



Una estrategia para el diseño de filtros activos RC de alto orden utilizando algoritmos evolutivos

A Strategy for the Design of High-Order RC Active Filters Using Evolutionary Algorithms

Presentación: 03/07/2024

Aprobación: 03/02/2025

Publicación: 12/02/2025

Mónica Andrea Lovay

 0000-0003-4895-4375

Universidad Tecnológica Nacional, Facultad Regional Villa María, Grupo de Estudio de calidad en Mecatrónica (GeCAM), Argentina
mlovay@frvm.utn.edu.ar

Gabriela Marta Peretti

 0000-0003-1489-7982

Universidad Tecnológica Nacional, Facultad Regional Villa María, Grupo de Estudio de calidad en Mecatrónica (GeCAM), Argentina
Universidad Nacional de Córdoba, Facultad de Matemática, Astronomía, Física y Computación Grupo de Desarrollo en Electrónica e Instrumental, Argentina
gperetti@frvm.utn.edu.ar

Eduardo Abel Romero

 0000-0001-8378-950X

Universidad Tecnológica Nacional, Facultad Regional Villa María, Grupo de Estudio de calidad en Mecatrónica (GeCAM), Argentina
Universidad Nacional de Córdoba, Facultad de Matemática, Astronomía, Física y Computación Grupo de Desarrollo en Electrónica e Instrumental, Argentina
eduardoromero@frvm.utn.edu.ar

Resumen

Este trabajo presenta una estrategia modular de diseño para filtros de alto orden implementados en cascada usando algoritmos evolutivos (EAs, Evolutionary Algorithms). El método propuesto evidencia notables ventajas con respecto a los formulados en la literatura para dimensionar este tipo de filtros. Se emplean cuatro EAs del estado del arte: PSO (Particle Swarm Optimization), DE (Differential Evolution), ADE (Average Differential Evolution) y EHO (Elephant Herding Optimization). Como casos de estudio se adoptan dos

filtros Chebyshev de décimo orden considerados en trabajos previos. Los resultados muestran que mediante la estrategia propuesta se obtienen configuraciones de filtro con menor error de diseño que otras alternativas, aun empleando componentes de mayor tolerancia. El esfuerzo computacional también resulta ser notablemente más bajo. Una comparación estadística entre los EAs utilizados muestra que DE y ADE son más eficientes en cuanto a la calidad de las configuraciones encontradas.

Palabras claves: algoritmos evolutivos; diseño de filtros de alto orden; selección de componentes pasivos; estrategia de diseño de filtros.

Abstract

This paper presents a modular design strategy for high-order filters implemented in cascade using evolutionary algorithms (EAs). The proposed method demonstrates notable advantages over those formulated in the literature for sizing this type of filters. Four state-of-the-art EAs are employed: PSO (Particle Swarm Optimization), DE (Differential Evolution), ADE (Average Differential Evolution), and EHO (Elephant Herding Optimization). Two tenth-order Chebyshev filters considered in previous works are adopted as case studies. The results show that the proposed strategy obtains filter configurations with lower design errors than other alternatives, even when using lower-cost components. The computational effort is also notably lower. A statistical comparison among the EAs used shows that DE and ADE are more efficient in terms of the quality of the found configurations.

Keywords: evolutionary algorithms; high order filter design; selection of passive components; filter design strategy.

1. Introducción

Los filtros activos RC, compuestos por amplificadores operacionales, capacitores y resistencias, son muy utilizados en multiplicidad de aplicaciones, como en filtrado de señales de sensores en sistemas automotrices, *antialiasing* en electrónica digital, reducción de oscilaciones en sistemas de control, filtrado de precisión en telecomunicaciones, entre otras. Hay diferentes topologías para implementarlos, como por ejemplo, los filtros en escalera (Deliyannis et al., 1999), que tienen propiedades de sensibilidad superiores a los bicuadráticos o de segundo orden. Sin embargo, los bicuadráticos son muy usados debido a que simplifican el diseño de filtros de orden superior mediante la conexión en cascada (Karki, 2023; Zumbahlen, 2007).

El diseño de filtros activos RC es un proceso complejo que requiere el dimensionamiento de numerosos componentes pasivos para satisfacer el conjunto de especificaciones planteadas. Los enfoques tradicionales (Deliyannis et al., 1999; Raut R. & Swamy M. N. S., 2010) reducen los grados de libertad para simplificar el problema, a expensas de presentar resultados que no siempre alcanzan los objetivos. Además, brindan como resultado números reales para los valores de componentes. Estos deben aproximarse a los disponibles comercialmente, que están estandarizados de acuerdo con las denominadas series E. Como consecuencia, surgen errores que requieren iterar hasta encontrar la solución adecuada.

Estos inconvenientes desafían la tarea de diseño, siendo motivo para que la comunidad científica desarrolle metodologías basadas en algoritmos evolutivos (EAs, Evolutionary

Algorithms). Estas mapean el problema a una optimización que minimiza una función de error de las especificaciones de un diseño particular respecto de las deseadas. Los EAs imitan las fuerzas de la evolución natural y la autoadaptación (Slowik, 2020; Yu X. & Gen M., 2010), para obtener soluciones cercanas a la óptima en espacios de búsqueda grandes. En el contexto del dimensionamiento de filtros se utilizan para obtener errores considerablemente menores a los del tradicional y agilizar el proceso debido a la facilidad de sistematización que presentan.

Particularmente, el diseño de filtros de alto orden se presenta como un desafío para los EAs debido a que el problema de optimización crece en número de variables a medida que aumenta el orden. Alternativas para filtros de alto orden realizados como cascada de secciones bicuadráticas se presentan en la literatura. En (Durmuş et al., 2020) se considera un filtro pasabajo Chebyshev de décimo orden usando PSO (Particle Swarm Optimization), DE (Differential Evolution), ADE (Average Differential Evolution), y CSS (Charged System Search). CSS es empleado en (Hiçdurmaz et al., 2019) y (Temurtaş, 2020), para seleccionar los componentes de dos filtros pasabajo Bessel y Butterworth, respectivamente, ambos de décimo orden. En (Beşkirli & Kiran, 2023), se considera la respuesta pasa-bajo y pasa-alto de los dos filtros mencionados, dimensionándolos con I-TSA (Improved Tree-Seed Algorithm). El algoritmo HBO (Heap-Based Optimizer) es utilizado en (Kuyu & Vatansever, 2023) para el dimensionamiento de tres filtros pasabajo Butterworth de sexto, octavo y décimo orden.

En los trabajos anteriores, una función costo que contempla los valores de todos los componentes del circuito guía el proceso de optimización. Consecuentemente, el proceso se complejiza conforme el orden del filtro aumenta, requiriendo la adopción de series E de alta precisión para obtener resultados con bajo error, que resulta ingenierilmente cuestionable. Por otro lado, proceder a la optimización completa del filtro ignorando la natural partición y aislación que se da en estas estructuras conduce a un problema más difícil de resolver. Estas características sugieren que una optimización etapa por etapa potencialmente debiera conducir a mejores resultados. Esta hipótesis es la que motiva la exploración presentada aquí.

En particular, en este trabajo se aplican cuatro EAs: PSO, DE, ADE y EHO (Elephant Herding Optimization) para diseñar cada etapa bicuadrática de los filtros de alto orden por separado. Se adoptan los casos de estudio presentados en (Durmuş et al., 2020) y (Kuyu & Vatansever, 2023), con el objeto de comparar desempeños. Nuestros resultados muestran una clara ventaja frente a los presentados en (Durmuş et al., 2020) y (Kuyu & Vatansever, 2023), con menores costos computacionales e incluso empleando componentes de tolerancias mayores, usualmente de menor costo.

2. Desarrollo

2.1. Filtros bajo estudio

Una de las técnicas más comunes para la realización de filtros de alto orden es conectar en cascada dos o más secciones de primer y segundo orden (Raut R. & Swamy M. N. S., 2010). Para ello, los polinomios del numerador y del denominador de la función de transferencia del filtro $H(s)$, expresada en (1), se factorizan en términos bicuadráticos (y un término de primer orden en caso de filtros de orden impar). Sin pérdida de generalidad, consideramos los filtros de orden par, que pueden factorizarse como (2).

$$H(s) = \frac{(a_m s^m + a_{(m-1)} s^{(m-1)} + \dots + a_1 s + a_0)}{(s^n + b_{(n-1)} s^{(n-1)} + \dots + b_1 s + b_0)} \quad (1)$$

$$H(s) = \prod_{i=1}^{n/2} G_i \frac{\alpha_{2i} s^2 + \alpha_{1i} s + \alpha_{0i}}{s^2 + s\omega_i / Q_i + \omega_i^2} = \prod_{i=1}^{n/2} H_i(s) \quad (2)$$

En (2), Q_i es el factor de calidad, ω_i es la frecuencia de polo, los coeficientes α_{2i} , α_{1i} y α_{0i} determinan el tipo de función de segundo orden (pasaalto, pasabanda, pasabajo, etc) y G_i es la ganancia. La implementación de cada etapa bicuadrática i puede realizarse utilizando diversas topologías empleando uno o varios amplificadores operacionales (Deliyannis et al., 1999; Raut R. & Swamy M. N. S., 2010). Debe mencionarse que ω_i puede especificarse como el producto de un coeficiente FSFi que varía para cada etapa, por una frecuencia de corte f_c , que suele darse como especificación global del filtro (3). Esta nomenclatura será útil para especificar los casos de estudio que se detallan a continuación.

$$\omega_i = 2\pi FSF_i \cdot f_c \quad (3)$$

Este trabajo adopta como casos de estudio dos filtros Chebyshev de décimo orden, uno con 1dB de ripple y el otro con 3 dB de ripple, ambos con frecuencia de corte $f_c=10$ KHz y ganancia unitaria (Durmuş et al., 2020). Los valores especificados para FSF ($FSF_{e,i}$) y Q ($Q_{e,i}$) de cada etapa i , con $i=1..5$, se reportan en la tabla 1.

