función objetivo que se propone como posible solución) se lanza a volar en el espacio de búsqueda guiadas por la partícula que mejor solución ha encontrado hasta el momento y que cumple con la función de líder de la bandada. Las partículas evolucionan teniendo en cuenta la mejor solución encontrada en su recorrido y la del líder. A continuación se presenta la fundamentación del algoritmo desde el punto de vista de la mecánica Newtoniana, que no fue precisamente su interpretación inicial. Si la posición en el espacio de una partícula, o elemento de un conjunto o enjambre, de masa  $m$  se expresa mediante el vector  $\vec{p}$  de  $N$  dimensiones, entonces su velocidad queda definida como

$$\vec{v} = \frac{d\vec{p}}{dt}. \quad (1)$$

De la segunda ley de Newton se llega a que la aceleración  $\vec{a}$  de la partícula y la fuerza  $\vec{F}$  que actúa sobre ella están relacionadas de la forma

$$\frac{d\vec{v}}{dt} = \vec{a} = \frac{\vec{F}}{m}. \quad (2)$$

Si se evalúan estas dos primeras derivadas mediante diferencias finitas y se despejan  $\vec{p}_t$  y  $\vec{v}_t$ , siendo  $t$  un entero positivo que identifica la iteración, se tiene que

$$\vec{p}_t = \vec{p}_{t-1} + \vec{v}_t \Delta t, \quad (3)$$

$$\vec{v}_t = \vec{v}_{t-1} + \frac{\vec{F}_{t-1}}{m} \Delta t. \quad (4)$$

Considerando la masa y el paso de tiempo *unitarios* se llega a las expresiones,

$$\vec{p}_t = \vec{p}_{t-1} + \vec{v}_t, \quad (5)$$

$$\vec{v}_t = \vec{v}_{t-1} + \vec{F}_{t-1}. \quad (6)$$

En este caso la fuerza  $\vec{F}_{t-1}$  se puede representar como una fuerza de atracción generada por resortes lineales, de la forma

$$\vec{F}_{t-1} = C_1(\vec{b}_{t-1} - \vec{p}_{t-1}) + C_2(\vec{g}_{t-1} - \vec{p}_{t-1}), \quad (7)$$

en donde  $\vec{b}_{t-1}$  y  $\vec{g}_{t-1}$  representan la mejor posición encontrada por la partícula y la mejor

posición global encontrada por todas las otras partículas, respectivamente. En su interpretación mecánica, los dos términos en (7) representan la dirección y longitud de dos resortes, siendo las constantes  $C_1$  y  $C_2$  las constantes de Hooke (ver [6]). Resumiendo, las dos expresiones que condensan el algoritmo PSO original quedan como

$$\vec{p}_t = \vec{p}_{t-1} + \vec{v}_t, \quad (8)$$

$$\vec{v}_t = \vec{v}_{t-1} + C_1(\vec{b}_{t-1} - \vec{p}_{t-1}) + C_2(\vec{g}_{t-1} - \vec{p}_{t-1}). \quad (9)$$

Si bien estas dos ecuaciones vectoriales describen el comportamiento de una partícula en forma determinística, la notación matricial involucrará todo el enjambre, por lo cual se asume que existen  $M$  partículas en un espacio de dimensión  $N$ . De esta manera, cada una de las variables quedará almacenada en matrices de orden  $M \times N$ . Ahora bien, dado que se trata de modelar un sistema natural, debe existir un cierto grado de aleatoriedad que se manifieste en el algoritmo. El proceso iterativo definido por el anterior sistema de ecuaciones matriciales establece una base de la forma como cada partícula utiliza el conocimiento cognitivo  $C_1$  y el social  $C_2$  para explorar el espacio de solución, pero de una manera determinística. En otras palabras, una vez el enjambre se inicializa, las trayectorias y posiciones finales de todas las partículas quedan definidas por estas expresiones. Sin embargo, en procesos que tratan de imitar la impredecible naturaleza se adapta el comportamiento del enjambre a diferentes ambientes al implementar factores estocásticos en el algoritmo. Esto se logra al multiplicar las constantes de Hook en (7) por sendos coeficientes aleatorios  $rand_1$  y  $rand_2$ , que son escalares con una distribución uniforme en el intervalo  $[0, 1]$ , o bien, son vectores con sus componentes distribuidos uniformemente en el mismo intervalo.

