Hibridación de Evolución Diferencial utilizando Hill Climbing para resolver problemas de optimización con restricciones

Sebastián Hernández[1], Guillermo Leguizamón[2], and Efrén Mezura-Montes[3]

[1] Dpto.de Ciencias Exactas y Naturales, Unidad Académica Río Gallegos, Universidad Nacional de la Patagonia Austral
[2] Laboratorio de Investigación y Desarrollo en Inteligencia Computacional, Universidad Nacional de San Luis
[3] Laboratorio Nacional de Informática Avanzada, Xalapa, Veracruz, México

Abstract. La Evolución Diferencial es una metaheurística de mucha utilidad cuando se requiere optimizar numéricamente funciones o problemas multidimensionales que no pueden ser resueltos por algún método tradicional de optimización global. A su vez, si se agregan condiciones de frontera, será necesario utilizar alguna técnica de manejo de restricciones. Para mejorar el desempeño de ED, una posible alternativa es combinarlo con algún algoritmo de búsqueda local. En este trabajo se presenta la hibridación de Evolución Diferencial y *Hill Climbing*, obteniendo resultados de calidad similar o superior a los conseguidos por métodos ya testeados.

Keywords: evolución diferencial, hill climbing, hibridación, optimización restringida.

1 Introducción

Optimización es un área que comenzó a crecer décadas atrás, en paralelo con el auge del procesamiento computacional. Constantemente se desarrollan nuevos algoritmos y técnicas ya que, de acuerdo al avance de disciplinas como biología, química o física, se requiere optimizar problemas con modelos cada vez más reales y por ende más complejos.

Evolución Diferencial (ED)[5] es una de las técnicas metaheurísticas más conocidas y utilizadas. Ésto es por su sencillez de implementación y la calidad de resultados que obtiene. Es considerada una técnica de optimización global porque explora, de acuerdo a parámetros internos, un espacio de búsqueda por más amplio que éste sea. También existen algoritmos de búsqueda local, que a través de operaciones simples, buscan puntos óptimos sobre pequeñas regiones de puntos elegidos al azar. *HilClimbing* (HC) [4] es uno de ellos.

En este trabajo se realiza la hibridación de ED y HC, utilizando los mejores individuos de ED para aplicar HC. La implementación de HC no es la clásica, sino que se opera con más de una dimensión por vez, determinado ésto por

un vector aleatorio. A su vez, se comparan los resultados obtenidos con los del algoritmo ganador de CEC 2010: ϵDEag [6].

La organización del trabajo es la siguiente: (2) Evolución Diferencial; (3) *Hill Climbing*; (4) Algoritmo híbrido ED+HC; (5) Experimentos, (6) Conclusiones y (7) Agradecimientos.

2 Evolución Diferencial

La ED es una técnica metaheurística basada en poblaciones de vectores numéricos. El proceso seguido por ED para resolver un problema se caracteriza por iterar sobre una población de vectores para hacer evolucionar soluciones candidatas con respecto a una función de aptitud llamada *fitness*. Se siguen claramente cuatro etapas: Inicialización; Mutación; Cruzamiento y Selección.

- **Inicialización**. Antes de que la población de vectores sea inicializada, se debe definir un límite superior e inferior para cada una de los variables a utilizar, b_U y b_L respectivamente. Una vez determinados, un generador de números aleatorios crea, para cada variable del vector, un valor numérico entre b_U y b_L. Por ejemplo, el valor inicial, $g = 0$, de la j-ésima variable de i-ésimo vector es

$$x_{j,i,0} = rand_j\,[0,1) \cdot (b_{j,U} - b_{j,L}) + b_{j,L}.$$

  El generador de números aleatorios se ejecuta una vez para cada valor de variable, lo que asegura una población completamente aleatoria. Sin importar que una variable sea discreta o continua, debe ser inicializada con un valor continuo, puesto que ED trabaja con números decimales de punto flotante. Cada uno de esos vectores generados es identificado con un número desde 0 hasta N_{p-1}.
- **Mutación**. Como otros métodos basados en poblaciones, ED genera nuevos puntos que no son otra cosa que perturbaciones de puntos existentes. ED perturba, uno a uno, todos los puntos de la población sumándole la diferencia escalada de dos puntos diferentes ($\mathbf{x}_{r1}$ y $\mathbf{x}_{r2}$) de la misma población escogidos al azar. Para el vector $\mathbf{x}_i$ se genera el vector $\mathbf{v}_i$ (llamado vector de mutación o vector de ruido) por medio de la siguiente definición

