

Evolución diferencial con memoria de parámetros para la optimización de mecanismos de cuatro barras

Maury-Faney Zapata-Zapata¹, Efrén Mezura-Montes²,
Edgar-Alfredo Portilla-Flores³

¹ Laboratorio Nacional de Informática Avanzada (LANIA A. C.), Xalapa, Veracruz, México

² Universidad Veracruzana, Xalapa, Veracruz, México

³ Instituto Politécnico Nacional, CIDETEC, Ciudad de México, México

mzapata.mca15@lania.edu.mx, emezura@uv.mx, aportilla@ipn.mx

Resumen. L-SHADE es un algoritmo basado en evolución diferencial que adapta automáticamente los parámetros relacionados con la mutación (F) y la cruce (CR) a través de una memoria de parámetros, y además controla el tamaño de la población (NP) usando una función lineal. En este trabajo se incorpora una estrategia de manejo de restricciones a L-SHADE con el fin de optimizar tres casos de estudio de un mecanismo de cuatro barras y se llama C-LSHADE a este nuevo enfoque. En la experimentación, se compara el desempeño de C-LSHADE contra un algoritmo DE/rand/1/bin del estado del arte que utiliza un enfoque de parámetros aleatorios para resolver problemas mecánicos de optimización. Los resultados obtenidos demuestran que C-LSHADE es capaz de llegar a mejores o iguales soluciones que DE/rand/1/bin, empleando un porcentaje considerablemente menor de evaluaciones de la función objetivo y liberando al usuario de la tarea de ajustar los parámetros F , CR y NP .

Palabras clave: Optimización, evolución diferencial, control de parámetros, mecanismo de cuatro barras, L-SHADE.

Differential Evolution with Parameter Memory to Optimize Four-Bar Mechanisms

Abstract. L-SHADE is an algorithm based on differential evolution which automatically adapts the parameters related to mutation (F) and crossover (CR) using a parameter memory. L-SHADE also controls the population size (NP) with a linear function. In this work, a constraint-handling technique is added to L-SHADE to optimize three four-bar

mechanisms and this new approach is called C-LSHADE. During the experimentation, the performance of C-LSHADE is compared against a state-of-the-art DE/rand/1/bin algorithm that uses a random parameter approach to solve mechanical optimization problems. The obtained results suggest that C-LSHADE is able to find better or equal solutions than DE/rand/1/bin, using a considerably lower percentage of evaluations of the objective function while keeping the user from adjusting the parameters F , CR y NP .

Keywords: Optimization, differential evolution, parameter control, four-bar mechanism, L-SHADE.

1. Introducción

Los algoritmos evolutivos (AEs) se han convertido en una técnica popular para resolver problemas complejos de optimización debido a su facilidad de uso y robustez. El éxito de dichas técnicas puede ser encontrado en la literatura especializada [3,13]. No obstante, a pesar del auge que han tenido, uno de los problemas importantes que influye en la obtención de buenos resultados cuando se aplica un AE, es la configuración de sus parámetros [4].

De acuerdo con [4], existen dos formas de abordar la problemática mencionada: (1) el ajuste de parámetros y (2) el control de parámetros. El ajuste de parámetros es la estrategia que se practica comúnmente, que consiste en hallar buenos valores para los parámetros antes de ejecutar el algoritmo y después correr el algoritmo con los valores seleccionados, los cuales se mantienen constantes durante todo el proceso. Por otra parte, en el control de parámetros, los valores se modifican durante la ejecución del algoritmo de tal forma que se promueva una búsqueda exitosa de la solución.

Ambas formas de configuración son válidas, sin embargo, el ajuste de parámetros puede ser inapropiado debido a los siguientes inconvenientes [4]:

- Los parámetros no son independientes, lo que hace prácticamente imposible probar todas las combinaciones diferentes.
- El proceso de ajustar los parámetros requiere mucho tiempo.
- Para un problema dado, los valores de parámetros seleccionados no necesariamente son los óptimos, incluso si el esfuerzo realizado en la configuración fue significativo.

Como solución a los inconvenientes mencionados, se pueden aplicar mecanismos de control que sean lo suficientemente útiles para realizar un buen proceso de configuración de parámetros y liberar a los usuarios de dicha tarea. Sin embargo, desarrollar una técnica de control que adapte los parámetros de forma automática implica un proceso de diseño y por tal motivo, la investigación realizada en este tema se encuentra en su etapa inicial [4]. Es por esto que los investigadores y usuarios de los AEs, aún prefieren realizar un ajuste de parámetros con base en una experimentación previa.

De acuerdo a [5], los estudios en control de parámetros no han sido enfocados a dominios específicos como lo es el de problemas de diseño mecánico. Como ejemplos de este dominio en particular se pueden mencionar los trabajos presentados en [12] y [16]. En ambos casos se realiza un ajuste de parámetros; en el primero, no se explica claramente cómo se llegó a los valores seleccionados y en el segundo se toman valores recomendados en la literatura. También se han propuesto enfoques como los presentados en [1] y [7], en donde los valores de los parámetros relacionados con la cruce y la mutación son generados aleatoriamente en cada ciclo del algoritmo, que si bien no se establecen valores constantes, tampoco se tiene un control sobre el progreso de la búsqueda. Finalmente, en [5], se propone un algoritmo que incorpora una técnica de control con comportamiento senoidal, pero ésta sólo es aplicada al parámetro relacionado con la mutación.