Así mismo, con el propósito de incrementar la convergencia del enjambre se propone desde un inicio un factor de peso inercial  $\omega$ , variable en el tiempo, como en ([5], [7]). Un valor típico para este factor es 0,9 al inicio del proceso de optimización, y va decreciendo hasta 0,4 al final del proceso. Igualmente reportan que los parámetros  $C_1$  y  $C_2$  mantienen un valor

constante e igual a dos, argumentando condiciones de convergencia. Así, con la actualización de la matriz de velocidad, las partículas evolucionan de acuerdo con el siguiente conjunto de ecuaciones matriciales, quedando de esta manera el algoritmo PSO original de la forma

$$\hat{P}_t = \hat{P}_{t-1} + \hat{V}_t, \quad (10)$$

$$\hat{V}_t = \omega \hat{V}_{t-1} + C_1 \text{rand}_1(\hat{B}_{t-1} - \hat{P}_{t-1}) + C_2 \text{rand}_2(\hat{G}_{t-1} - \hat{P}_{t-1}) \quad (11)$$

Dado que en las aplicaciones prácticas las fronteras del espacio de solución están usualmente definidas por las limitaciones geométricas, cualquier partícula fuera de estos límites no solamente conducirá a una solución sin significado físico, sino que también afectará en forma adversa el proceso de optimización. Para prevenir que cualquier partícula se salga de la solución de una manera frecuente se sugirió, desde el inicio de su desarrollo, que la velocidad máxima tuviera un valor de 0,2 veces el rango dinámico en cada dimensión de la partícula. Por ejemplo, si una variable está restringida dentro del intervalo  $(a_1, a_2)$ , la máxima velocidad en esta dimensión debe ser en el rango dinámico  $0,2|a_1 - a_2|$  en ambas direcciones. Otros autores sugieren una técnica para la definición de las condiciones de frontera clasificándolas como paredes o límites absorbentes, reflectivas e invisibles (ver [7]-[9]). Cuando una partícula se aproxima a una frontera absorbente su velocidad en la dimensión asociada es completamente absorbida por la frontera y se convierte en cero. En la siguiente interacción, la partícula únicamente se mueve en otra dimensión y es mantenida en el espacio de soluciones. Cuando la partícula toca una frontera reflectiva la velocidad se invierte en lugar de hacerse cero. La partícula reflejada permanece en el espacio de soluciones y está disponible para la siguiente evaluación en la función objetivo. En la condición de frontera invisible a la partícula se le permite pasar a través de la frontera manteniendo su propia velocidad original. Para evitar la contaminación de la información se le adjudica directamente un valor de bondad de ajuste erróneo a esta partícula, sin que se haya realizado la evaluación de la función objetivo. En [8] se demuestra que el costo computacional

se reduce y que estas condiciones de fronteras invisibles ayudan a la convergencia del algoritmo. En literatura más reciente, se proponen algunas otras técnicas de manejo de las condiciones de frontera, tales como las denominadas condiciones de frontera híbridas amortiguadas. En ellas se utilizan las características positivas de las condiciones de frontera absorbentes y reflectivas simultáneamente (ver [9] y [10]).

Los parámetros cognitivo  $C_1$  y social  $C_2$  controlan el flujo de información dentro del enjambre. Si  $C_2 > C_1$  entonces la partícula va a *confiar* más en los resultados de la búsqueda del enjambre que en los propios. En caso contrario, esto es cuando  $C_2 < C_1$ , la partícula va a tener más *confianza* en los resultados de su búsqueda que en los del resto del enjambre. Estos aspectos se ilustran en las Figuras 1 y 2, utilizando como ejemplo la función de Venter,

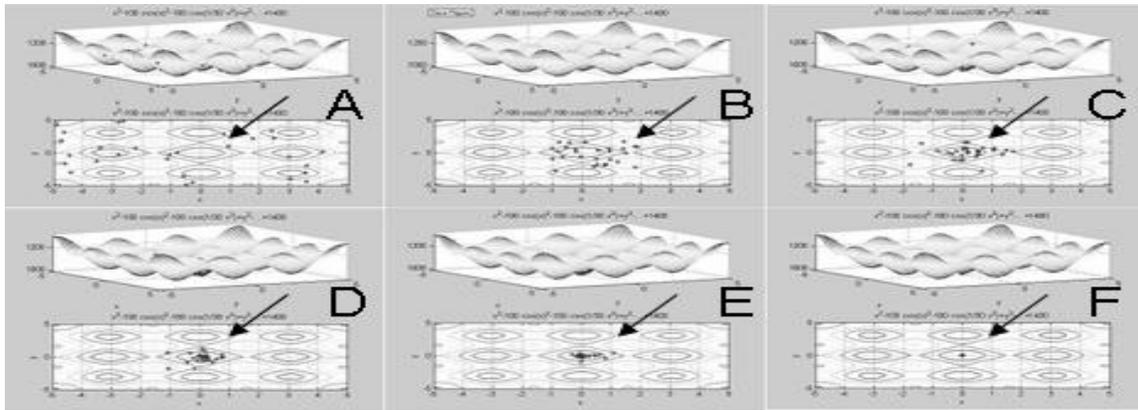
$$z(x, y) = x^2 - 100 \cos(x^2) - 100 \cos\left(\frac{x^2}{30}\right) + y^2 - 100 \cos(y^2) - 100 \cos\left(\frac{y^2}{30}\right) + 1400, \quad (12)$$