Filtro	Etapa 1		Etapa 2		Etapa 3		Etapa 4		Etapa 5	
	$FSF_{e,1}$	$Q_{e,1}$	$FSF_{e,2}$	$Q_{e,2}$	$FSF_{e,3}$	$Q_{e,3}$	$FSF_{e,4}$	$Q_{e,4}$	$FSF_{e,5}$	$Q_{e,5}$
1 dB de ripple	0,2121	0,7495	0,4760	1,8639	0,7214	3,5609	0,9024	6,9419	0,9981	22,2779
3 dB de ripple	0,1796	1,0289	0,4626	2,935	0,7126	5,7012	0,8954	11,1646	0,9916	35,9274

Tabla 1: Valores FSF y Q especificados para los casos de estudio

Ambos filtros se sintetizan con la topología de la Fig. 1, en donde cada etapa bicuadrática está implementada por una estructura IGMFB (Infinite-gain multiple feedback) (Raut R. & Swamy M. N. S., 2010).

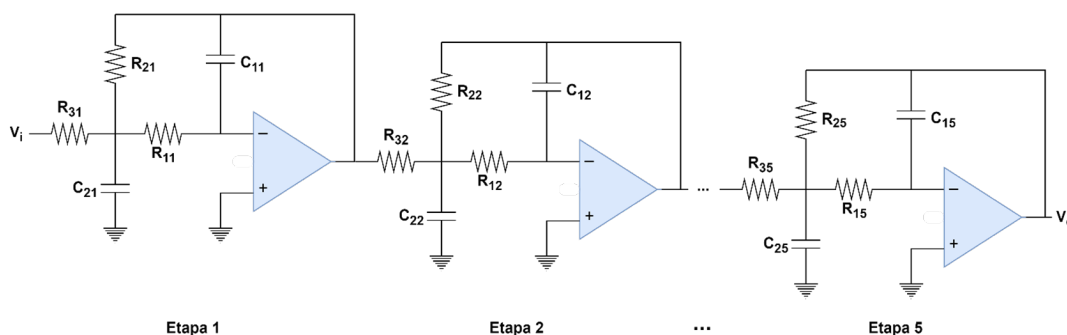


Fig. 1: Filtro IGMFB de décimo orden

En términos de las especificaciones dadas anteriormente, la función de transferencia de un filtro pasabajo de orden 10 es:

$$H(s) = \prod_{i=1}^5 H_i(s) = \prod_{i=1}^5 \left(\frac{G_i}{(2\pi F S F_i \cdot f_c \cdot s)^2 + \frac{s}{2\pi F S F_i \cdot f_c \cdot Q_i} + 1} \right) \quad (4)$$

Por otro lado, una etapa IGMFB tiene una función de transferencia en función de los componentes dada por:

$$H_i(s) = \left(\frac{-R_{2i} / R_{3i}}{s^2 R_{1i} R_{2i} C_{1i} C_{2i} + s \left(R_{1i} + R_{2i} + \frac{R_{1i} R_{2i}}{R_{3i}} \right) C_{1i} + 1} \right) \quad (5)$$

Se considera $G_i=1$ para todas las etapas, por lo que $R_{2i}=R_{3i}$.
 Por similitud entre las ecuaciones (4) y (5), se encuentra que:

$$F S F_i = \frac{1}{2 f_c \sqrt{R_{1i} R_{2i} C_{1i} C_{2i}}}; \quad Q_i = \frac{\sqrt{R_{1i} R_{2i} C_{1i} C_{2i}}}{(2 R_{1i} + R_{2i}) C_{1i}} \quad (6)$$

Para la selección de los valores de los componentes se consideran tres escenarios: el primero asume valores de la serie E192, el segundo de E96 y el tercero de E24.

El escenario E192 presenta la mayor cantidad de componentes disponibles, asociados a una menor tolerancia de fabricación (mayor costo). El E96 considera menos componentes, asociados a mayor tolerancia. Finalmente, E24 presenta la menor cantidad de componentes y mayor tolerancia de fabricación (pero menor costo).

En todos los casos, el rango de valores admitidos es [100Ω; 1MΩ] para las resistencias y [10pF; 10μF] para los capacitores. Estos valores se consideran adecuados para el correcto funcionamiento de las etapas IGMFB de la Fig. 1 y son los sugeridos por los fabricantes (Karki, 2023; Zumbahlen, 2007).

Finalmente, la selección de los componentes activos también debe realizarse teniendo en cuenta las recomendaciones de los fabricantes, como por ejemplo las realizadas en (Karki, 2023), garantizando que sus especificaciones no afecten el desempeño del filtro. Posteriormente, deberían efectuarse simulaciones a nivel eléctrico (de tipo Spice) para cuantificar el impacto de las no-idealidades.

2.2. Formulación del problema

Primeramente, consideramos la formulación realizada previamente en (Durmuş et al., 2020; Kuyu & Vatansever, 2023), a la que denominaremos de tipo global, para explicitar sus problemas. A continuación, definiremos nuestra estrategia. En (Durmuş et al., 2020; Kuyu & Vatansever, 2023), se considera un individuo Y_j que contiene los 25 valores de los

componentes pasivos del filtro de décimo orden como los adoptados en este trabajo:

$$Y_j = [y_1, y_2, \dots, y_{25}] \quad (7)$$

En base a este individuo, se determinan los errores en FSF (8) y en Q (9) como la suma de los errores de cada etapa i con respecto a los valores especificados $FSF_{e,i}$ y $Q_{e,i}$ dados en la tabla 1:

$$Error_{FSF}(Y_j) = \sum_{i=1}^5 \frac{|FSF_{e,i} - FSF_i(Y_j)|}{Q_{e,i}} \quad (8)$$

$$Error_Q(Y_j) = \sum_{i=1}^5 \frac{|Q_{e,i} - Q_i(Y_j)|}{Q_{e,i}} \quad (9)$$

El proceso de optimización utiliza como guía la función $Error_{Total}$, que resulta del promedio de los errores (8) y (9), buscando los individuos Y_j que la minimicen:

$$Error_{Total}(Y_j) = 0,5 (Error_{FSF}(Y_j) + Error_Q(Y_j)), \quad Y_j = [y_1, y_2, \dots, y_{25}] \quad (10)$$

Si bien el diseñador del algoritmo tiene libertad para definir como se guía la optimización, resulta claro que la función de búsqueda del mejor filtro utilizando $Error_{Total}(Y_j)$ no asume independencia en las etapas. En este sentido, cada individuo (Y_j) presenta los valores de todos los componentes y por ende lleva la información de todas las etapas a la vez. Por otro lado, como el fitness ($Error_{Total}(Y_j)$) se calcula mediante la suma de los errores de diseño de cada etapa, el valor de cada componente influye solamente en uno de los sumandos de la función fitness (el que representa el error de la etapa a la que corresponde ese componente). En consecuencia, se puede dar la circunstancia de que un individuo que presenta un error muy bajo en una de las etapas también posea altos errores en las restantes y se descarte. Se perderá por tanto la información de un subconjunto de componentes que presentaban un error bajo en una etapa en particular, que podría haber resultado promisoria. Esta característica resulta en una exploración deficiente del espacio de búsqueda.

Por el contrario, la optimización propuesta en este trabajo toma ventaja de la natural partición en secciones bicuadráticas y optimiza etapa por etapa, dado que la determinación de los componentes de una no afecta a las demás. Basados en estas consideraciones, ahora cada individuo $Y_{j,i}$ del algoritmo es un conjunto de valores de resistencias y capacitores de la etapa i :

$$Y_{j,i} = [y_{1,j,i}, y_{2,j,i}, \dots, y_{5,j,i}], \quad \text{para } i = 1, 2, \dots, 5 \quad (11)$$

El error en FSF (12) y Q (13) se definen para cada etapa i :

$$Error_{FSF,i}(Y_{j,i}) = \left(\frac{FSF_{e,i} - FSF_i(Y_{j,i})}{FSF_{e,i}} \right) \quad (12)$$

$$Error_{Q,i}(Y_{j,i}) = \left(\frac{Q_{e,i} - Q_i(Y_{j,i})}{Q_{e,i}} \right) \quad (13)$$

Y el proceso de optimización busca el mejor individuo que minimice el promedio de error, que es la función $Error(Y_{j,i})$:

$$Error(Y_{j,i}) = 0,5 \left(Error_{FSF,i}(Y_{j,i}) + Error_{Q,i}(Y_{j,i}) \right) \quad (14)$$

Nuestra propuesta resuelve el problema anterior, dado que cada individuo no lleva información de otras etapas por lo que no se descartan soluciones que presenten error mínimo.

El proceso se repite para todas las etapas del filtro, obteniendo el mejor individuo $Y_{mejor,i}$ para cada una de ellas. Finalmente, la mejor configuración Y_{mejor} resulta de la unión entre $Y_{mejor,i}$:

$$Y_{mejor} = \left(Y_{mejor,1}, Y_{mejor,2}, Y_{mejor,3}, Y_{mejor,4}, Y_{mejor,5} \right) \quad (15)$$

Posteriormente, sobre Y_{mejor} se puede aplicar la función $Error_{Total}$, que utilizaremos como métrica para caracterizar el desempeño de los EAs y comparar con otros trabajos.