Derivado de lo anterior, surge la motivación de este trabajo, en donde L-SHADE, un algoritmo que incorpora mecanismos de control en los parámetros relacionados con la mutación, cruce y población, y que originalmente fue diseñado para resolver problemas de optimización sin restricciones, es adaptado para solucionar problemas restringidos de diseño mecánico. El objetivo de esta investigación, es promover el uso de dicho algoritmo en la solución de problemas relacionados con mecanismos de cuatro barras, evitando así, que se realice un ajuste manual de parámetros que posiblemente no sea el más adecuado.

El resto del artículo se encuentra organizado de la siguiente manera: en la Sección 2 se describen los problemas de optimización correspondientes a tres casos de estudio de un mecanismo de cuatro barras. Posteriormente, en la Sección 3, se muestra el algoritmo basado en evolución diferencial empleado para este trabajo, enfatizando las modificaciones realizadas a la versión original para resolver problemas con restricciones. En la Sección 4, se presentan y discuten los resultados obtenidos y en la Sección 5 se exponen las conclusiones derivadas de esta investigación.

2. Declaración de problemas de optimización

En la Figura 1 se muestra un mecanismo de cuatro barras conformado por: barra de referencia (\bar{r}_1), barra de entrada (\bar{r}_2), acoplador (\bar{r}_3) y barra de salida o balancín (\bar{r}_4). Para efectos del análisis, se utilizan dos sistemas de coordenadas; el primero, se encuentra fijo al mundo real (OXY) y el segundo es para referencia ($O_2X_rY_r$), en donde (x_0, y_0) , es la distancia entre los puntos de origen de ambos sistemas, θ_0 es el ángulo de rotación del segundo sistema con respecto del primero, θ_i ($i = 1, 2, 3, 4$) denota los ángulos de las cuatro barras, y el par de coordenadas (r_{cx}, r_{cy}) definen la posición del acoplador C [2].

La síntesis del mecanismo de cuatro barras se lleva a cabo para calcular la longitud de sus barras, el ángulo de rotación con respecto al sistema OXY , la distancia entre los dos sistemas de coordenadas y el conjunto de ángulos para la barra de entrada que permitirán generar la trayectoria deseada.

El problema de optimización consiste en minimizar la distancia entre el punto de contacto y el punto de precisión predefinido. La función propuesta para dicho

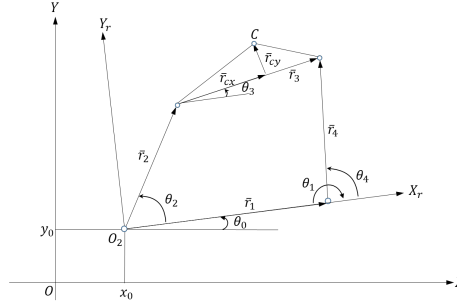


Fig. 1: Mecanismo de cuatro barras. Adaptada de [2].

cálculo es:

$$f(\theta_2^i) = \sum_{i=1}^N [(C_{xd}^i - C_x^i)^2 - (C_{yd}^i - C_y^i)^2], \quad (1)$$

en donde C_x^i y C_y^i son las coordenadas del punto C , mientras que C_{xd}^i y C_{yd}^i son las coordenadas del punto de precisión que define la trayectoria deseada. Más detalles del análisis del mecanismo mostrado, pueden ser encontrados en [2,6].

En este trabajo se obtuvo la síntesis dimensional de tres casos de estudio distintos, en donde el punto C del mecanismo debe seguir una trayectoria de puntos cartesianos previamente definidos. A continuación, se describe cada caso de estudio.

2.1. Caso de estudio 1: seguimiento de una trayectoria lineal vertical, sin sincronización previa - CE1

En el primer caso de estudio, tomado de [6], el acoplador debe pasar por los puntos de precisión $\Omega = \{(20, 20), (20, 25), (20, 30), (20, 35), (20, 40), (20, 45)\}$, los cuales se encuentran alineados verticalmente. El vector de variables de diseño es: $\mathbf{p} = [r_1, r_2, r_3, r_4, r_{cx}, r_{cy}, \theta_0, x_0, y_0, \theta_2^1, \theta_2^2, \theta_2^3, \theta_2^4, \theta_2^5, \theta_2^6]^T$, en donde las primeras cuatro variables corresponden a las barras del mecanismo, las dos siguientes definen la posición del acoplador, θ_0 , x_0 y y_0 indican la orientación del sistema $O_2X_rY_r$ con respecto al sistema OXY , y las variables restantes corresponden a la secuencia de ángulos de la barra de entrada (\bar{r}_2).