la cual alcanza el mínimo global 1000 en el punto de coordenadas (0,0). A manera de ejemplo, en la Figura 1 se muestra una secuencia de las iteraciones del algoritmo PSO cuando  $C_2 > C_1$ . Se observa que todas las partículas del enjambre convergen al mínimo global conforme crece el número de iteraciones. Caso contrario sucede cuando  $C_2 < C_1$ , observándose que al final existen partículas alejadas del enjambre debido a la *confianza* que tienen en su búsqueda, a pesar de lo cual el algoritmo converge al óptimo global.

El factor de inercia  $\omega$  controla la influencia de la velocidad previa de la partícula sobre la velocidad actual. Un factor de inercia alto facilita la exploración global del espacio de búsqueda, mientras que un valor pequeño posibilita realizar una búsqueda local. Es posible reducir el número de iteraciones del algoritmo PSO al encontrar un equilibrio entre las capacidades de búsqueda local y de búsqueda global del algoritmo. Con el fin de mantener éste equilibrio usualmente se utiliza un rango de valores para estos parámetros. A pesar de los

valores guías reportados en la literatura, la escogencia de estos parámetros depende de las particularidades del problema a resolver, por lo

el espacio de parámetros, de modo que asegure una configuración *cuasi* óptima para que el



**Figura 1.** Comportamiento del enjambre cuando  $C_2 > C_1$ . El recuadro **A** corresponde a la primera iteración, **B** a la número 10, **C** a la 20, **D** a la 30, **E** a la 40 y **F** a la iteración final

**Figure 1.** Swarm's behavior when  $C_2 > C_1$ . Graph **A** corresponds to the first iteration, **B** to number 10, **C** to 20, **D** to 30, **E** to 40 and **F** to the final iteration

cual se deben realizar pruebas exhaustivas para encontrar el mejor conjunto de parámetros que garanticen el éxito de la búsqueda del algoritmo PSO en cada caso, dado que valores inadecuados de ellos pueden causar fallas en la convergencia del algoritmo. De otro lado, existen modificaciones posteriores a este algoritmo original, como la propuesta por Clerc [11], quien propuso la adición de otro coeficiente que garantice la convergencia del algoritmo. Esta modificación generó el algoritmo conocido hoy día como algoritmo PSO de convergencia garantizada (GCPSO). En el presente artículo se utilizará el algoritmo PSO convencional.

## 2.2. Algoritmo PSOSX de evolución paramétrica

Este algoritmo híbrido se desarrolla con el fin de incorporar una estrategia de búsqueda autoconfigurada de forma tal que se garantice un conjunto *cuasi* óptimo de parámetros del algoritmo PSO que aseguren el éxito en la búsqueda independientemente de las particularidades del problema estudiado [12-13]. La idea central de esta topología es hallar mediante el método simplex los mejores valores de los parámetros  $N$ ,  $\omega$ ,  $C_1$  y  $C_2$  contenidos en

algoritmo PSO realice la evaluación de la función objetivo en cada iteración. Dentro de esta heurística, cada vértice del método simplex queda definido por las coordenadas  $N$ ,  $\omega$ ,  $C_1$  y  $C_2$ . En este punto, el algoritmo PSO toma los valores de estos parámetros heurísticos y valida la función objetivo del problema de acuerdo a las ecuaciones (10) y (11) (ver [5], [11], [12]). A este esquema o topología de interacción del algoritmo PSO con el método simplex lo hemos denominado *algoritmo PSOSX de evolución paramétrica*. Otro esquema de interacción alternante entre ellos se describe en detalle en [13] y [14]. Para cada punto encontrado por el método simplex producto de cualquier reflexión, contracción, expansión o reducción, se valida por un enjambre independiente, caracterizado por el vector  $X_i(N_i, \omega_i, C_{1i}, C_{2i})$ ,  $i = 1, \dots, k + 1$ , en donde  $k$  es el número de parámetros del algoritmo PSO y  $N_i$  es un número entero que representa el tamaño del enjambre. Esta última variable fue incluida con la finalidad de estimar su valor óptimo. Tal topología mejora las capacidades de búsqueda del algoritmo PSOSX e independiza los parámetros heurísticos, debido a que realiza una búsqueda dirigida por parámetros *cuasi* óptimos de forma automática.