2.3. Algoritmos evolutivos adoptados e implementación

Los EAs PSO, DE, ADE y EHO se codifican en Matlab. Estos algoritmos presentan varias características comunes: están basados en poblaciones que se generan de manera aleatoria, utilizan una función objetivo (o función *fitness*) para guiar la búsqueda y operan iterativamente actualizando la población en cada generación. La diferencia entre ellos radica en la manera en que cada uno realiza el proceso de selección y la información que memorizan o comparten durante la búsqueda (Slowik, 2020; Yu X. & Gen M., 2010).

Por otro lado, los EAs no evalúan todas las posibles soluciones, sino que realizan una búsqueda aproximada, guiada por heurísticas y elementos aleatorios, lo que implica que no es posible asegurar su convergencia al óptimo global en todos los casos. Sin embargo, cada una incorpora estrategias particulares para escapar de óptimos locales, aumentando su robustez frente a problemas complejos y multimodales (Durmuş et al., 2020; Slowik, 2020).

La generación aleatoria de individuos en las poblaciones es una característica compartida. Para una etapa IGMFB (Fig. 1) definimos el siguiente individuo que contiene los valores de los componentes pasivos:

$$Y_{j,i} = \left[R_{1,j,i}, R_{2-3,j,i}, C_{1,j,i}, C_{2,j,i} \right] \quad (16)$$

En (16), $Y_{j,i}$ presenta cuatro componentes en vez de cinco porque $R_{2,j,i} = R_{3,j,i}$, debido a la ganancia unitaria (Sección 2.1).

La compatibilidad con las series E implica un universo de búsqueda discreto por lo que se realiza un cambio a variable entera en el proceso. Para ello, se define el siguiente individuo:

$$X_{j,i} = [x_{1,j,i}, x_{2,j,i}, x_{3,j,i}, x_{4,j,i}] \quad (17)$$

En (17), cada elemento de $X_{j,i}$ es un valor entero que representa la ubicación que tiene un valor para $R_{1,j,i}$, $R_{2-3,j,i}$, $C_{1,j,i}$ y $C_{2,j,i}$ en un vector que contiene todos los valores posibles de la serie E en el rango especificado. Sobre este vector se realizan las diferentes operaciones de los algoritmos.

La Fig. 2 muestra el pseudocódigo de los algoritmos, que será descrito a continuación. Este se realiza para las cinco etapas bicuadráticas, indicadas por la variable i . En cada generación, los individuos se diferencian por la variable j .

```

1.  INICIO
2.   $i = 1$  //etapa bicuadrática
3.  Mientras  $i \leq 5$  hacer
4.      Generar población inicial de  $N$  vectores posición  $X_{j,i} = [x_{1,j,i}, x_{2,j,i}, x_{3,j,i}, x_{4,j,i}]$ ,
        donde  $j=1,2,\dots,N$ .
5.      Generación=1
6.      Repetir
7.          Aplicar función de mapeo para obtener población de  $N$  individuos  $Y_{j,i}$ 
8.          Evaluar el fitness de cada individuo  $Y_{j,i}$  usando (14)
9.          Según Sea EA Hacer //aplicar operadores del EA sobre vectores  $X_{j,i}$ 
10.             PSO: Aplicar operador de actualización de velocidad y
                de actualización de posición a  $X_{j,i}$ 
11.             DE: Aplicar operador de mutación diferencial, de recombinación y
                de selección a  $X_{j,i}$ 
12.             ADE: Aplicar operador de mutación de ADE, de recombinación y de
                selección a  $X_{j,i}$ 
13.             EHO: Aplicar operador de actualización de clan y de separación a  $X_{j,i}$ 
14.             Fin Según
15.             Aplicar función de ajuste a cada vector  $X_{j,i}$ 
16.             Generación= Generación + 1
17.             Hacer población actual<-nueva población de  $N$  vectores  $X_{j,i}$ 
18.             Hasta que Generación> $M$ 
19.             Hacer  $Y_{mejor,i}$  <- individuo  $Y_{j,i}$  con el menor fitness
20.              $i=j+1$ 
21.             Fin Mientras
22.             Retornar  $Y_{mejor} = (Y_{mejor,1}, Y_{mejor,2}, Y_{mejor,3}, Y_{mejor,4}, Y_{mejor,5})$ 
23.             FIN

```

Fig. 2: Pseudocódigo de los EAs

En la línea 4 del pseudocódigo se genera un conjunto N de posiciones $X_{j,i}$ aleatorias iniciales. Para evaluar el fitness o qué tan bien un individuo cumple con el objetivo se utiliza (14). Sin embargo, esta función requiere de los valores de los componentes, no de las posiciones. Se necesita, por tanto, para cada $X_{j,i}$ obtener el individuo $Y_{j,i}$ correspondiente. Esto se realiza utilizando la función de mapeo (línea 7) que devuelve para cada valor de $X_{j,i}$ su correspondiente valor de resistencia o capacitor, de acuerdo con la serie E elegida y dentro de los rangos especificados.

Luego de la evaluación del fitness se procede a aplicar sobre cada vector $X_{j,i}$ los operadores específicos de cada algoritmo, los cuales permiten, entre otros aspectos, compartir la información entre individuos, actualizar la población y evitar la convergencia a óptimos locales:

- En PSO (Durmuş et al., 2020; Jin et al., 2007; Slowik, 2020), (línea 10), cada $X_{j,i}$ es considerado una partícula que se mueve en un espacio de búsqueda multidimensional. La trayectoria de cada partícula en el espacio de búsqueda es ajustada dinámicamente mediante la alteración de su velocidad. Teniendo en cuenta la memoria de la mejor posición individual alcanzada (influencia cognitiva) y el conocimiento de la mejor posición global (influencia social), el operador actualiza la velocidad y la posición de cada partícula. Requiere definir los parámetros C_1 y C_2 , que representan los coeficientes social y cognitivo, respectivamente. Además, W_{min} y W_{max} definen el rango de valores en el que varía la masa inercial durante el proceso de optimización.
- En DE (Durmuş et al., 2020; Slowik, 2020), (línea 11), cada $X_{j,i}$ se considera un vector numérico. El operador aplicado en cada iteración es mutación diferencial, recombinación y selección. Requiere definir los parámetros tasa de mutación (F) y probabilidad de cruce (CR).
- ADE (Durmuş et al., 2020), (línea 12) es una versión de DE que utiliza un operador de mutación diferente. Además, usa el promedio de individuos en una población, y monitorea al mejor individuo.
- En EHO (Durmuş et al., 2020; Slowik, 2020), (línea 13), cada individuo representa a un elefante que forma parte de un clan liderado por una matriarca. Cada elefante implementa un operador de actualización de clan para actualizar su posición y la de la matriarca. Posteriormente, el peor individuo es reemplazado por un operador separador que acelera la velocidad de convergencia y mejora la diversidad de la población. Requiere definir la influencia de la matriarca α , mientras que β controla la influencia del centro del clan.

Luego de este proceso cada $X_{j,i}$ adquiere para algunos o todos sus elementos valores decimales. Por este motivo, es aplicada una *función de ajuste* (línea 15) en cada elemento de $X_{j,i}$. Si bien existen muchas propuestas para realizar esta función de ajuste, utilizaremos la más sencilla que consiste en eliminar la parte decimal (Jin et al., 2007; Lampinen & Zelinka, 1999; Şimşir & Taşpınar, 2021). Si luego de aplicar este proceso de discretización existen elementos del vector $X_{j,i}$ que se encuentran fuera del rango, éstos son ajustados al valor extremo más cercano.

La nueva generación obtenida en el paso anterior reemplaza a la generación actual (línea 17) y se repite el proceso.

El criterio de finalización para una etapa i , es alcanzar el máximo número de generaciones M (línea 18). Cuando esto ocurre se selecciona de la corrida del algoritmo la configuración $Y_{j,i}$ que presentó el menor valor de fitness y se almacena en $Y_{mejor,i}$. Luego se repite el

proceso completo (desde la línea 4) hasta completar las cinco etapas bicuadráticas. El resultado final es la configuración total Ymejor.

Para facilitar la comprensión del pseudocódigo, la Fig. 3 ilustra cómo se generan los valores de los diferentes conjuntos involucrados en los procesos de optimización. Esta considera un hipotético vector $X_{3,2}=[2,21 \ 958,55 \ 1344,88 \ 3,01]$ que es el tercer vector posición de una población para la etapa 2 ($X_{3,2}$). Luego se ajusta a $X_{3,2}=[2 \ 958 \ 1344 \ 3]$. En este último, el primer elemento $x_{1,3,2}=2$ corresponde a la posición de una resistencia de valor 101Ω , por lo que $R_{1,3,2}=101$; mientras que el valor $x_{3,3,2}=1344$ corresponde a un capacitor de valor $9,88\mu\text{F}$, por lo que $C_{1,3,2}=9,88\text{E-}06$. Sobre $Y_{3,2}$ se aplica la función fitness (14).