Los límites superior e inferior de las variables de diseño se encuentran definidos por: $r_1, r_2, r_3, r_4 \in [0, 60]$, $r_{cx}, r_{cy}, x_0, y_0 \in [-60, 60]$ y $\theta_0, \theta_2^1, \theta_2^2, \theta_2^3, \theta_2^4, \theta_2^5, \theta_2^6 \in [0, 2\pi]$

Se considera el problema de optimización mono-objetivo descrito de la Ecuación 2 a la 11:

$$\min f(\mathbf{p}) = \sum_{i=1}^N [(C_{xd}^i - C_x^i)^2 - (C_{yd}^i - C_y^i)^2], \quad \mathbf{p} \in \mathbb{R}^{15}. \quad (2)$$

Sujeto a:

$$g_1(\mathbf{p}) = r_1 + r_2 - r_3 - r_4 \leq 0, \quad (3)$$

$$g_2(\mathbf{p}) = r_2 - r_3 \leq 0, \quad (4)$$

$$g_3(\mathbf{p}) = r_3 - r_4 \leq 0, \quad (5)$$

$$g_4(\mathbf{p}) = r_4 - r_1 \leq 0, \quad (6)$$

$$g_5(\mathbf{p}) = \theta_2^1 - \theta_2^2 \leq 0, \quad (7)$$

$$g_6(\mathbf{p}) = \theta_2^2 - \theta_2^3 \leq 0, \quad (8)$$

$$g_7(\mathbf{p}) = \theta_2^3 - \theta_2^4 \leq 0, \quad (9)$$

$$g_8(\mathbf{p}) = \theta_2^4 - \theta_2^5 \leq 0, \quad (10)$$

$$g_9(\mathbf{p}) = \theta_2^5 - \theta_2^6 \leq 0. \quad (11)$$

2.2. Caso de estudio 2: seguimiento de una trayectoria no alineada, con sincronización previa - CE2

En este caso de estudio se realiza una sincronización prescrita y la trayectoria es no alineada, la cual consta de los puntos: $\Omega = \{(3, 3), (2.759, 3.363), (2.372, 3.663), (1.890, 3.862), (1, 3.55), (3.943)\}$. El vector de variables de diseño es: $\mathbf{p} = [r_1, r_2, r_3, r_4, r_{cx}, r_{cy}]^T$. Las variables $x_0, y_0, \theta_0 = 0$, y la sincronización prescrita está dada por la siguiente secuencia de ángulos:

$$\theta_2^i = \left\{ \frac{2\pi}{12}, \frac{3\pi}{12}, \frac{4\pi}{12}, \frac{5\pi}{12}, \frac{6\pi}{12} \right\}. \quad (12)$$

Los límites superior e inferior de las variables de diseño se encuentran definidos por: $r_1, r_2, r_3, r_4 \in [0, 50]$ y $r_{cx}, r_{cy} \in [-50, 50]$.

En este caso, el problema de optimización mono-objetivo se describe de la Ecuación 13 a la 17:

$$\min f(\mathbf{p}) = \sum_{i=1}^N [(C_{xd}^i - C_x^i)^2 - (C_{yd}^i - C_y^i)^2], \quad \mathbf{p} \in \mathbb{R}^6, \quad (13)$$

sujeto a:

$$g_1(\mathbf{p}) = r_1 + r_2 - r_3 - r_4 \leq 0, \quad (14)$$

$$g_2(\mathbf{p}) = r_2 - r_3 \leq 0, \quad (15)$$

$$g_3(\mathbf{p}) = r_3 - r_4 \leq 0, \quad (16)$$

$$g_4(\mathbf{p}) = r_4 - r_1 \leq 0. \quad (17)$$

2.3. Caso de estudio 3: generación de movimiento delimitado por un conjunto de pares de puntos - CE3

Finalmente se tiene el tercer caso de estudio, tomado de [2], en donde se consideran diez pares de puntos de precisión ($K = 10$). El vector de variables

de diseño es: $\mathbf{p} = [r_1, r_2, r_3, r_4, r_{cx}, r_{cy}, \theta_0, x_0, y_0, \theta_2^1, \dots, \theta_2^{10}]^T$, con los límites superior e inferior definidos como: $r_1, r_2, r_3, r_4 \in [0, 60]$, $r_{cx}, r_{cy}, x_0, y_0 \in [-60, 60]$ y $\theta_0, \theta_2^1, \dots, \theta_2^{10} \in [0, 2\pi]$.

La función de este caso es diferente a la de los problemas anteriores, ya que considera la distancia del acoplador a los dos puntos de precisión de cada par de coordenadas. Los pares de puntos pueden ser encontrados en [2] y el problema de optimización mono-objetivo se describe de la Ecuación 18 a la 31:

$$\min f(\mathbf{p}) = \sum_{i=1}^K [(C_{1xd}^i - C_x^i)^2 - (C_{1yd}^i - C_y^i)^2] + \sum_{i=1}^K [(C_{2xd}^i - C_x^i)^2 - (C_{2yd}^i - C_y^i)^2],$$

$$\mathbf{p} \in \mathbb{R}^{19}, \tag{18}$$

Sujeto a:

$$g_1(\mathbf{p}) = r_1 + r_2 - r_3 - r_4 \leq 0, \tag{19}$$

$$g_2(\mathbf{p}) = r_2 - r_3 \leq 0, \tag{20}$$

$$g_3(\mathbf{p}) = r_3 - r_4 \leq 0, \tag{21}$$

$$g_4(\mathbf{p}) = r_4 - r_1 \leq 0, \tag{22}$$

$$g_5(\mathbf{p}) = \theta_2^1 - \theta_2^2 \leq 0, \tag{23}$$

$$g_6(\mathbf{p}) = \theta_2^2 - \theta_2^3 \leq 0, \tag{24}$$

$$g_7(\mathbf{p}) = \theta_2^3 - \theta_2^4 \leq 0, \tag{25}$$

$$g_8(\mathbf{p}) = \theta_2^4 - \theta_2^5 \leq 0, \tag{26}$$

$$g_9(\mathbf{p}) = \theta_2^5 - \theta_2^6 \leq 0, \tag{27}$$

$$g_{10}(\mathbf{p}) = \theta_2^6 - \theta_2^7 \leq 0, \tag{28}$$

$$g_{11}(\mathbf{p}) = \theta_2^7 - \theta_2^8 \leq 0, \tag{29}$$

$$g_{12}(\mathbf{p}) = \theta_2^8 - \theta_2^9 \leq 0, \tag{30}$$

$$g_{13}(\mathbf{p}) = \theta_2^9 - \theta_2^{10} \leq 0. \tag{31}$$

3. Enfoque propuesto

3.1. Algoritmo L-SHADE

El algoritmo Evolución Diferencial (DE por sus siglas en inglés) fue desarrollado por Price y Storn en 1995 para ser un optimizador de funciones confiable, versátil y fácil de usar. Desde entonces, DE ha sido probado en competencias como el ICEO (*International Contest on Evolutionary Optimization*) de la IEEE en 1996 y 1997, y en el mundo real en una amplia variedad de aplicaciones [11].

SHADE (Success-History Based DE) [14] es un algoritmo basado en Evolución Diferencial, que adapta los parámetros CR y F utilizando un historial de parámetros exitosos y surge como una versión extendida de JADE [9]. SHADE

obtuvo el cuarto lugar en la Sesión Especial y Competencia de Optimización Mono-Objetivo en Parámetros Reales CEC-2013.

L-SHADE [15] es una versión mejorada de SHADE, ganador de la misma competencia en el año 2014, que incorpora un mecanismo de reducción lineal del tamaño de población (LPSR por sus siglas en inglés) en el cual, de acuerdo a una función lineal, el tamaño de la población disminuye con el paso de las generaciones. En el resto de esta sección se describen las novedades presentadas en L-SHADE.

Asignación de parámetros basada en una memoria histórica

Como se puede observar en la Tabla 1, L-SHADE cuenta con una memoria de H entradas para los parámetros F y CR , denotadas como M_{CR} y M_F respectivamente. El contenido de $M_{CR,k}$ y $M_{F,k}$ ($k = 1, \dots, H$) se inicializa en 0.5. En cada generación g los valores F_i y CR_i empleados por cada individuo $x_{i,g}$, son generados seleccionando un índice o posición r_i de $[1, H]$ y aplicando las siguientes ecuaciones [15]:

$$CR_i = \begin{cases} 0 & \text{si } M_{CR,r_i} = \perp, \\ randn_i(M_{CR,r_i}, 0.1) & \text{en caso contrario,} \end{cases} \quad (32)$$

$$F_i = randc_i(M_{F,r_i}, 0.1). \quad (33)$$

Las expresiones $randn_i$ y $randc_i$ son distribuciones normal y de cauchy con medias M_{CR,r_i} , M_{F,r_i} y varianza de 0.1. En caso de que el valor de CR_i se encuentre fuera de los límites $[0,1]$, será reemplazado por el límite más cercano al valor obtenido. Si el valor $F_i > 1$, F_i se trunca a 1 y cuando $F_i < 0$, se aplica nuevamente la Ecuación 33 hasta obtener un valor aceptable. La forma en que se generan los parámetros CR_i y F_i fue adoptada del algoritmo JADE. En la Ecuación 32, el símbolo \perp hace referencia a un “valor terminal”, si M_{CR_i} tiene dicho valor asignado, entonces $CR_i = 0$.

Tabla 1: Memoria Histórica M_{CR} , M_F . Adaptada de [15].

Índice	1	2	...	$H - 1$	H
M_{CR}	$M_{CR,1}$	$M_{CR,2}$...	$M_{CR,H-1}$	$M_{CR,H}$
M_F	$M_{F,1}$	$M_{F,2}$...	$M_{F,H-1}$	$M_{F,H}$

Mutación de vectores utilizando current-to-pbest/1/bin

L-SHADE utiliza la estrategia de mutación current-to-pbest/1 (Ecuación 34) propuesta originalmente en [9], la cual es una variante de current-to-best/1, que incorpora un porcentaje p encargado de controlar qué tan rápido converge el algoritmo.

- current-to-pbest/1

$$v_{i,g} = x_{i,g} + F(x_{best}^p - x_{i,g}) + F(x_{r1,g} - x_{r2,g}), \quad (34)$$

en donde x_{best}^p es seleccionado aleatoriamente de entre los $p \times NP$ mejores individuos de la población. De acuerdo con los autores de dicha estrategia de mutación, el rango de valores recomendado para p es entre el 5% y el 20%, es decir, $p \in [0.05, 0.2]$.