### 3. EVALUACIÓN EXPERIMENTAL Y ANÁLISIS DE RESULTADOS

Uno de los objetivos de este artículo es mostrar el desempeño del algoritmo PSOSX frente al método de optimización cuadrático y el método simplex, dos de los métodos numéricos de optimización de uso más frecuente en ingeniería. Debido a los buenos resultados entregados por el método de optimización cuadrático, este se ha convertido casi en un estándar, por lo que se usará para medir los resultados y las características del nuevo algoritmo híbrido PSOSX. Además, se confrontarán estos resultados únicamente con los del método simplex (Nelder-Mead). Como criterios de evaluación de los algoritmos se tendrán en cuenta la precisión, la exactitud y el costo computacional.

Con el fin de explorar las capacidades del algoritmo PSOSX en diferentes tipos de problemas se evaluaron ecuaciones en tres dimensiones que a la fecha se destacan como pruebas de rigor difíciles para cualquier procedimiento de optimización. En cada caso el método se evaluó 200 veces y los resultados se compararon con los respuestas obtenidas por algoritmos de gran precisión como lo son: algoritmo cuadrático (ejecutado con la función de MATLAB<sup>MR</sup> `quadprog`), algoritmo simplex (ejecutado con la función de MATLAB<sup>MR</sup> `fminsearch`) y método quasi-Newton (ejecutado con la función de MATLAB<sup>MR</sup> `fminunc`) [17]. Para las simulaciones se utilizó un computador DELL<sup>MR</sup> con procesador Intel Centrino Duo<sup>TM</sup>, 1GHz en memoria RAM y con sistema operativo Windows<sup>MR</sup> XP.

**Función Cuadrática:** La primera prueba con la que se evaluó el algoritmo PSOSX fue en la búsqueda del mínimo global de una función cuadrática,

$$z(x, y) = ax^2 + bxy + cy^2 \quad (13)$$

en donde  $a$ ,  $b$  y  $c$  son números reales dados y tales que  $|a| + |b| + |c| \neq 0$ . Para este tipo de función el método de optimización cuadrática ha mostrado gran robustez en la búsqueda del

óptimo global, razón por la que su aplicación se ha convertido casi en regla para el diseño de sistemas de control predictivos. La principal desventaja de su uso en otras áreas es que no todos los procesos se rigen por una función cuadrática, debiéndose ajustar la función objetivo del problema a una función de este tipo. En la mayoría de los casos se debe segmentar el dominio de la función objetivo y encontrar varios ajustes para cada segmento del dominio de forma que se pueda realizar una aproximación más exacta. Para aplicar el método cuadrático es necesario reescribir (13) en la forma matricial

$$z(\vec{x}) = \frac{1}{2}\vec{x}^T H \vec{x} + \vec{f}^T \vec{x}, (14)$$

pudiendo tener restricciones de la forma  $A\vec{x} \leq \vec{b}$ , donde  $H$  es la matriz Hessiana de la función a minimizar,  $A$  es la matriz de restricciones y  $\vec{f}$ ,  $\vec{b}$  y  $\vec{x}$  son vectores. En las Tablas 2 y 3 se presentan un análisis estadístico elemental de los resultados obtenidos con el algoritmo PSOSX, y de solamente el método simplex y el método cuadrático.

Es posible observar que el método cuadrático ofrece una excelente precisión en sus resultados al lograr una desviación estándar igual a cero. Por su parte el algoritmo PSOSX muestra una precisión aceptable logrando una desviación estándar de 2,177E-09, mientras que el método simplex presenta una desviación estándar igual a 9,162E-10. A pesar de mostrar excelentes resultados, la gran desventaja de usar el método cuadrático radica en la limitación que tiene al ser aplicado exclusivamente a funciones cuadráticas, lo que hace necesario que en cada problema se realice una aproximación de la función objetivo mediante una función que satisfaga (14).