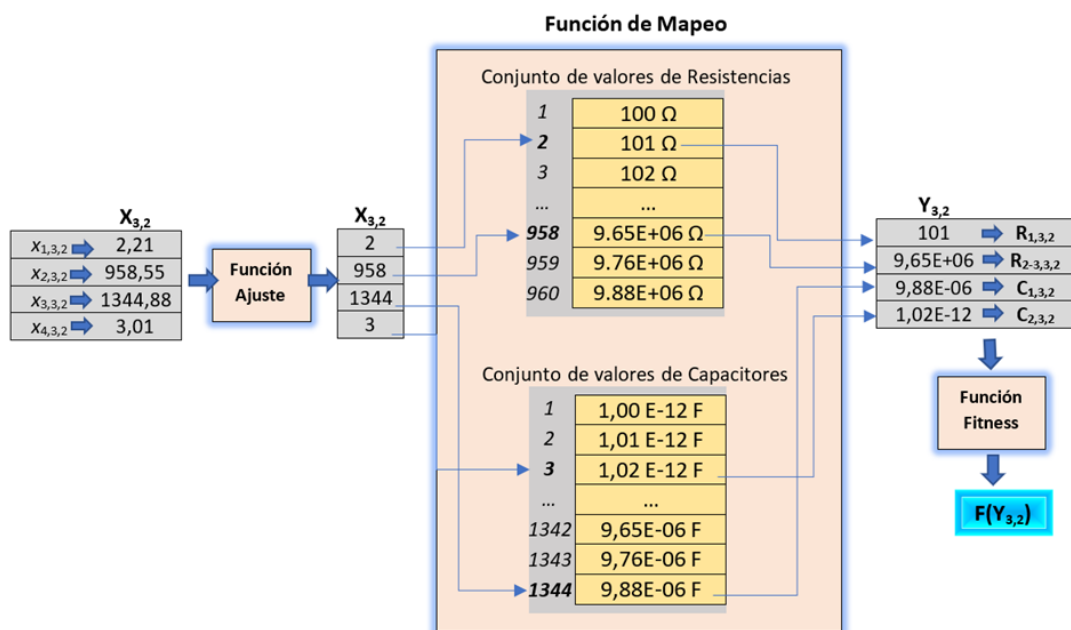


Fig. 3: Ejemplo de aplicación de las funciones de ajuste y mapeo implementadas por los algoritmos

2.3.1. Determinación de parámetros de los EAs

Para encontrar el conjunto de valores más apropiados de tamaño de población N y cantidad de generaciones M se realizan pruebas empleando diferentes combinaciones. Se adoptan valores desde 10 a 500, incrementados en pasos de 10 para N y para M . Para cada algoritmo y cada combinación de parámetros en cada etapa bicuadrática son efectuadas 30 corridas independientes. En estas corridas, se utiliza un conjunto diferente de semillas para inicializar la población, lo que incrementa la probabilidad de encontrar el óptimo global al explorar de manera más efectiva el espacio de búsqueda. Este enfoque asegura condiciones iniciales diversas y permite una evaluación más robusta de las combinaciones de parámetros evaluadas. Para realizar una comparación consistente con (Durmuş et al., 2020) se consideran aquellas combinaciones de N y M en las que la cantidad de Evaluaciones de la Función Objetivo (*EFO*) por etapa bicuadrática es menor o igual a 10000. Se persigue no sobrepasar las 50000 *EFO* para dimensionar el filtro, que es el valor utilizado en (Durmuş et al., 2020).

Cada algoritmo utiliza la misma combinación de valores de sus parámetros para dimensionar las cinco etapas del filtro. Para seleccionarlos se tiene en cuenta el menor valor de $Error_{Total}$ (10) alcanzado para cada combinación considerada. Por otro lado, cada algoritmo presenta parámetros particulares. Estos fueron obtenidos a partir de experimentos realizados tomando como base los usados en (Durmuş et al., 2020; Lovay et al., 2016, 2022). La tabla 2 resume los valores adoptados para los parámetros en función de la serie E.

EA	PSO			DE			ADE			EHO		
Serie E	E192	E96	E24	E192	E96	E24	E192	E96	E24	E192	E96	E24
M	140	380	330	480	400	420	500	370	220	20	20	20
N	60	10	20	10	20	20	20	20	30	490	260	490
EFO total	42000	19000	33000	24000	40000	42000	50000	37000	33000	49000	26000	49000
Parámetros particulares	$w_{min}=0,4; w_{max}=0,9$			F=0,7			F=2			$\alpha=0,8$		
	$C_1=2; C_2=2$			CR=0,5			CR=0,8			$\beta=0,1$		

Tabla 2: Valores adoptados para los parámetros de cada algoritmo en función de la serie E

3. Resultados y discusión

3.1. Filtro Chebyshev de 1dB

La tabla 3 muestra la caracterización estadística de los resultados obtenidos por cada EA en cada escenario propuesto. Se utiliza como métrica $Error_{Total}$ y el tiempo de ejecución. Para cada escenario y característica se resaltan en negrita los valores que corresponden al mínimo encontrado para los errores y los tiempos de ejecución.

Serie E	EA	Error _{total}				Tiempo de ejecución (seg)
		Menor	Media	Mayor	Desv. Estandar	
E192	PSO	7,03E-04	9,23E-03	2,26E-02	2,47E-03	0,2475
	DE	2,51E-04	2,01E-02	5,20E-01	8,96E-02	0,2965
	ADE	5,18E-04	1,19E-02	1,29E-01	1,44E-02	0,5893
	EHO	3,85E-04	4,19E-03	1,11E-02	1,12E-03	0,4847
E96	PSO	1,42E-03	0,29E+00	2,49E+00	3,43E-01	0,1426
	DE	6,98E-04	6,50E-03	2,31E-02	2,03E-03	0,4898
	ADE	1,20E-03	6,13E-02	1,03E+00	1,53E-01	0,4437
	EHO	1,30E-03	9,76E-03	2,46E-02	2,45E-03	0,2449
E24	PSO	2,69E-03	7,35E-02	0,54E+00	1,57E-01	0,2200
	DE	2,23E-03	1,69E-02	4,05E-02	4,56E-03	0,4905
	ADE	1,98E-03	2,92E-02	9,34E-02	9,79E-03	0,3336
	EHO	3,99E-03	3,30E-02	8,11E-02	9,12E-03	0,4171

Tabla 3: Caracterización de resultados para los cuatro EAs. Filtro Chebyshev de 1dB

Para la serie E192 se aprecia que, si bien DE encuentra la configuración con menor error, también se le atribuye la del mayor, presentando una media y desviación estándar más elevados que el resto de los algoritmos. Por otro lado, EHO es el que presenta los valores más bajos para estos tres parámetros, alcanzando también el segundo puesto con respecto al menor error. Para la serie E96, DE obtiene los errores más bajos, con una diferencia de hasta un orden de magnitud respecto al resto para el menor error. PSO es el que alcanza los peores valores en todos los casos. Para el escenario que adopta la serie E24, ADE alcanza el menor $Error_{total}$. No obstante, DE con un valor levemente más alto presenta a su vez los mejores valores para los tres parámetros restantes (mayor error, media, desviación estándar). El peor desempeño se evidencia para PSO también en este caso.

La tabla 3 también reporta los tiempos de ejecución como la suma del tiempo medio de cada etapa bicuadrática, utilizando una PC Intel Core i5-3317U, 1.70 GHz, 4.0 GB de RAM. EFO, PSO y EHO presentan valores más bajos en un orden de magnitud respecto a DE y ADE, para los tres escenarios propuestos.

Teniendo en cuenta las especificaciones de la tabla 1 se obtiene una función de transferencia a partir de (4), a la que denominamos filtro nominal. La respuesta en frecuencia del filtro nominal es utilizada para observar de manera gráfica el cumplimiento de las especificaciones por parte de las soluciones encontradas por los EAs. En la Fig. 4 se muestra la magnitud de la respuesta en frecuencia del filtro nominal y la respuesta de la mejor configuración de filtro alcanzada por cada EA, en los tres escenarios adoptados. Particularmente, en la Fig. 5 se visualiza la magnitud de la respuesta en frecuencia de cada filtro en la zona de la frecuencia de corte, clasificándolas de acuerdo con la serie E considerada. Se observa que la respuesta de los filtros obtenidos por cada EA y la respuesta del filtro nominal son muy similares, presentando una dispersión creciente desde E192 a E24.