$$\mathbf{v}_{i,g} = \mathbf{x}_{r0,g} + F \cdot (\mathbf{x}_{r1,g} - \mathbf{x}_{r2,g})\,,$$

  donde F es el valor de escala utilizado, generalmente un valor definido por el usuario entre 0 y 1. Éste es el proceso denominado *mutación diferencial*.
- **Cruzamiento**. A fin de agregarle diversidad genética a la población, se utiliza otro proceso evolutivo denominado *cruzamiento uniforme*. También identificado como *recombinación discreta*, el cruzamiento genera vectores de prueba utilizando información que copia de dos diferentes vectores. En particular, ED cruza cada vector (llamado *trial* al momento de cruzarse) con el vector de mutación dependiendo de la probabilidad de cruzamiento de la

población (C_r). $C_r \in [0,1]$ es un valor definido por el usuario que controla qué valores de variables serán copiados del vector de mutación al vector hijo u_i:

$$\mathbf{u}_{i,g} = u_{j,i,g} = \begin{cases} v_{j,i,g} \text{ si } rand_j(0,1) \leq C_r \\ x_{j,i,g} \text{ en otro caso.} \end{cases}$$

– **Selección**. Si el vector hijo u_i generado luego del cruzamiento tiene un valor de ajuste mejor que el vector original, lo reemplazará en la población. De otra forma, el vector padre o *target* x permanecerá en la población por, al menos, una generación más. En nuestro caso, como queremos obtener un cero (o un mínimo en valor absoluto), la función de selección es

$$\mathbf{x}_{i,g+1} = \begin{cases} \mathbf{u}_{i,g} \text{ si } f(\mathbf{u}_{i,g}) < f(\mathbf{x}_{i,g}) \\ \mathbf{x}_{i,g} \text{ en otro caso.} \end{cases}$$

El esquema general planteado anteriormente corresponde a la versión *DE/rand/1/bin* debido a que el vector base para la mutación es elegido al azar, 1 diferencia de vectores de adicionada a él y porque el número de parámetros donados por el vector mutante sigue de cerca una distribución binomial.

2.1 Restricciones

La ED fue diseñada para optimizar problemas numéricos, donde la principal característica es que se posee una forma de calcular la calidad para cualquier solución potencial, llamada *fitness*. Sin embargo, como la mayoría de los problemas en los que se aplica ED son basados en modelos reales, existen ciertas restricciones sobre las variables que no están expresadas en la función de *fitness*. Es decir que además del lograr un *fitness* óptimo, las variables deben estar en una región factible del problema. Esas condiciones de factibilidad son las que impiden la aplicación directa de ED, ya que no fue creada originalmente para operar con restricciones sobre las variables. Sin embargo, las técnicas de manejo de restricciones son la clave para utilizar ED en este tipo de problemas complejos [3].

El objetivo de la optimización es encontrar el $\mathbf{x}$ óptimo, que minimice la función objetivo satisfaciendo simultáneamente las restricciones impuestas. Dos tipos de funciones, g y h representan las restricciones sobre las variables:

– **Regiones**. Son restricciones del tipo $g_i(\mathbf{x}) \leq 0$, donde es importante medir cuán lejos está un individuo de la región de factibilidad. En caso de obtener un valor positivo, puede ser utilizado como medida de violación de la restricción.
– **Fronteras**. Son restricciones del tipo $h_i(\mathbf{x}) = 0$. A estas restricciones se las puede transformar en restricciones de región, eligiendo un valor ϵ pequeño y reescribiéndola como $|h_i(\mathbf{x})| - \epsilon \leq 0$. También se la puede utilizar como restricción de frontera de acuerdo al método de manejo utilizado.