Adicionalmente, el algoritmo PSOSX alcanza el 100% de exactitud en todos los casos de búsqueda, mostrando su efectividad a la hora de hallar el óptimo global de una función cuadrática. A pesar del mayor tiempo de cálculo en la ejecución que necesitó el algoritmo PSOSX respecto a sus competidores, debido principalmente a que los métodos heurísticos necesitan evaluar un mayor número de veces la función objetivo; su uso en problemas con

funciones complejas se resume a procedimientos de búsqueda sencillos, de gran robustez y con un

100% de efectividad en todos los casos probados, como se mostrará también con otras funciones.

**Tabla 2.** Análisis estadístico de los resultados de la evaluación de la función cuadrática para  $-5 < x, y < 5$

**Table 2.** Statistic Analysis of quadratic function evaluation results for  $-5 < x, y < 5$

Estadística		PSOS	Simplex(Nealder-Mead)	Cuadrático
Óptimo Global	Mejor Valor	-2.250000	-2.250000	-2.250000
	Media de las muestras	-2.250000	-2.250000	-2.250000
	Varianza	4.7408E-18	8.4318E-19	0.0000E+00
	Desviación Estandar	2.1773E-09	9.1626E-10	0.0000E+00
Coordenada X	Mejor Valor	1.500000	1.500001	1.500000
	Media de las muestras	1.499993	1.499997	1.500000
	Varianza	9.4155E-09	5.1101E-10	0.0000E+00
	Desviación Estandar	9.7033E-05	2.2653E-05	0.0000E+00
Coordenada Y	Mejor Valor	1.500001	1.500000	1.500000
	Media de las muestras	1.499993	1.499996	1.500000
	Varianza	1.5539E-08	8.2107E-10	0.0000E+00
	Desviación Estandar	1.2465E-04	2.8668E-05	0.0000E+00
Exactitud		100%	100%	100%
Tiempo Medio de Cálculo (s)		6.716	0.019	0.034

**Tabla 3.** Análisis estadístico de los resultados de la evaluación de la función cuadrática para  $-30 < x, y < 30$

**Table 3.** Statistic Analysis of quadratic function evaluation results for  $-30 < x, y < 30$

Estadística		PSOS	Simplex(Nealder-Mead)	Cuadrático
Óptimo Global	Mejor Valor	-2.250000	-2.250000	-2.250000
	Media de las muestras	-2.250000	-2.250000	-2.250000
	Varianza	2.1333E-09	5.7735E-19	0.0000E+00
	Desviación Estandar	4.5509E-18	7.5852E-10	0.0000E+00
Coordenada X	Mejor Valor	1.500000	1.500000	1.500000
	Media de las muestras	1.500003	1.500001	1.500000
	Varianza	8.8959E-05	4.7089E-10	0.0000E+00
	Desviación Estandar	7.9136E-09	2.1647E-05	0.0000E+00
Coordenada Y	Mejor Valor	1.499998	1.500000	1.500000
	Media de las muestras	1.499999	1.500002	1.500000
	Varianza	1.1102E-04	8.5604E-10	0.0000E+00
	Desviación Estandar	1.2326E-08	2.9227E-05	0.0000E+00
Exactitud		100%	100%	100%
Tiempo Medio de Cálculo (s)		8.065	0.019	0.021

**Primera función de prueba:** Con esta función se evalúa la capacidad de búsqueda del algoritmo en problemas con superficies regulares donde la pendiente de la zona que aloja el óptimo global es muy pequeña y se hace difícil identificar con exactitud este punto. Esta función se define de la forma

$$z(x, y) = ((y - x)^2)^2 + (x - 1)^2 + 4, \quad (15)$$

Esta función alcanza su mínimo global 4 en el punto de coordenadas (1, 1). La Tabla 4 reúne el análisis estadístico de los resultados.

**Tabla 4.** Análisis estadístico de los resultados de la evaluación de la función Test 1 para  $-5 < x, y < 5$ **Table 4.** Statistic Analysis of Test 1 function evaluation results for  $-5 < x, y < 5$ 