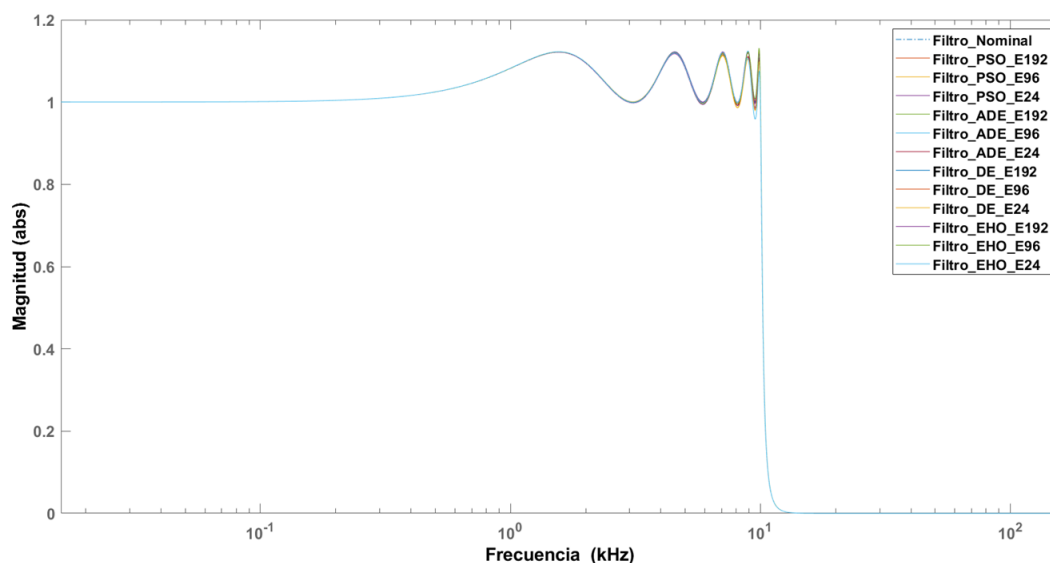


Fig. 4: Magnitud de la respuesta en frecuencia del filtro Chebyshev de 1 dB

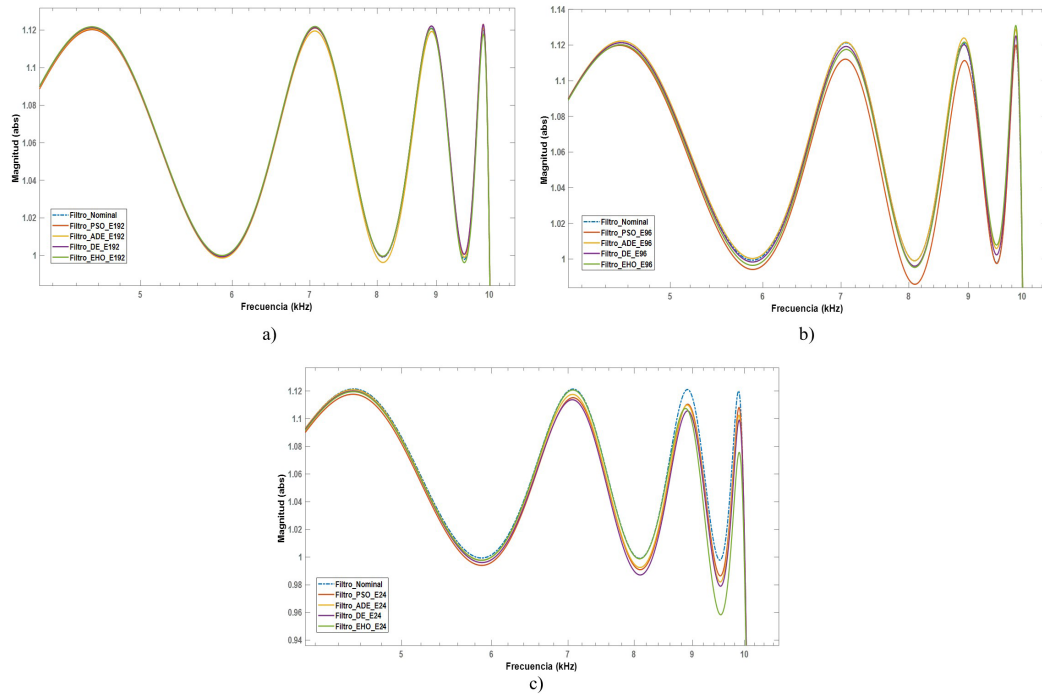


Fig. 5: Magnitud de la respuesta en frecuencia en la zona de la frecuencia de corte, filtro Chebyshev de 1dB. a) Serie E192. b) Serie E96. c) Serie E24

3.2. Filtro Chebyshev de 3dB

La tabla 4 presenta las estadísticas realizadas sobre los resultados alcanzados para $Error_{Total}$ y los tiempos de ejecución para cada escenario propuesto. Los errores y EFO presentan una tendencia similar respecto de los obtenidos para el filtro de 1dB.

Serie E	EA	Error _{total}				Tiempo de ejecución (seg)
		Menor	Media	Mayor	Desv. Estandar	
E192	PSO	4,21E-04	7,14E-03	2,05E-02	2,49E-03	0,3953
	DE	2,42E-04	1,50E-02	3,70E-01	6,49E-02	0,5316
	ADE	2,93E-03	2,37E-02	9,39E-02	9,44E-03	0,8644
	EHO	4,84E-04	3,73E-03	9,88E-03	1,10E-03	0,6866
E96	PSO	1,92E-03	5,66E-01	2,50E+00	4,41E-01	0,1878
	DE	1,80E-03	5,45E-03	1,55E-02	1,44E-03	0,7405
	ADE	2,42E-03	3,40E-02	6,25E-01	9,10E-02	0,6444
	EHO	2,38E-03	9,64E-03	2,57E-02	2,73E-03	0,3563
E24	PSO	6,48E-03	1,52E+00	2,87E-01	2,96E-01	0,2711
	DE	2,03E-03	4,09E-02	1,34E-02	4,26E-03	0,6448
	ADE	3,19E-03	5,60E-01	4,28E-02	9,05E-02	0,4861
	EHO	7,19E-03	3,48E-02	8,76E-02	9,36E-03	0,7270

Tabla 4: Caracterización de resultados para los cuatro EAs. Filtro Chebyshev de 3dB

En la Fig. 6 se muestra la magnitud de la respuesta en frecuencia del filtro nominal y la respuesta de la mejor configuración de filtro alcanzada por cada algoritmo en todos los escenarios. Para facilitar la visualización del desempeño de cada EA en cada caso, se presentan en la Fig. 7 las respuestas en frecuencia de cada filtro en la zona de la frecuencia de corte, agrupadas para cada serie E considerada. Las figuras evidencian que las respuestas son similares a la del filtro nominal, siendo la serie E192 la que presenta los mejores resultados. Para las series restantes, a pesar de emplear componentes de menor precisión, se observan diferencias leves.

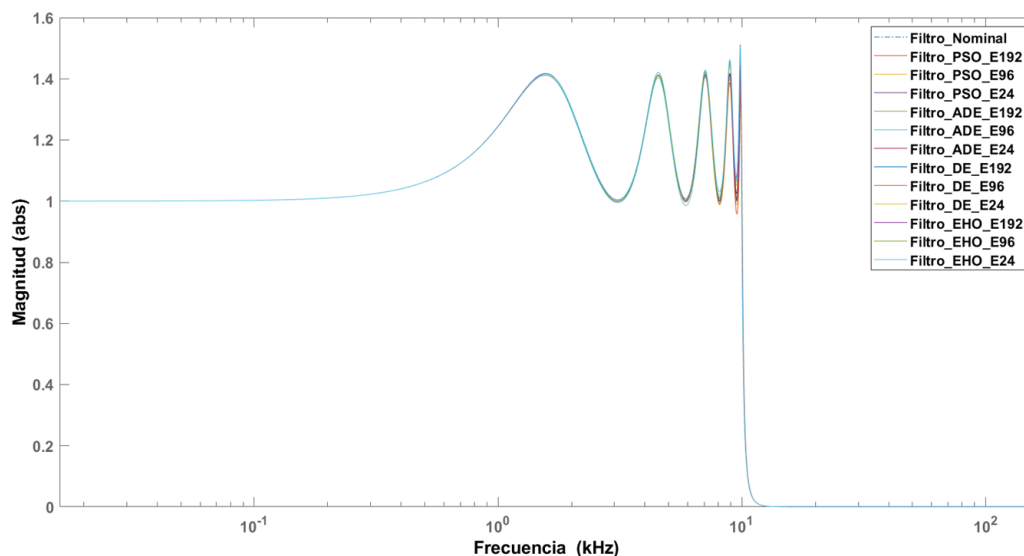


Fig. 6: Magnitud de la respuesta en frecuencia del filtro Chebyshev de 3dB.

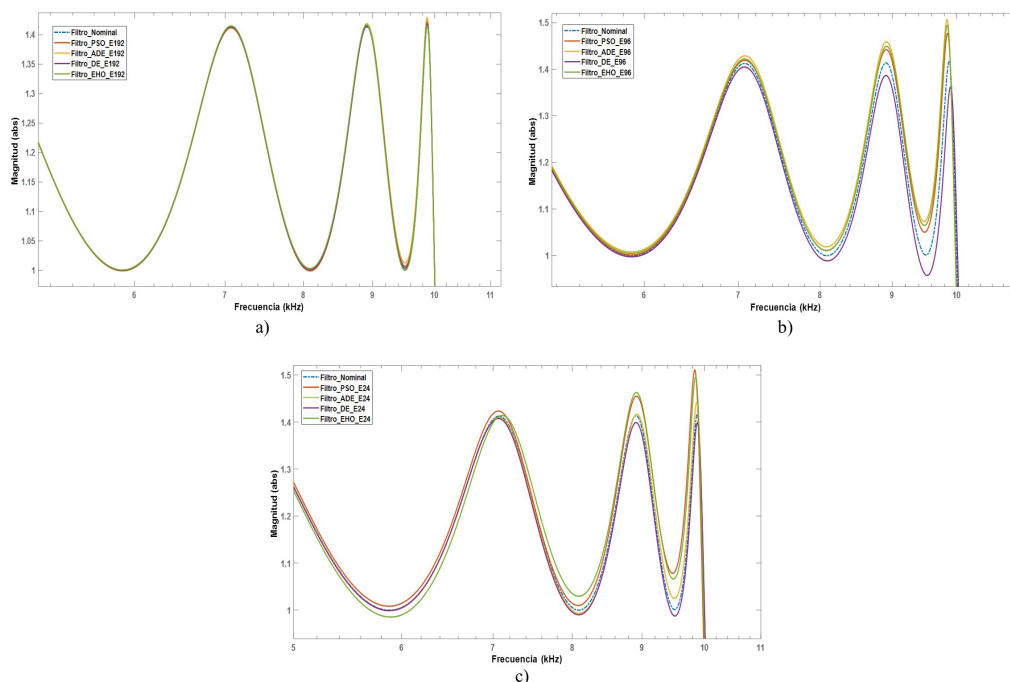


Fig. 7: Magnitud de la respuesta en frecuencia en la zona de la frecuencia de corte, filtro Chebyshev de 3dB. a) Serie E192. b) Serie E96. c) Serie E24

3.3. Caracterización estadística de los EAs

La comparación cuantitativa del desempeño de los EAs se realiza utilizando el test de Friedman (Demšar, 2006; García et al., 2009) para establecer si hay diferencias estadísticamente significativas entre los cuatro EAs, tomando como métrica el valor de $Error_{Total}$. Esta prueba realiza comparaciones múltiples y no asume normalidad en los datos (no paramétrico), con hipótesis nula que considera la igualdad de las medianas entre las poblaciones. La hipótesis alternativa es la negación de la hipótesis nula, por lo que posteriormente de confirmarse la hipótesis alternativa, debe establecerse una comparación adicional para establecer la combinación ganadora (García et al., 2009).