Actualización de memoria histórica

En cada generación del algoritmo, los valores de F_i y CR_i que generaron un vector *trial* $u_{i,g}$ mejor que el *target* $x_{i,g}$, se guardan temporalmente en S_F y S_{CR} , y al final de la generación la memoria se actualiza usando el Algoritmo 1 [15].

Algoritmo 1 Actualización de memoria

```

1: if  $S_{CR} \neq \emptyset$  and  $S_F \neq \emptyset$  then
2:   if  $M_{CR,k,g} = \perp$  or  $\max(S_{CR} = 0)$  then
3:      $M_{CR,k,g+1} = \perp$ ;
4:   else
5:      $M_{CR,k,g+1} = mean_{WL}(S_{CR})$ ;
6:   end if
7:    $M_{F,k,g+1} = mean_{WL}(S_F)$ ;
8:    $k + +$ ;
9:   if  $k > H$  then
10:     $k = 1$ ;
11:   end if
12: end if

```

En el Algoritmo 1, el índice k ($1 \leq k \leq H$) determina la posición de la memoria que será actualizada. Al inicio del algoritmo, k se inicializa en 1 y se incrementa cada vez que se añade un nuevo valor a la memoria. Si $k > H$ entonces nuevamente se le asigna el valor 1. Cuando todos los vectores *trial* de la generación g son peores que los vectores *target*, S_F y S_{CR} se encontrarán vacíos y la memoria no se actualizará. El promedio ponderado de Lehmer $mean_{WL}(S)$, se calcula con la Ecuación 35. S se refiere tanto a S_F como a S_{CR} :

$$mean_{WL}(S) = \frac{\sum_{k=1}^{|S|} w_k \cdot S_k^2}{\sum_{k=1}^{|S|} w_k \cdot S_k}, \quad (35)$$

en donde w_k es calculado usando la diferencia de aptitud (Δf_k) entre el *trial* y el *target*, con el fin de influenciar la adaptación de parámetros (Ecuaciones 36 y 37):

$$w_k = \frac{\Delta f_k}{\sum_{m=1}^{|S|} \Delta f_m}, \quad (36)$$

$$\Delta f_k = |f(u_k, g) - f(x_k, g)|. \quad (37)$$

Durante la actualización de M_{CR} , si $M_{CR,k,g} = \perp$ o el máximo de los valores almacenados en S_{CR} es 0, se asignará a $M_{CR,k,g}$ el valor terminal \perp .

Reducción lineal del tamaño de población

El mecanismo de reducción de población en L-SHADE, decrementa linealmente el tamaño de la población como una función del número de evaluaciones de la función objetivo (NFE). En donde el tamaño de la población en la generación 1 es N^{init} , y el tamaño al final del proceso es N^{min} . Después de cada generación g , la población de la generación siguiente (NP_{g+1}) se calcula de acuerdo a la Ecuación 38 [15]:

$$NP_{g+1} = \text{round} \left[\left(\frac{N^{min} - N^{init}}{MAX_NFE} \right) \cdot NFE + N^{init} \right], \quad (38)$$

en donde a N^{init} se le asigna el número mínimo de individuos necesarios para aplicar los operadores evolutivos. En el caso de current-to-pbest (Ecuación 34), el número mínimo de individuos es 4. NFE es el número de evaluaciones de la función objetivo realizadas hasta el momento y MAX_NFE se refiere al número máximo de evaluaciones. Cada vez que NP_{g+1} sea menor que NP , los peores $NP - NP_{g+1}$ individuos de la población serán eliminados.

3.2. Adaptación de L-SHADE para Resolver Problemas Restringidos

Como se mencionó anteriormente, L-SHADE fue diseñado para resolver problemas sin restricciones, sin embargo, en este trabajo ha sido adaptado para manejar las restricciones relacionadas con el mecanismo de cuatro barras. A este nuevo enfoque se le ha llamado C-LSHADE (*Constrained-LSHADE*). En el Algoritmo 2 se muestra el pseudocódigo correspondiente a C-LSHADE y se marcan con una \Rightarrow las modificaciones realizadas al algoritmo original, las cuales fueron las siguientes:

- Añadir los criterios propuestos por Deb [10], para la selección del mejor individuo (línea 17). Dichos criterios contemplan tres escenarios, los cuales son: (1) cualquier individuo factible es preferible sobre cualquier individuo no factible, (2) entre dos individuos factibles, aquel que tenga el mejor valor de función objetivo será seleccionado y (3) entre dos individuos no factibles, aquel que tenga la menor suma de violación de restricciones (SVR) será seleccionado.

Es importante mencionar que para los escenarios 2 y 3, se consideró mejor al *trial* $u_{i,g}$, sólo si el valor comparado (Función Objetivo o SVR) era menor ($<$) que el valor del *target* $x_{i,g}$:

- Almacenar los valores Δf en $S_{\Delta f}$ (líneas 7 y 18). Aunque los autores de L-SHADE no lo mencionan, se pueden guardar los valores Δf (Ecuación 37) para realizar de forma más cómoda el cálculo de w_k (Ecuación 36) al final de la generación. En este trabajo, dichos valores se almacenan en $S_{\Delta f}$, una estructura similar a S_F y S_{CR} . Debe notarse que sin importar el escenario de comparación de Deb, siempre que el *trial* sea mejor que el *target*, se almacenará la diferencia de sus funciones objetivo.