Estadística		PSOS	Simplex(Nealder-Mead)	Quasi Newton
Óptimo Global	Mejor Valor	4.000000	4.000000	4.000000
	Media de las muestras	4.000000	4.000000	4.000000
	Varianza	1.0313E-19	6.9916E-28	7.1550E-23
	Desviación Estandar	3.2114E-10	2.6442E-14	8.4587E-12
Coordenada X	Mejor Valor	1.000000	1.000000	1.000000
	Media de las muestras	1.000001	1.000000	1.000000
	Varianza	2.0461E-09	1.9321E-15	9.7764E-15
	Desviación Estandar	4.5234E-05	4.3956E-08	9.8876E-08
Coordenada Y	Mejor Valor	1.000000	1.000001	1.000001
	Media de las muestras	1.000334	0.999983	0.999938
	Varianza	2.0795E-05	1.5561E-08	2.1216E-06
	Desviación Estandar	4.5602E-03	1.2475E-04	1.4566E-03
Exactitud		100%	100%	100%
Tiempo Medio de Cálculo (s)		6.233	0.023	0.034

**Tabla 5.** Análisis estadístico de los resultados de la evaluación de la función Rosenbrock para  $-30 < x, y < 30$ **Table 5.** Statistic Analysis of Rosenbrock's function evaluation results for  $-30 < x, y < 30$ 

Estadística		PSOS	Simplex(Nealder-Mead)	Quasi Newton
Óptimo Global	Mejor Valor	0.000000	0.000000	0.000000
	Media de las muestras	0.000000	1.433539	0.704949
	Varianza	1.9058E-18	9.5115E+00	1.8156E+01
	Desviación Estandar	1.3805E-09	3.0841E+00	4.2610E+00
Coordenada X	Mejor Valor	1.000000	1.000000	1.000000
	Media de las muestras	1.000053	0.528487	0.907564
	Varianza	4.1281E-07	1.1647E+00	6.9306E-01
	Desviación Estandar	6.4250E-04	1.0792E+00	8.3250E-01
Exactitud Coordenada X		100%	53%	91%
Coordenada Y	Mejor Valor	1.000000	1.000000	1.000001
	Media de las muestras	1.000112	1.431097	1.510626
	Varianza	1.6964E-06	2.6242E+00	1.5755E+01
	Desviación Estandar	1.3025E-03	1.6199E+00	3.9693E+00
Exactitud Coordenada Y		100%	57%	49%
Tiempo Medio de Cálculo (s)		13.926	0.022	0.040

Con esta función de prueba queda demostrada la exactitud y precisión de los métodos usados, sin embargo, se nota una diferencia importante en los tiempos de ejecución de los métodos simplex y quasi-Newton con respecto al algoritmo PSOSX. Se evaluó esta misma función en diferentes dominios de las variables independientes y se encontró que la exactitud de los métodos en este caso se mantiene intacta, puesto que la función carece de otros puntos críticos a largo de su recorrido. Esta información no se incluye.

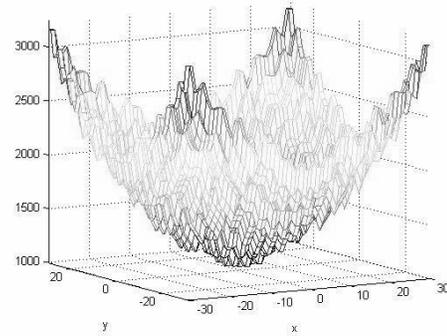
**Segunda función de prueba, la función de Rosenbrock:** Esta función dibuja un valle en curva cuyo fondo desciende con un declive muy suave hacia el mínimo global 0 que alcanza en el punto de coordenadas (1,1). La función de Rosenbrock se define como

$$z(x, y) = 100(y - x^2)^2 + (1 - x)^2. \quad (16)$$

Aquí comienzan a aparecer las fortalezas del algoritmo PSOSX. La más evidente de ellas, en este caso, es su robustez a la hora de tratar el problema en diferentes espacios de búsqueda garantizando el óptimo valor en las coordenadas exactas en todos los problemas, mientras que los otros métodos presentaron resultados deficientes en este aspecto. Por otra parte, el tiempo de ejecución continúa siendo el talón de Aquiles del algoritmo PSOSX. La Tabla 5 muestra algunos de los resultados obtenidos en los experimentos.

Se observa que los métodos numéricos necesitan que su espacio de búsqueda esté restringido a un perímetro muy cercano alrededor del óptimo para obtener resultados confiables. A continuación se mostrará que en funciones con relieve escabroso y numerosos mínimos locales se hace más evidente este problema, lo que permite que el PSOSX ofrezca resultados 100% confiables a cambio del sacrificio en lo que a tiempos de ejecución se refiere.