El test de Friedman proporciona un valor de ranking medio para cada EA. Se trata de una ponderación que permite ordenarlos de acuerdo con su desempeño, asignando un valor de ranking menor para los de mejor desempeño (Fig. 8). Para nuestro estudio, el desempeño es el menor $Error_{Total}$ obtenido por cada EA para cada filtro en cada escenario (columna 3 de las tablas 3 y 4). El valor obtenido para p es 0,0141, inferior al nivel de significancia considerado (0,05), por lo que se rechaza la hipótesis nula con un nivel de confianza del 95%, y se considera que existen diferencias significativas entre los resultados arrojados por cada EA.

Se complementa el análisis aplicando la prueba de Bonferroni-Dunn (Demšar, 2006; García et al., 2009) para detectar diferencias significativas respecto de DE, que es el EA que alcanza el mejor ranking, asumido como algoritmo de control. Empleando este test es calculado el valor de Diferencia Crítica (DC) con un nivel de significancia de 0,05 y 0,10. Aquellos EAs cuyo ranking medio es superior al mejor ranking en al menos la DC presentan un desempeño que se considera significativamente inferior que DE.

La DC de Bonferroni-Dunn se grafica en la Fig. 8, como una línea de puntos vertical trazada para cada nivel de significancia considerado en el estudio (línea naranja: DC=1,9202 con =0,05, línea verde: DC=1,7338 con =0,10). Cada línea se traza en base a la suma entre el valor de ranking del algoritmo de control y el valor de DC. Los EAs cuyo ranking se encuentra por encima de la línea de puntos vertical se consideran de peor desempeño que el algoritmo de control. La figura evidencia que, con referencia al menor $Error_{Total}$, EHO presenta un desempeño significativamente inferior a DE, para ambos niveles de significancia. Por otro lado, PSO tiene una diferencia estadísticamente significativa respecto del algoritmo de control para el 90% de confianza, pero no es posible mantener esta afirmación para el 95%. Con referencia a ADE, es posible concluir que no presenta diferencias significativas respecto de DE, para el aspecto evaluado.

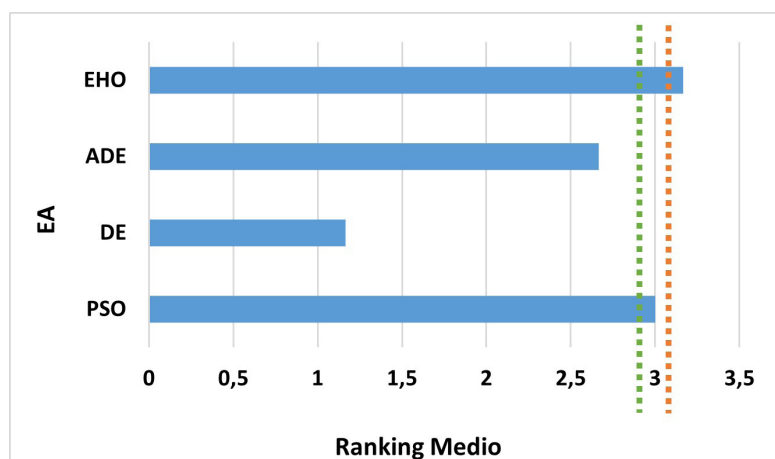


Fig. 8: Resultados de test de Friedman y prueba de Bonferroni-Dunn

Para tener una aproximación acerca de la eficiencia computacional de cada algoritmo, se presenta en la Fig. 9 la cantidad promedio de EFO que realiza cada algoritmo en 1 segundo. Teniendo en cuenta este criterio, se observa que PSO es el más eficiente, luego le sigue EHO y finalmente ADE y DE, respectivamente.

De los resultados se concluye que DE y ADE no evidencian diferencias estadísticamente significativas entre sí respecto al menor $Error_{Total}$, no obstante, demandan mayor costo computacional que el resto.

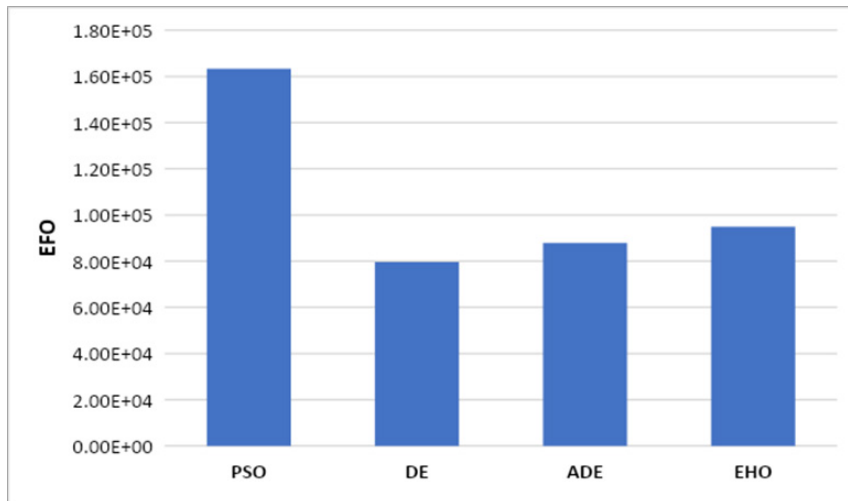


Fig. 9: Cantidad promedio de EFO que realiza cada algoritmo en 1 segundo.

3.4. Comparación con trabajos previos

Esta sección compara los resultados alcanzados para ambos filtros en todos los escenarios propuestos con los obtenidos en (Durmuş et al., 2020). Este último artículo realiza el diseño del filtro bajo estudio utilizando PSO, CSS, DE y ADE empleando únicamente la serie E192 mediante un enfoque de diseño global. Las Fig. 10 y 11 presentan el menor $Error_{Total}$ alcanzado por cada EA para el filtro de 1 dB y 3 dB, respectivamente.

La mejor solución encontrada por cualquiera de los cuatro EAs presenta siempre un error total menor al alcanzado en (Durmuş et al., 2020), con diferencias significativas. Particularmente, si se compara el desempeño de PSO, DE y ADE, algoritmos empleados tanto en el presente trabajo como en el referenciado, se observa que cada uno de ellos encuentra soluciones con un error de diseño considerablemente inferior cuando se aplica como parte del método propuesto. La diferencia más notable se presenta para DE en el filtro de 3 dB, donde la mejor configuración tiene un error $2,42E-04$, a diferencia de $1,84E-02$ que es el error de (Durmuş et al., 2020). Por otro lado, es destacable que los cuatro EAs obtienen errores más bajos que en (Durmuş et al., 2020), aun utilizando series de mayor tolerancia.