- Ordenar a los individuos de la población con base en su SVR y Función Objetivo (línea 25). Para eliminar a los peores individuos, éstos deben ordenarse dando prioridad a la SVR y posteriormente a la Función Objetivo.

Algoritmo 2 C-LSHADE

```

1: Inicio
2: Crear aleatoriamente la población inicial  $x_{i,0}$ , donde  $i = 0, \dots, NP - 1$ 
3: Evaluar la población  $f(x_{i,0})$ , donde  $i = 1, \dots, NP$ 
4:  $g = 1, NP = N^{init}$ 
5: Establecer todos los valores de  $M_F$  y  $M_{CR}$  en 0.5
6: while (El criterio de paro no se haya cumplido) do
7:  $\Rightarrow S_F = \emptyset, S_{CR} = \emptyset, S_{\Delta f} = \emptyset$ 
8:   for  $i = 1$  hasta  $NP$  do
9:     Seleccionar aleatoriamente  $r_i$  de  $[1, H]$ 
10:     $F_{i,g} = randc_i(M_F, r_i, 0.1)$ 
11:    if  $M_{CR} = \perp$  then
12:       $CR_{i,g} = 0$ 
13:    else
14:       $CR_{i,g} = randn_i(M_{CR}, r_i, 0.1)$ 
15:    end if
16:    Generar el vector trial  $u_{i,g}$ , de acuerdo a current-to-pbest/1/bin
17:  $\Rightarrow$  if  $f(u_{i,g})$  es mejor que  $f(x_{i,g})$  según criterios de Deb
18:  $\Rightarrow F_{i,g} \rightarrow S_F, CR_{i,g} \rightarrow S_{CR}, |f(u_{i,g}) - f(x_{i,g})| \rightarrow S_{\Delta f}$ 
19:     $x_{i,g+1} = u_{i,g}$ 
20:  end if
21: end for
22: Actualizar las memorias  $M_F$  y  $M_{CR}$  (Algoritmo 1)
23: Calcular  $NP_{g+1}$  con la Ecuación 38
24: if  $NP_{g+1} < NP$  then
25:  $\Rightarrow$  Ordenar los individuos de la población  $P$ , de forma ascendente con base en su SVR y Función
    Objetivo, y eliminar los peores  $NP - NP_{g+1}$  miembros
26: end if
27:  $g++$ 
28: end while
29: Fin

```

4. Resultados

Dos experimentos fueron realizados para comparar el desempeño del algoritmo C-LSHADE contra un algoritmo DE/rand/1/bin del estado del arte [1], que promueve el uso de valores aleatorios para F y CR en la solución de problemas mecánicos de optimización. El primer experimento tenía como objetivo comparar los resultados obtenidos por los dos algoritmos, mientras que el segundo pretendía analizar el comportamiento de convergencia en cada uno. Con el fin de conocer qué tan difícil es encontrar soluciones factibles en cada caso de estudio, se calculó la medida ρ , que estima la proporción de la zona factible con respecto al espacio de búsqueda, calculando cuántas soluciones factibles existen en un millón de soluciones generadas de forma aleatoria. Los valores obtenidos fueron $\rho = 0.0033\%$, $\rho = 2.0607\%$ y $\rho = 0.0\%$ para CE1, CE2 y CE3, respectivamente. Estos valores indican que la región factible es muy pequeña y por lo tanto la optimización numérica con restricciones se complica.

En el primer experimento, se realizaron 31 ejecuciones independientes en ambos algoritmos por cada caso de estudio descrito en la Sección 2, y se obtuvieron

Tabla 2: Configuración de parámetros para DE/rand/1/bin basado en enfoque de [1].

Parámetros	DE(CE1)	DE(CE2)	DE(CE3)
NP	138	50	138
MAX_GEN	5435	600	5435
F (aleatorio entre)	[0.3, 0.9]	[0.3, 0.9]	[0.3, 0.9]
CR (aleatorio entre)	[0.8, 1)	[0.8, 1)	[0.8, 1)

medidas estadísticas. Para validar los datos obtenidos, se utilizó la prueba de la suma de rangos de Wilcoxon con un nivel de significancia del 95 % y en todos los casos las diferencias fueron significativas.

La configuración de parámetros empleada para el algoritmo DE/rand/1/bin se puede apreciar en la Tabla 2. El tamaño de la población NP y el número máximo de generaciones MAX_GEN se ajustaron con base en una experimentación previa, tomando en cuenta que para el CE2 se realizaran 30,000 evaluaciones y para los otros dos casos de estudio se efectuaran 750,000 aproximadamente, y seleccionando aquellos valores que brindaron mejores resultados.

En contraste, los parámetros F , CR y NP no necesitan ser configurados en L-SHADE ya que se controlan de forma automática. Sin embargo, L-SHADE incorpora nuevos parámetros que a pesar de ser menos sensibles que los parámetros originales de Evolución Diferencial, también deben ajustarse. En la variante⁴ de L-SHADE implementada en este trabajo deben configurarse los parámetros H , N^{init} y p , que corresponden al número de entradas de la memoria, la población inicial y el porcentaje de current-to- p best respectivamente. Para los tres casos de estudio, los valores que se utilizaron en los parámetros mencionados se mantuvieron iguales que los recomendados por los creadores del algoritmo, que son: $H = 6$, $N^{init} = (D \times 18)$ y $p = 0.11$ [15]. Cabe mencionar que N^{init} varía de acuerdo a la dimensión del problema. El “valor terminal” \perp mencionado en la Ecuación 32 se estableció en -1 .