Una de las técnicas más directas y simples para manejo de restricciones es la de **Penalidad Estática** [3], [1]. Un mínimo factible debe satisfacer todas las

restricciones solicitadas, además de minimizar el valor objetivo. La función de penalidad estática, exterior al *fitness*, convierte las violaciones de las restricciones en un valor que se suma al obtenido en el *fitness*, creando de esta forma una función de pseudo-objetivo, $\phi(\mathbf{x})$:

$$\phi(\mathbf{x}) = f(\mathbf{x}) + P(\mathbf{x}),$$

En este trabajo se utilizó esta técnica de manejo de restricciones y la función de penalidad utilizada $P(\mathbf{x})$ es la siguiente:

$$P(\mathbf{x}) = \sum_{i=1}^{n} C_p \cdot \max\{0, g_i(\mathbf{x})\},$$

donde C_p es una constante a elección del usuario. Se debe tener en cuenta que, si C_p es pequeña, el peso de $P(\mathbf{x})$ será poco en $\phi(\mathbf{x})$ pero permitirá una amplia exploración del espacio de soluciones y una convergencia lenta. En caso contrario, considerando un valor de C_p grande, se puede lograr una convergencia prematura al castigar excesivamente a las soluciones no factibles.

3 *Hill Climbing*

El método de *Hill Climbing* (HC) es un bucle que continuamente se mueve en dirección del valor creciente de una función [4]. El proceso se inicia seleccionando al azar un individuo de la población. Dicho individuo, denominado *padre* es mutado, generando un *hijo*. De estos dos individuos se escoge aquel que tenga mejor aptitud, es decir el mayor valor de *fitness* para permanecer en la población. Después de cada selección, el algoritmo comprueba la edad del individuo seleccionado. Cuando un padre es seleccionado dos veces seguidas, su edad aumenta. Cada vez que un nuevo individuo se ha seleccionado, su edad se heredada de su padre. Cada mutación se realiza mediante la adición de un escalar a una coordenada del individuo. Si el espacio de búsqueda involucra restricciones, se debe tener cuidado para no sobrepasar los límites impuestos. En un espacio n-dimensional, se aplican $2n$ mutaciones, una para cada dirección posible en el espacio. La magnitud de las mutaciones disminuye con la edad. Esto asegura la adaptación del grano de búsqueda para la estructura local del espacio [7].

Si bien es un algoritmo simple de implementar y muy útil en espacios de una dimensión, se observan ciertos problemas en la ejecución en espacios de dimensión superior. Realizando una analogía con un problema real de escalado de una montaña, se pueden dar los siguientes casos:

- **Máximos locales**. Como es un algoritmo de búsqueda local y no exhaustiva, no se asegura que converja al mejor valor posible. Esto puede ocurrir sólo si se inicia la búsqueda sobre una región cercana al máximo global.
- **Mesetas**. El algoritmo puede explorar localmente una superficie plana (en el sentido de mismos valores de *fitness*). En este caso, no conseguirá mejoras respecto al punto inicial. Tanto el padre como el hijo tienen el mismo valor de *fitness*.

– **Crestas**. De todas las direcciones posibles, sólo una es la que mejora el *fitness*. Siguiendo cualquier camino por el resto de las direcciones, se consiguen valores menores. Si la exploración no se realiza sobre esa dirección especial, no se consigue un *fitness* mejor.

4 Algoritmo híbrido ED+HC

Para lograr la versión híbirida de ED con HC se realizaron cambios en el bucle de HC descrito previamente. Los cambios introducidos son:

– El individuo seleccionado para mutar (padre) no es escogido al azar, sino que es el mejor individuo de la población de ED. Si el *fitness* del hijo es menor[4] que el de su padre, éste quedará en la población, eliminando al padre.
– No se conserva información con respecto a la edad del individuo seleccionado.
– Para elegir las variables a mutar, se genera un vector aleatorio de valores de 1 a D, donde D es la dimensión del problema a optimizar. El primer valor del vector antes mencionado, v_1, indica la cantidad de variables a mutar y las posiciones de las variables a mutar están determinadas por los primeros v_1 elementos del vector. Por ejemplo, si se intenta optimizar una función de dimensión 10, un posible vector aleatorio es