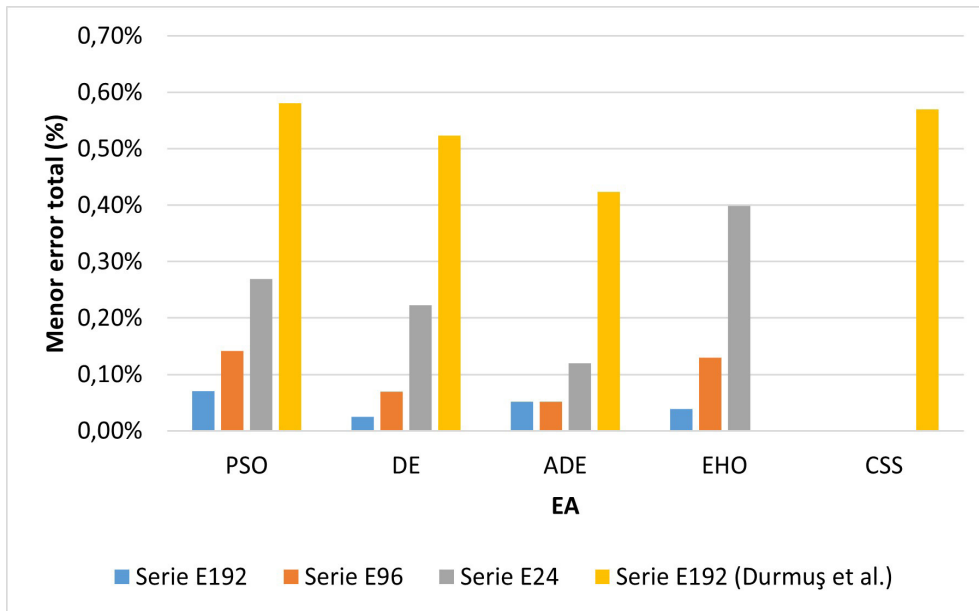


Fig. 10: Menor $Error_{Total}$, filtro Chebyshev de 1dB

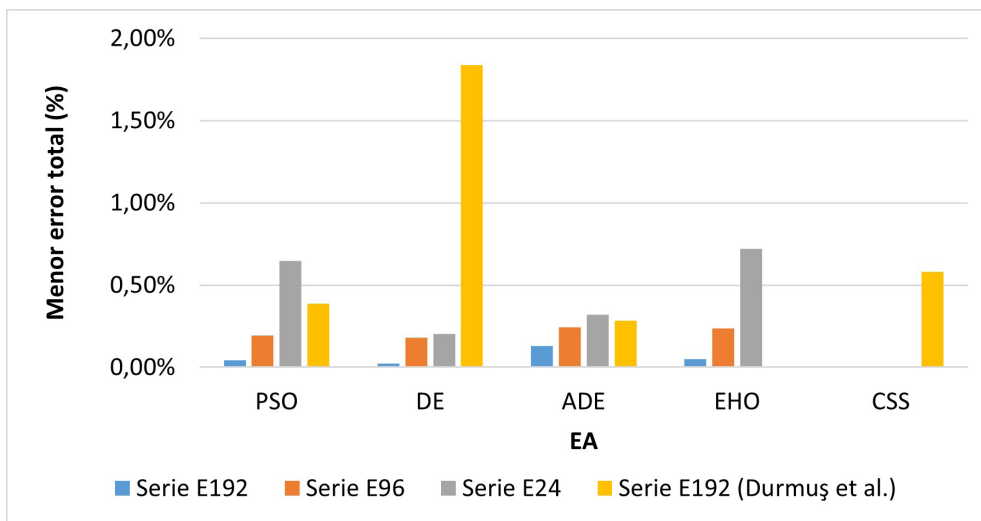


Fig. 11: Menor $Error_{Total}$, filtro Chebyshev de 3dB

En la Fig. 12 se presenta la respuesta en frecuencia del mejor y peor diseño para el filtro de 1dB en términos de $Error_{Total}$, donde el mejor es alcanzado por DE con la serie E192 y el peor corresponde a EHO con la serie de menos precisión, E24. También se grafica la respuesta de la mejor y peor configuración encontradas en (Durmuş et al., 2020) (la mejor corresponde a ADE y la peor a PSO). Se observa que las dos configuraciones de nuestro trabajo presentan respuestas muy parecidas a la del filtro nominal, mientras que las alcanzadas en (Durmuş et al., 2020) difieren considerablemente.

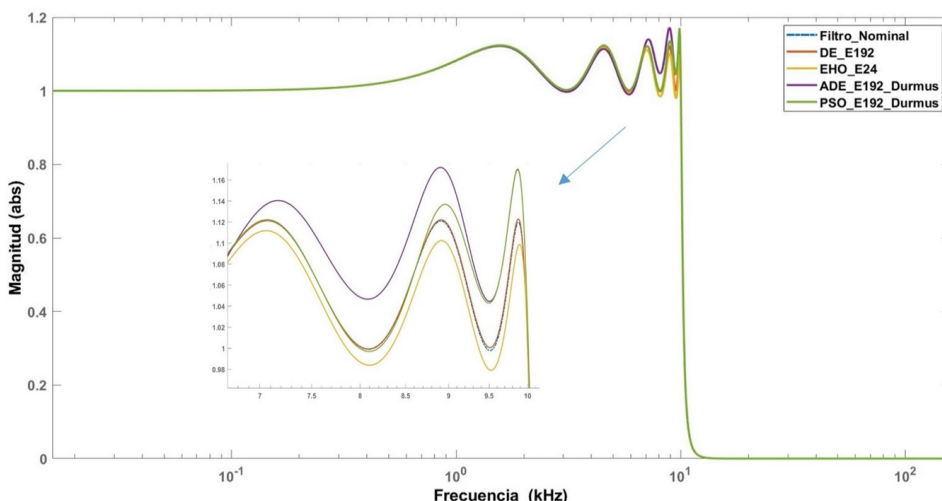


Fig. 12: Magnitud de la respuesta en frecuencia del filtro nominal y del mejor y peor filtro obtenidos en este trabajo y en (Durmuş et al., 2020), filtro Chebyshev de 1dB

Por otro lado, en la Fig. 13 se realiza la misma comparación para el filtro de 3dB. Los filtros con menor $Error_{Total}$ presentan respuestas muy similares a la del nominal. Para el caso del peor filtro, tanto la configuración obtenida por EHO con la serie E24 como la alcanzada por DE para la serie E192 en (Durmuş et al., 2020) presentan diferencias con la nominal. No obstante, se debe tener en cuenta que el peor filtro de este trabajo considera componentes con menor precisión, una ventaja para el método de diseño modular propuesto.

Por otra parte, los resultados alcanzados por cada EA se obtienen empleando menos cantidad de EFO o en el peor de los casos la misma cantidad de EFO (tabla 2) que en el trabajo previo referenciado (50000). Particularmente, el algoritmo DE que es el que alcanza la configuración de filtro con el error de diseño más bajo para ambos filtros, emplea tan solo 24000 EFO, es decir, 52% menos que en (Durmuş et al., 2020).

Con referencia a los tiempos de ejecución, no es posible realizar una comparación, debido a que no se posee información acerca de las características del equipo empleado para ejecutar estos algoritmos en (Durmuş et al., 2020).

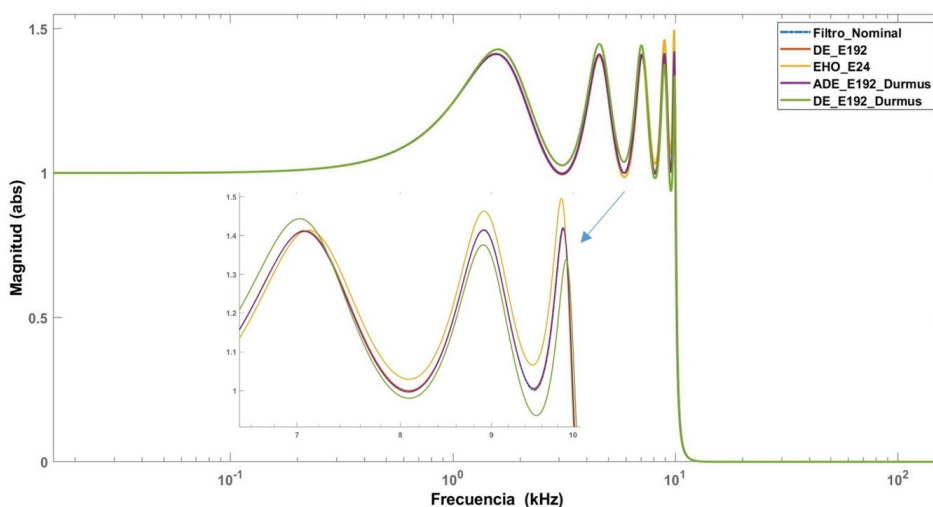


Fig. 13: Magnitud de la respuesta en frecuencia del filtro nominal y del mejor y peor filtro obtenidos en este trabajo y en (Durmuş et al., 2020), filtro Chebyshev de 3dB

3.5. Aplicación del método en otros casos de estudio

Adicionalmente, se tomó como referencia el trabajo realizado en (Kuyu & Vatansver, 2023) donde se dimensionan filtros Butterworth pasabajo de alto orden.

La tabla 5 presenta algunos resultados exploratorios alcanzados con PSO para filtros de sexto y octavo orden considerando las series E96 y E24. Se muestran las características del $Error_{Total}$ y la cantidad de EFO (como métrica de tiempo de desempeño, ya que se asume que a mayor número de EFO, mayor es el tiempo de ejecución) para PSO y para HBO-CO, que es el mejor algoritmo reportado en (Kuyu & Vatansver, 2023). Se aprecia que el menor error total alcanza valores considerablemente más bajos para PSO con la estrategia de dimensionamiento propuesta, respecto a los de HBO-CO con el enfoque global. Por otro lado, la cantidad de EFO es mucho menor para PSO en todos los casos, llegando a ser hasta 27 veces inferior.

Orden del Filtro	Serie E	EA	$Error_{Total}$			EFO
			Menor (%)	Media (%)	Desv. Estándar (%)	
Sexto Orden	E24	HBO-CO	0,42	0,50	0,06	1348100
		PSO	0,28	0,84	0,26	90000
	E96	HBO-CO	0,03	0,05	0,03	2415450
		PSO	0,01	0,04	0,02	90000
Octavo Orden	E24	HBO-CO	1,10	1,63	0,31	347450
		PSO	0,77	1,73	0,56	90000
	E96	HBO-CO	0,10	0,18	0,07	1947450
		PSO	0,06	0,13	0,03	90000

Tabla 5: Comparación de desempeño entre PSO y HBO-CO (Kuyu & Vatansver, 2023) para filtros Butterworth pasabajo

La respuesta en frecuencia de los filtros obtenidos en este trabajo con PSO y las alcanzadas con HBO-CO en (Kuyu & Vatansver, 2023) para ambas series E se visualizan en las Fig. 14 y 15, para el filtro de sexto y octavo orden, respectivamente, junto a la respuesta del filtro nominal. Para los dos filtros, las respuestas son similares, a excepción de las obtenidas mediante HBO_CO para la serie E24, que evidencian una diferencia más significativa con el resto. En cambio, para esta misma serie, PSO alcanza en cada caso una configuración cuya respuesta difiere levemente de la respuesta del filtro nominal, lo cual se traduce en una ventaja para el enfoque modular propuesto.