Los resultados finales obtenidos por ambos algoritmos se muestran en la Tabla 3, en donde se puede observar que en el CE1, C-LSHADE llega a un mejor resultado que DE/rand/1/bin y emplea un 46.66 % menos evaluaciones para encontrarlo. Para el CE2 y el CE3, ambos algoritmos encuentran soluciones competitivas, sin embargo, C-LSHADE emplea un 33.33 % menos evaluaciones en el CE2 y un 40 % menos en el CE3. Con base en los resultados presentados del primer experimento, se puede afirmar lo siguiente:

- C-LSHADE es capaz de encontrar soluciones mejores o iguales que DE.
- El porcentaje de evaluaciones de la función objetivo que emplea C-LSHADE para encontrar la mejor solución, es considerablemente menor que en DE.
- C-LSHADE libera al usuario de la tarea de ajustar los parámetros F, CR y NP.

⁴ El algoritmo original tiene dos variantes que se distinguen por el uso de un archivo externo. En tal archivo, se almacenan los vectores *targets* que no fueron seleccionados y posteriormente se ocupan para realizar la mutación.

Tabla 3: Resultados estadísticos obtenidos por C-LSHADE y DE/rand/1/bin.

	CE1		CE2		CE3	
	C-LSHADE	DE	C-LSHADE	DE	C-LSHADE	DE
Mínimo	0	1.76E-28	2.62E-03	2.62E-03	2.74E-01	2.74E-01
Máximo	6.42E-04	2.76E-02	2.62E-03	2.62E-03	1.35E+01	1.35E+01
Promedio	2.40E-04	2.98E-03	2.62E-03	2.62E-03	2.43E-01	2.91E+00
Mediana	3.61E-08	4.27E-06	2.62E-03	2.62E-03	2.74E-01	4.39E-01
Desv. Estándar	2.76E-04	8.19E-03	2.46E-13	1.85E-11	4.93E+00	5.27E+00
Varianza	7.66E-08	6.70E-05	6.07E-26	3.45E-22	2.43E+01	2.78E+01
Evaluaciones	400,000	750,030	20,000	30,000	450,000	750,030

En el segundo experimento se graficaron los puntos de convergencia correspondientes a la ejecución ubicada en el valor de la mediana de las 31 ejecuciones independientes. En la Figura 2 se pueden apreciar las gráficas correspondientes a cada caso de estudio. Como se puede notar, C-LSHADE es capaz de encontrar mejores soluciones de forma más rápida que DE/rand/1/bin en el CE1 y el CE3. En el CE2, que es el más sencillo según la medida ρ , el algoritmo DE presenta un mejor comportamiento, sin embargo, al final de la ejecución C-LSHADE se recupera y llega incluso a una mejor solución que DE/rand/1/bin (Tabla 3).

Finalmente se muestran las mejores soluciones encontradas por C-LSHADE en los tres casos de estudio:

CE1: $\mathbf{p} = [38.45761229, 8.538400025, 28.15726641, 38.40204633, 37.83976344, 16.61315669, 3.949817649, -9.473622057, 59.45027718, 1.753788656, 2.466800077, 2.97662654, 3.472145065, 4.019649903, 5.124622599] = 0$

CE2: $\mathbf{p} = [14.31454712, 2.211165827, 14.31454712, 14.31454712, 2.174361582, 0.02220978] = 2.62E - 03$

CE3: $\mathbf{p} = [2.614591256, 1.034801411, 1.826885061, 2.207891397, 1.250924809, 0.447340736, 5.826803988, 0.099169548, 1.328797661, 0.410281169, 1.039364885, 1.6500082, 2.260034437, 2.865981272, 3.490250928, 4.163941448, 4.905546352, 5.416480201, 6.067608264] = 2.74E - 01$

5. Conclusiones

En este trabajo se modificó L-SHADE, un algoritmo adaptativo basado en Evolución Diferencial que controla los parámetros F , CR y NP , para resolver tres casos de estudio de la optimización de un mecanismo de cuatro barras. La modificación principal consistió en agregar los criterios de factibilidad propuestos por Deb en la etapa correspondiente a la selección del mejor individuo. A este enfoque se le llamó C-LSHADE.