$$v = [5, 4, 10, 1, 8, 6, 9, 3, 2, 7],$$

donde se indica que cinco variables serán mutadas y la primera que muta (mutan todas en simultáneo) es x_5. Los números 6, 9, 3, 2 y 7 corresponden a variables que no mutan por no estar dentro de las primeras cinco, siendo las variables x_5, x_4, x_{10}, x_1 y x_8 las variables mutadas. Si el vector fuera

$$v = [6, 3, 7, 8, 5, 1, 2, 4, 9, 10],$$

mutan las variables correspondientes a las primeras 6 posiciones. Es decir que mutan las variables x_6, x_3, x_7, x_8, x_5 y x_1. Las demás variables (x_2, x_4, x_9 y x_{10}) no se ven involucradas en la exploración local.
– Las variables seleccionadas se mutan de acuerdo a la siguiente fórmula:

$$H_j = x_j + (-1)^{v_j} * k * inc,$$

donde H representa al hijo, x es el padre, k es un factor de escala entero ($k_0 = 1$) para la mutación e inc es el valor real ($rand/100$) con el que será mutada la variable. Este proceso se ejecuta una cantidad determinada de veces (parámetro denominado hc). Si luego de la mutación, el *fitness* de H no difiere del *fitness* de x en menos que 0.0001 en valor absoluto (parámetro modificable del algoritmo), se aumenta el factor de escala k duplicando su valor, las veces que sea necesario. Para el caso en que el individuo seleccionado se encuentre en una supuesta meseta, es decir que su valor de *fitness* no cambie considerablemente, se estable un máximo para k de 128.

[4] problema de minimización

5 Experimentos

Para evaluar el desempeño de ED+HC con respecto a ED se utilizaron las primeras doce funciones de dimensión 30 definidas en el *Single Objective Constrained Real - Parameter Optimization* del CEC2010 [2]. Para establecer un punto de comparación general, se tomaron las soluciones obtenidas por los ganadores del concurso, Takahama y Sakai [6].

Los parámetros de ejecución de ED y ED+HC son

$$ED + HC(f, popsize, t_{max}, Cr, F, dim, hc),$$

donde

- f es el número de función a optimizar, variando entre 1 y 12;
- $popsize$ es el tamaño de la población. Cuando se ejecutó ED se utilizaron 85 individuos mientras que para ED+HC se utilizaron 55;
- t_{max} muestra el tiempo de evolución, en ambos casos se utiliza $t_{max} = 7000$. Ya sea en ED o en ED+HC la cantidad de evaluaciones de función (FEs) no supera los 6×10^5 para poder comparar con los resultados de Takahama y Sakai;
- Cr es la probabilidad de cruzamiento y F es el factor de escala de ED. En ambos se utilizó 0.6 a fin de asegurar un balance entre exploración y explotación;
- dim indica la dimensión de las funciones utilizadas. En este caso, $dim = 30$;
- hc indica si se efectuará *Hill Climbing* o no. Para ED, $hc = 0$ y para ED+HC, $hc = 30$. De esta forma, $FEs = (popsize + hc)t_{max} = 5.95 \times 10^5$

Las restricciones de frontera fueron transformadas a restricciones de región, a fin de utilizar un único coeficiente de penalidad, $C_p = 150$. El valor de ϵ utilizado fue de 0.0001.

Las tablas 1 y 2 muestran los resultados obtenidos en los experimentos. Para cada función, con los algoritmos ED, ED+HD y ϵDEag, se realizaron 25 ejecuciones. La fila *Mejor* muestra el *fitness* del mejor individuo factible; *Mediana*, *Peor*, *Promedio* y *Desviación* son los datos estadísticos de la población final; *Violaciones* muestra la cantidad de violaciones cometidas con respecto a las restricciones sobre el individuo cuyo *fitness* es la mediana y $\overline{v}$ es la violación promedio cometida sobre la solución mediana y se calcula de la siguiente forma:

$$v = \frac{\sum_{i=1}^{p} G_i(X) + \sum_{j=p+1}^{m} H_j(X)}{m},$$

donde

$$G_i(X) = \begin{cases} g_i(X), \text{ si } g_i(X) > 0 \\ 0, \text{ si } g_i(X) \leqslant 0 \end{cases}$$

y

$$H_j(X) = \begin{cases} |h_j(X)|, \text{ si } |h_j(X)| - \varepsilon > 0 \\ 0, \text{ si } |h_j(X)| - \varepsilon \leqslant 0 \end{cases}.$$