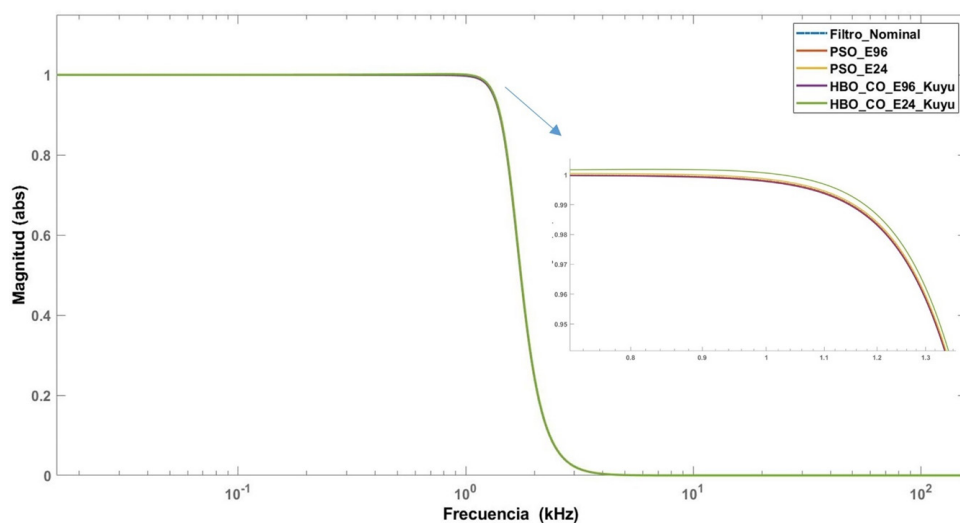


Fig. 14: Magnitud de la respuesta en frecuencia. Filtro Butterworth de sexto orden

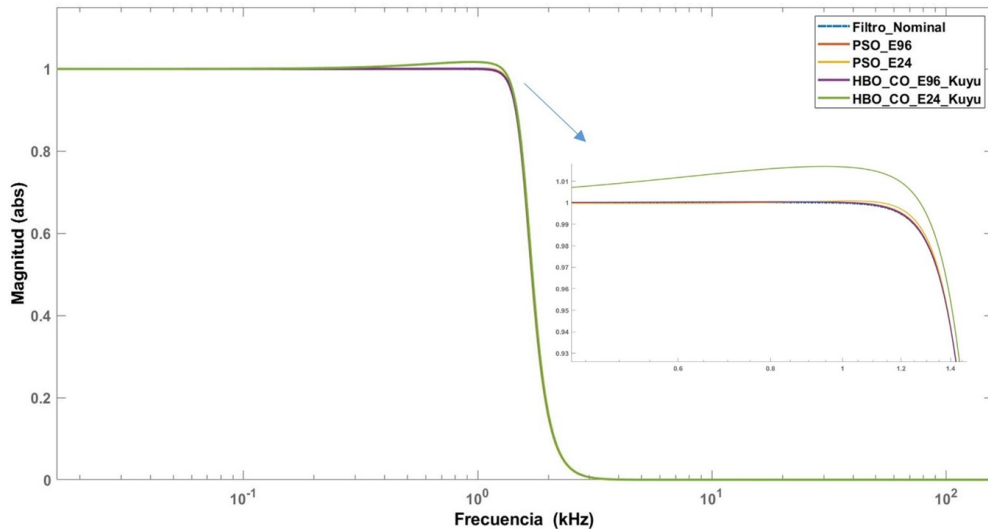


Fig. 15: Magnitud de la respuesta en frecuencia. Filtro Butterworth de octavo orden

4. Conclusiones

Este trabajo formula una estrategia de dimensionamiento de componentes pasivos para filtros de alto orden implementados en cascada. La misma tiene dos características importantes, la primera de ellas es la aplicación de EAs para seleccionar los valores de las respectivas resistencias y capacitores. La segunda característica es la modularidad, debido a que se aplica individualmente a cada etapa bicuadrática del filtro.

Para los casos de estudio abordados se observa que los cuatro EAs adoptados alcanzan muy buenos resultados, siendo DE y ADE estadísticamente mejores en términos del menor valor de $Error_{total}$, mientras que PSO presenta mejores tiempos de ejecución. La comparación con trabajos previos que usan otros métodos de dimensionamiento muestra que la estrategia propuesta alcanza un mejor desempeño con un costo computacional más bajo, aún con componentes de mayor tolerancia.

Referencias

- Beşikli, M., & Kiran, M. S. (2023). Optimization of Butterworth and Bessel Filter Parameters with Improved Tree-Seed Algorithm. *Biomimetics*, 8(7). <https://doi.org/10.3390/biomimetics8070540>
- Deliyannis, T., Sun, Y., & Fidler, J. (1999). *Continuous-Time Active Filter Design* CRC 1999.
- Demšar, J. (2006). Statistical Comparisons of Classifiers over Multiple Data Sets. *Journal of Machine Learning Research*, 7, 1–30.
- Durmuş, B., Temurtaş, H., & Özyön, S. (2020). The design of multiple feedback topology Chebyshev low-pass active filter with average differential evolution algorithm. *Neural Computing and Applications*, 32(22), 17097–17113. <https://doi.org/10.1007/s00521-020-04922-7>
- García, S., Molina, D., Lozano, M., & Herrera, F. (2009). A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: A case study on the CEC'2005 Special Session on Real Parameter Optimization. *Journal of Heuristics*, 15(6), 617–644. <https://doi.org/10.1007/s10732-008-9080-4>
- Hiçdurmaz, B., Ertaç Durak, F., Özyön, S., Dumlupınar Üniversitesi, K., Fakültesi, M., & Elektronik Mühendisliği, E. (2019). The Estimation of Bessel Type Low-Pass Active Filter Parameters with Charged System Search Algorithm. *International Scientific and Vocational Studies Journal*, 3(2), 67–75.
- Jin, Y. X., Cheng, H. Z., Yan, J. Y., & Zhang, L. (2007). New discrete method for particle swarm optimization and its application in transmission network expansion planning. *Electric Power Systems Research*, 77(3–4), 227–233. <https://doi.org/10.1016/j.epsr.2006.02.016>
- Karki, J. (2023). *Nota de Aplicación: Active Low-Pass Filter Design*. www.ti.com
- Kuyu, Y. Ç., & Vatansever, F. (2023). Heap-based optimizer embedded with search strategies applied to high-order analog filter designs: a comparative study with up-to-date metaheuristics. *Neural Computing and Applications*, 35(2), 1447–1467. <https://doi.org/10.1007/s00521-022-07835-9>
- Lampinen, J., & Zelinka, I. (1999). Mixed integer-discrete-continuous optimization by differential evolution. In *Proceedings of the 5th International Conference on Soft Computing*, 71–76.
- Lovay, M., Peretti, G., & Romero, E. (2022). Diseño de filtros activos basados en optimización de cría de elefantes. *Jornadas de Ciencia y Tecnología*.
- Lovay, M., Romero, E., & Peretti, G. (2016). Aplicación del algoritmo de Optimización por Enjambre de Partículas en el dimensionamiento óptimo de componentes para Filtros Activos. *Proceedings of SII 2016, 5th Argentine Symposium on Industrial Informatics, 45 JAIIO - 45th Argentine Conference on Informatics*, 13–24.
- Raut R., & Swamy M. N. S. (2010). *Modern Analog Filter Analysis and Design*. Wiley.

Şimşir, Ş., & Taşpınar, N. (2021). A novel discrete elephant herding optimization-based PTS scheme to reduce the PAPR of universal filtered multicarrier signal. *Engineering Science and Technology, an International Journal*, 24(6), 1428–1441. <https://doi.org/10.1016/j.jestch.2021.03.001>

Slowik, A. (2020). *Swarm Intelligence Algorithms*. CRC Press.

Temurtaş, H. (2020). The estimation of low and high-pass active filter parameters with opposite charged system search algorithm. *Expert Systems with Applications*, 155. <https://doi.org/10.1016/j.eswa.2020.113474>

Yu X., & Gen M. (2010). *Introduction to Evolutionary Algorithms*. Springer.

Zumbahlen, H. (2007). *Nota de Aplicación: Basic linear design, analog devices* (Inc.: Norwood, Ed.; Vol. 11).

Contribución de los Autores

Nombres y Apellidos del autor	Colaboración Académica													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Mónica Lovay			X	X		X	X	X	X	X	X		X	X
Gabriela Peretti	X	X		X	X	X	X	X	X	X	X	X	X	X
Eduardo Romero	X	X		X	X	X	X	X	X			X	X	X

1-Administración del proyecto, **2**-Adquisición de fondos, **3**-Análisis formal, **4**-Conceptualización, **5**-Curaduría de datos, **6**-Escritura - revisión y edición, **7**-Investigación, **8**-Metodología, **9**-Recursos, **10**-Redacción - borrador original, **11**-Software, **12**-Supervisión, **13**-Validación, **14**-Visualización.