Los resultados finales obtenidos por C-LSHADE fueron comparados contra un algoritmo DE/rand/1/bin del estado del arte que promueve valores aleatorios en F y CR para resolver problemas mecánicos de optimización. Con base en los

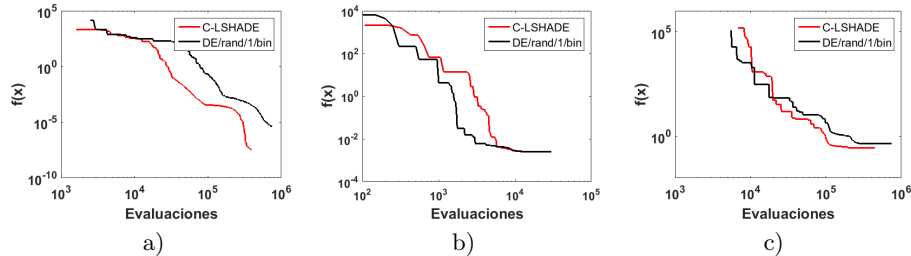


Fig. 2: Gráficas de convergencia de C-LSHADE y DE/rand/1/bin para el a) CE1, b) CE2 y c) CE3.

resultados presentados, se puede decir que aplicar C-LSHADE a problemas de diseño mecánico permite obtener soluciones mejores o iguales que DE/rand/1/bin, empleando un porcentaje considerablemente menor de evaluaciones de la función objetivo. Además, en los casos de estudio más complejos las gráficas de convergencia indicaron que el algoritmo propuesto es capaz de encontrar más rápido mejores soluciones. Finalmente, C-LSHADE libera al usuario de la tarea de realizar un ajuste manual de parámetros, que comúnmente se realiza con base en prueba y error. Como trabajo futuro se propone estudiar y adaptar la versión más reciente de L-SHADE presentada en [8] con el fin de comparar su desempeño contra C-LSHADE.

Agradecimientos. El primer autor agradece el apoyo de CONACyT a través de una beca para cursar estudios de posgrado en LANIA. El segundo autor agradece el apoyo por parte de CONACyT a través del proyecto No. 220522.

Referencias

1. Portilla-Flores, E. A., Mezura-Montes, E., Alvarez-Gallegos, J., Coello-Coello, C. A., Cruz-Villar, C. A., Villarreal-Cervantes, M. G.: Parametric reconfiguration improvement in non-iterative concurrent mechatronic design using an evolutionary-based approach. *Engineering Applications of Artificial Intelligence*, 24(5):757–771 (2011)
2. Sánchez-Márquez, A., Vega-Alvarado, E., Portilla-Flores, E. A., Mezura-Montes, E.: Synthesis of a planar four-bar mechanism for position control using the harmony search algorithm. In: *Electrical Engineering, Computing Science and Automatic Control (CCE)*, 11th International Conference on, pp. 1–6, IEEE (2014)
3. Dasgupta, D., Michalewicz, Z.: *Evolutionary algorithms in engineering applications*. Springer Science & Business Media (2013)
4. Eiben A. E., Michalewicz Z., Schoenauer M., Smith J. E.: Parameter control in evolutionary algorithms. In: *Parameter setting in evolutionary algorithms*, pp. 19–46, Springer (2007)

5. Mezura-Montes, E., Portilla-Flores, E. A., Capistran-Gumersindo, E.: Dynamic parameter control in differential evolution with combined variants to optimize a three-finger end effector. In: Power, Electronics and Computing (ROPEC), IEEE International Autumn Meeting on, pp. 1–6, IEEE (2015)
6. Vega-Alvarado, E., Portilla-Flores, E. A., Mezura-Montes, E., Flores-Pulido, L., Calva-Yáñez, M. B.: A memetic algorithm applied to the optimal design of a planar mechanism for trajectory tracking. *Polibits*, 53:83–90 (2016)
7. Villarreal-Cervantes, M. G., Cruz-Villar, C. A., Alvarez-Gallegos, J., Portilla-Flores, E. A.: Differential evolution techniques for the structure-control design of a five-bar parallel robot. *Engineering Optimization*, 42(6):535–565 (2010)
8. Brest, J., Maučec, M. S. y Bošković, B.: il-shade: Improved l-shade algorithm for single objective real-parameter optimization. In: Evolutionary Computation (CEC), IEEE Congress on, pp. 1188–1195, IEEE (2016)
9. Zhang, J., Sanderson, A. C.: Jade: adaptive differential evolution with optional external archive. *IEEE Transactions on evolutionary computation*, 13(5):945–958 (2009)
10. Deb, K.: An efficient constraint handling method for genetic algorithms. *Computer methods in applied mechanics and engineering*, 186(2):311–338 (2000)
11. Price, K., Storn, R., Lampinen, J.: Differential evolution: a practical approach to global optimization. Springer Science & Business Media (2006)
12. Pant, M., Thangaraj, R., Singh, V.P.: Optimization of mechanical design problems using improved differential evolution algorithm. *Int. Journal of Recent Trends in Engineering*, 1(5) (2009)
13. Siarry, P., Michalewicz, Z.: Advances in metaheuristics for hard optimization. Springer Science & Business Media (2007)
14. Tanabe, R., Fukunaga, A. S.: Success-history based parameter adaptation for differential evolution. In: Evolutionary Computation (CEC), IEEE Congress on, pp. 71–78, IEEE (2013).
15. Tanabe, R., Fukunaga, A. S.: Improving the search performance of shade using linear population size reduction. In: Evolutionary Computation (CEC), IEEE Congress on, pp. 1658–1665, IEEE (2014)
16. Shiakolas, P. S., Koladiya, D., Kebrle, J.: On the optimum synthesis of six-bar linkages using differential evolution and the geometric centroid of precision positions technique. *Mechanism and Machine Theory*, 40(3):319–335 (2005)