Los resultado obtenidos por esta propuesta son promisorios. Se logran valores de calidad superior con respecto a los obtenidos por Takahama y Sakai[6] en las funciones F4 a F12, mostrando la potencia del algoritmo híbrido. En las tablas 1 y 2 se muestran en negrita los mejores valores de Mediana. En este caso el mejor desempeño lo obtiene ED+HC, seguido de cerca por ED y ϵDEag, lo que sugiere que los tres algoritmos implementados consiguen buenos resultados en general. En la comparación directa ED y ED+HC se nota la mejora conseguida por hibridación, ya que si bien ED consigue resultados buenos, con ED+HC los valores de *fitness* disminuyen aún más.

En las figuras 1 y 2 se muestra la evolución del mejor individuo de ED y ED+HC a través de las 7000 iteraciones de las funciones F01, F02, F03 y F04. Del resto de las funciones se omitió el gráfico pues es similar a alguno de los ya mostrados. Las escalas son logarítmicas en ambos ejes. De estos gráficos se concluye que la convergencia es similar en los algoritmos mostrados. No se observa convergencia prematura sino que la curva de evolución sigue un descenso constante, salvo cerca de t_{max} donde los valores tienden a estabilizarse.

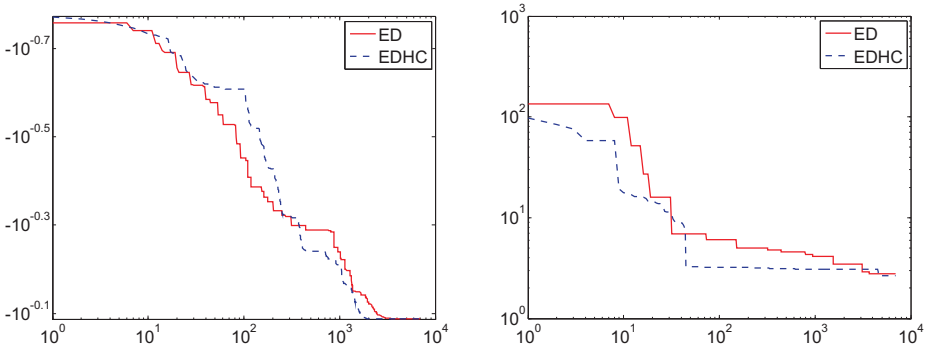

Fig. 1. Evolución del mejor individuo de la población para F01 y F02, respectivamente.

6 Conclusiones

Se presentó una versión híbrida entre ED y HC, a través de una implementación alternativa a la clásica de HC. La hibridación se logró aplicando HC a los mejores individuos obtenidos por ED. Este algoritmo fue ejecutado para optimizar doce funciones escogidas para el concurso de la sesión especial de optimización del CEC2010 y los resultados obtenidos fueron comparados con los que presentan los ganadores del concurso: Takahama y Sakai.

Para un trabajo futuro se considerará el *benchmark* completo del CEC 2010 y se incrementará la dimensión para estudio de escalabilidad, además de implementar otras técnicas de búsqueda local.

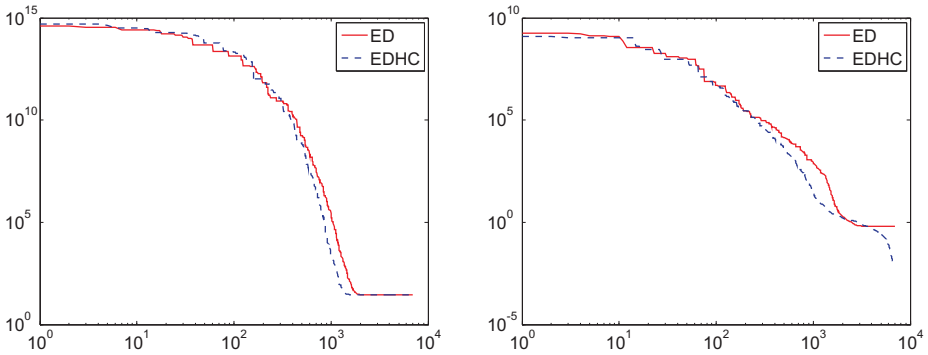

Fig. 2. Evolución del mejor individuo de la población para F03 y F04, respectivamente.