**Tercera función de prueba, la función de Venter:** Con el objetivo de medir las capacidades de búsqueda del algoritmo PSOSX en problemas con múltiples máximos y mínimos, se propone esta función que posee aproximadamente 300 mínimos locales como se muestra en la Figura 2. El mínimo global 100 lo alcanza el punto de coordenadas (0, 0). El mayor reto que enfrentaran los algoritmos en esta prueba será el de evitar caer prematuramente en un mínimo local. La Tabla 6 muestra los resultados estadísticos para un primer dominio.



**Figura 2.** Gráfica de la función de Venter  
**Figure 2.** Venter's function graph

**Tabla 6.** Análisis estadístico de los resultados de la evaluación de la función de Venter para  $-30 < x, y < 30$   
**Table 6.** Statistic Analysis of Venter's function evaluation results for  $-30 < x, y < 30$

Estadística		PSOS	Simplex(Nealder-Mead)	Quasi Newton
Óptimo Global	Mejor Valor	1000.000000	1014.823148	1000.000000
	Media de las muestras	1000.000000	1721.956461	1678.207918
	Varianza	3.3916E-18	1.8520E+05	1.8687E+05
	Desviación Estandar	1.8416E-09	4.3035E+02	4.3229E+02
Coordenada X	Mejor Valor	0.000000	0.000015	0.000000
	Media de las muestras	0.000000	-1.680441	-0.222193
	Varianza	1.0475E-10	3.1054E+02	2.9576E+02
	Desviación Estandar	1.0235E-05	1.7622E+01	1.7198E+01
Coordenada Y	Mejor Valor	0.000000	0.000005	0.000000
	Media de las muestras	0.000000	-2.804258	0.163032
	Varianza	7.3888E-11	2.9552E+02	2.7085E+02
	Desviación Estandar	8.5958E-06	1.7191E+01	1.6458E+01
Exactitud		100%	28%	32%
Tiempo Medio de Cálculo (s)		8.965	0.019	0.015

Note que a pesar de lo reducida que se encuentra la zona de búsqueda los resultados de los métodos numéricos no ofrecen la completa exactitud que sí puede ofrecer el algoritmo híbrido PSOSX, no obstante se mantiene la ventaja que los primeros tienen sobre el segundo, en lo que a tiempos de ejecución se refiere.

Se observa que no sólo se ve comprometida la exactitud de los métodos numéricos sino también que la precisión de los algoritmos es bastante baja; por otra parte, el algoritmo híbrido PSOSX ofrece resultados 100% confiables y un desempeño robusto, proporcionándole un valor agregado a cada segundo adicional que invierte en realizar la búsqueda.

**Cuarta función de prueba, la función de Levy:** Esta función posee 760 mínimos locales con pendientes abruptas que dificultan la identificación del óptimo global (ver la Figura 3). La función de Levy está definida como

$$z(x, y) = (x + 1, 42513)^2 + (y + 0, 80032)^2 + f_1(x)f_2(y), \quad (17)$$

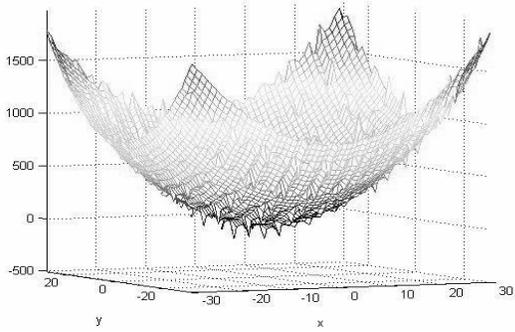
en donde

$$f_1(x) = \sum_{i=1}^5 f_1(x) + i \cos((i - 1)x + i),$$

$$f_2(y) = \sum_{i=1}^5 f_2(y) + i \cos((i - 1)y + i),$$

y alcanza su mínimo global  $-176,1375$  en el punto de coordenadas  $(-1, 3068; -1, 4248)$ .

Se observa un relieve quebrado que predomina en la gráfica de esta función, sin perder de vista la gran cantidad de puntos de inflexión que existe en su recorrido. Nótese las pendientes abruptas que dan forma a sus máximos y mínimos. Nuevamente por brevedad sólo se muestran en la Tabla 7 los resultados de una de las simulaciones realizadas.



**Figura 3.** Gráfica de la función de Levy  
**Figure 3.** Levy's function graph

A pesar de la dificultad que propone esta función, se aprecia la exactitud y precisión impuesta por el algoritmo híbrido PSOSX. En contraste, se observa el bajo rendimiento ofrecido por los métodos simplex y quasi-Newton. Aún si se reduce el espacio de búsqueda, los resultados del algoritmo híbrido PSOSX se encuentran lejos de ser logrados por los otros dos métodos.