7 Agradecimientos

El segundo autor agradece el apoyo de MinCyT a través del proyecto bilateral México-Argentina MX1103 y del proyecto P38403 de la UNSL. El tercer autor agradece el apoyo de CONACyT a través del proyecto bilateral México-Argentina No. 164626 y del proyecto 79809.

References

1. Carlos A. Coello Coello. Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art, 2002.
2. Rammohan Mallipeddi and Ponnuthurai Nagaratnam Suganthan. Problem definitions and evaluation criteria for the cec 2010 competition on constrained real-parameter optimization. Technical report, Nanyang Technological University, Singapore, 2010.
3. Efrén Mezura-Montes and Carlos A. Coello Coello. Constraint-handling in nature-inspired numerical optimization: Past, present and future. *Swarm and Evolutionary Computation*, 1(4):173–194, 2011.
4. Stuart Rusell and Peter Norvig. *Inteligencia Artificial (2da Edición) - Un Enfoque Moderno*. Pearson - Prentice Hall, 2004.
5. Rainer Storn and Kenneth Price. Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11:341–359, 1997.
6. Tetsuyuki Takahama and Setsuko Sakai. Constrained optimization by the ϵconstrained differential evolution with an archive and gradient-based mutation. *IEEE World Congress on Computational Intelligence*, I:1680–1688, 2010.
7. Deniz Yuret and Michael de la Maza. Dynamic hill climbing: Overcoming the limitations of optimization techniques. Technical report, Numinous Noetics Group, Artificial Intelligence Laboratory, MIT, 1993.

	F01			**F02**		
	ED	**ED+HC**	**ϵDEag**	**ED**	**ED+HC**	**ϵDEag**
Mejor	-8,199328E-01	-8,199345E-01	-8,218255E-01	-1,729535E+00	-1,961395E+00	-2,169248E+00
Mediana	-8,127212E-01	-8,199343E-01	**-8,206172E-01**	-1,414513E+00	-1,752225E+00	**-2,152145E+00**
Peor	-7,725882E-01	-8,127290E-01	-8,195466E-01	-1,139107E+00	-1,080479E+00	-2,117096E+00
Promedio	-8,108757E-01	-8,194010E-01	-8,208687E-01	-1,397607E+00	-1,683329E+00	-2,151424E+00
Desviación	1,139819E-02	1,663286E-03	7,103893E-04	1,528931E-01	2,786315E-01	1,197582E-02
Violaciones	0	0	0	1	0	0
$\overline{v}$	0,000000E+00	0,000000E+00	0,000000E+00	2,368519E-03	0,000000E+00	0,000000E+00
	F03			**F04**		
	ED	**ED+HC**	**ϵDEag**	**ED**	**ED+HC**	**ϵDEag**
Mejor	2,867347E+01	2,867347E+01	2,867347E+01	1,505793E-03	3,528574E-04	4,698111E-03
Mediana	**2,867347E+01**	**2,867347E+01**	**2,867347E+01**	3,196393E-03	**1,641278E-03**	6,947614E-03
Peor	2,867347E+01	2,867347E+01	3,278014E+01	8,019282E-03	3,836657E-03	1,777889E-02
Promedio	2,867347E+01	2,867347E+01	2,883785E+01	4,106104E-03	1,529283E-03	8,162973E-03
Desviación	4,683177E-09	1,347989E-06	8,047159E-01	2,002642E-03	8,342196E-04	3,067785E-03
Violaciones	0	0	0	0	0	0
$\overline{v}$	0,000000E+00	0,000000E+00	0,000000E+00	0,000000E+00	0,000000E+00	0,000000E+00
	F05			**F06**		
	ED	**ED+HC**	**ϵDEag**	**ED**	**ED+HC**	**ϵDEag**
Mejor	-4,833088E+02	-4,771455E+02	-4,531307E+02	-5,275624E+02	-5,304233E+02	-5,285750E+02
Mediana	**-4,815287E+02**	-4,746565E+02	-4,500404E+02	-5,253639E+02	**-5,296760E+02**	-5,280407E+02
Peor	-4,798812E+02	-4,665849E+02	-4,421590E+02	-5,171673E+02	-5,275613E+02	-5,264539E+02
Promedio	-4,815579E+02	-4,744888E+02	-4,495460E+02	-5,242637E+02	-5,294831E+02	-5,279068E+02
Desviación	1,022818E+00	3,132289E+00	2,899105E+00	3,095089E+00	6,397257E-01	4,748378E-01
Violaciones	2	0	0	0	2	0
$\overline{v}$	3,962481E-03	0,000000E+00	0,000000E+00	0,000000E+00	6,271008E-04	0,000000E+00

Table 1. Resultados de los experimentos con la función F01 a F06.

	F07			**F08**		
	ED	**ED+HC**	**ϵDEag**	**ED**	**ED+HC**	**ϵDEag**
Mejor	2,665074E-23	2,470121E-29	1,147112E-15	1,404690E-23	2,470121E-29	2,518693E-14
Mediana	4,872507E-22	**8,212328E-26**	2,114429E-15	8,151972E-22	**6,092315E-26**	6,511508E-14
Peor	5,596772E-21	1,261938E-24	5,481915E-15	9,180257E+01	9,180261E+01	2,578112E-13
Promedio	9,119600E-22	1,853336E-25	2,603632E-15	7,069181E+00	3,672104E+00	7,831464E-14
Desviación	1,249715E-21	2,904451E-25	1,233430E-15	2,448721E+01	1,836052E+01	4,855177E-14
Violaciones	0	0	0	0	0	0
$\overline{v}$	0,000000E+00	0,000000E+00	0,000000E+00	0,000000E+00	0,000000E+00	0,000000E+00
	F09			**F10**		
	ED	**ED+HC**	**ϵDEag**	**ED**	**ED+HC**	**ϵDEag**
Mejor	4,662842E-15	2,370624E-20	2,770665E-16	3,130917E+01	3,130908E+01	3,252002E+01
Mediana	6,947647E+01	6,920619E+01	**1,124608E-08**	3,131222E+01	**3,130909E+01**	3,328903E+01
Peor	1,164175E+02	1,617975E+02	1,052759E+02	2,255249E+02	2,650643E+02	3,463243E+01
Promedio	5,783478E+01	6,193226E+01	1,072140E+01	3,908073E+01	4,802656E+01	3,326175E+01
Desviación	3,090653E+01	4,094189E+01	2,821923E+01	3,884254E+01	5,830148E+01	4,545577E-01
Violaciones	0	0	0	0	0	0
$\overline{v}$	0,000000E+00	0,000000E+00	0,000000E+00	0,000000E+00	0,000000E+00	0,000000E+00
	F11			**F12**		
	ED	**ED+HC**	**ϵDEag**	**ED**	**ED+HC**	**ϵDEag**
Mejor	-3,923431E-04	-3,923384E-04	-3,268462E-04	-1,992635E-01	-1,992635E-01	-1,991453E-01
Mediana	**-3,923406E-04**	-3,922639E-04	-2,843296E-04	**-1,992635E-01**	-1,992634E-01	5,337125E+02
Peor	-3,923335E-04	-3,917380E-04	-2,236338E-04	1,181452E+02	1,181352E+02	5,461723E+02
Promedio	-3,923403E-04	-3,922238E-04	-2,863882E-04	-1,621667E+01	-1,460144E+01	3,562330E
Desviación	2,357245E-09	1,408836E-07	2,707605e-05	1,096712E+02	1,010505E+02	2,889253E+02
Violaciones	0	0	0	0	0	1
$\overline{v}$	0,000000E+00	0,000000E+00	0,000000E+00	0,000000E+00	0,000000E+00	3,240709E-01

Table 2. Resultados de los experimentos con la función F07 a F12.