**Tabla 7.** Análisis estadístico de los resultados de la evaluación de la función de Levy N°5 para  $-5 < x, y < 5$   
**Table 7.** Statistic Analysis of N°5 Levy's function evaluation results for  $-5 < x, y < 5$

Estadística		PSOS	Simplex(Nealder-Mead)	Quasi Newton
Óptimo Global	Mejor Valor	-176.137578	-176.137578	-176.137578
	Media de las muestras	-176.137578	-15.517691	-19.170472
	Varianza	4.2865E-18	1.8511E+03	1.7561E+03
	Desviación Estandar	2.0704E-09	4.3025E+01	4.1906E+01
Coordenada X	Mejor Valor	-1.306802	-1.306851	-1.306853
	Media de las muestras	-1.306860	0.186640	0.015359
	Varianza	1.7131E-08	7.4442E+00	8.3856E+00
	Desviación Estandar	1.3089E-04	2.7284E+00	2.8958E+00
Coordenada Y	Mejor Valor	-1.424820	-1.424838	-1.424845
	Media de las muestras	-1.424844	-0.039927	-0.064612
	Varianza	2.1487E-10	7.9138E+00	7.6926E+00
	Desviación Estandar	1.4658E-05	2.8132E+00	2.7736E+00
Exactitud		100%	9%	11%
Tiempo Medio de Cálculo (s)		10.914	0.018	0.017

#### 4. CONCLUSIONES

Los resultados obtenidos muestran la exactitud, precisión y robustez que ofrece el algoritmo híbrido PSOSX, especialmente en problemas que por su complejidad son difíciles de resolver mediante los dos métodos numéricos convencionales utilizados. La herramienta computacional desarrollada fue capaz de encontrar el óptimo de las funciones de prueba y sus coordenadas con una efectividad del 100%; ese no fue el caso para el simplex y el quasi-Newton. Por otro lado, para todos los casos estudiados se hace evidente la poca competitividad que ofrece este algoritmo híbrido en lo que a tiempo de ejecución se refiere.

#### REFERENCIAS

- [1] CHAN F., KUMAR M., Swarm Intelligence, I-TECH Education and Publishing, 2007.
- [2] DOWSLAND K. Heuristics Design and Fundamentals of the Simulated Annealing. Nottingham, U.K., AEPIA, 2003.
- [3] MELIÁN, B., MORENO J. Metaheuristics: A Global View, Ed., Santa Cruz de Tenerife, 2003.
- [4] NEALDER, J., MEAD, R., A Simplex method for function minimization, Computer Journal 7, 308- 313,1965.

- [5] KENNEDY J., EBERHART, R. Particle Swarm Optimization. Proc. IEEE Int. Conf. Neural Networks, 39-43, 1995.
- [6] VENTER G., SOBIESKI, J., A parallel particle swarm optimization algorithm accelerated by asynchronous evaluations. 6<sup>th</sup> World congress of structural and multidisciplinary optimization, Rio de Janeiro, Brazil, 2005.
- [7] XU S., RAHMAT Y., Boundary conditions in particle swarm optimization revisited, IEEE Trans. Antennas Propagat., Vol. 55, no.3, 760-765, Mar., 2007.
- [8] NANBO J., Particle Swarm Optimization in Engineering Electromagnetics, (Doctoral Thesis), University of California, Los Angeles, 2008.
- [9] HUANG T., MOHAN A., A hybrid boundary condition for robust particle swarm optimization, IEEE Antennas Wirless Propagat. Lett, Vol. 5, pp 112-117, 2005.
- [10] MIKKI S., KISHK A., Hybrid periodic boundary condition for particle swarm optimization, IEEE Trans. Antennas Propagat., Vol. 55, no. 11, 3252-3256, Nov., 2007.
- [11] CLERC M. Particle Swarm Optimization, Prentice Hall, London, U.K., 2006.
- [12] BEGAMBRE O., Algoritmo Híbrido para Avaliação da Integridade Estrutural: Uma Abordagem Heurística (Tesis doctoral). Escuela de Ingeniería de San Carlos, Universidad de Sao Paulo, 2007.
- [13] FAN S., ZAHARA E. A hybrid simplex search and particle swarm optimization for unconstrained optimization. European J. of Operat. Research, September, 527-548, 2007.
- [14] FAN S., LIANG Y., ZAHARA E. Hybrid simplex search and PSO for the global optimization of multimodal functions, J. of Eng. Optimization, 36(4), 401-418, 2004.