UNIVERSIDAD DIEGO PORTALES

FACULTAD DE INGENIERÍA Y CIENCIAS

ESCUELA DE INFORMÁTICA Y TELECOMUNICACIONES

Algoritmos Exactos y Metaheurísticas

Tarea 3

Profesor:
Víctor Reyes

Estudiante:
Felipe Méndez

Santiago, Chile 5 de junio del 2025

Índice

1. Introducción

La optimización de funciones busca encontrar los valores de las variables de entrada que minimizan o maximizan el valor de una función objetivo. Las funciones multimodales son desafiantes, estas son aquellas que poseen múltiples mínimos (o máximos) locales además de un único mínimo (o máximo) global. Métodos de optimización tradicionales basados en gradientes, como el descenso de gradiente, suelen quedar atrapados en mínimos locales, impidiendo la convergencia hacia el óptimo global deseado.

Ante este desafío, las metaheurísticas basadas en población han surgido como una gran alternativa. Estos algoritmos exploran el espacio de búsqueda utilizando múltiples puntos simultáneamente, lo que les permite tener una mayor capacidad para escapar de mínimos locales y explorar de manera más global el espacio de soluciones.

En esta tarea, se aborda la optimización de cuatro funciones multimodales continuas específicas utilizando una metaheurística basada en población: el algoritmo de Optimización por Enjambre de Partículas (PSO). Se implementa el algoritmo y se realizan experimentos para evaluar su rendimiento y la influencia de diferentes configuraciones de sus parámetros en la capacidad de encontrar el mínimo de cada función.

2. Diseño e Implementación

2.1. Justificación de la Elección del Algoritmo

Para abordar el problema, se seleccionó el algoritmo de Optimización por Enjambre de Partículas (PSO). A continuación, se enseñan las razones:

1. **Espacios Continuos:** Este algoritmo fue diseñado específicamente para problemas de optimización en dominios continuos. Sus operaciones fundamentales (actualización de velocidad y posición) operan directamente sobre vectores de números reales, lo que lo hace intrínsecamente adecuado para las funciones objetivo dadas. Otros algoritmos como por ejemplo, ACO, están naturalmente orientados a problemas discretos y requerirían adaptaciones más complejas para ser aplicados eficientemente en este problema.

2. **Multimodalidad:** Como metaheurística basada en población, PSO mantiene y evoluciona múltiples soluciones (partículas) simultáneamente en el espacio de búsqueda. Esta exploración concurrente desde diversos puntos reduce la probabilidad de que todo el enjambre quede atrapado en un único mínimo local, esto ayuda bastante para tratar con funciones multimodales que poseen múltiples óptimos subóptimos.

3. **Exploración y Explotación:** El mecanismo de actualización de partículas de PSO, influenciado por la inercia (w), la mejor experiencia individual (c_1) y la mejor experiencia global del enjambre (c_2), proporciona un balance efectivo entre la exploración de nuevas regiones del espacio de búsqueda y la explotación de áreas prometedoras ya identificadas. Esta característica permite, dependiendo de los valores de las constantes anteriores, explorar o explotar.

4. **Simplicidad de Implementación:** Comparado con los otros algoritmos posibles como Genéticos (GA) o Evolución Diferencial (DE), la estructura básica de PSO y sus ecuaciones de actualización son relativamente simples de implementar. Esto permite concentrarse más en la experimentación y el análisis del rendimiento paramétrico. Si bien GA y DE también son potentes para optimización continua y podrían haber sido opciones interesantes, PSO ofrece una buena combinación de efectividad y accesibilidad para resolver este problema.

2.2. Principios de PSO

Inspirado en el comportamiento social de especies como bandadas de aves y cardúmenes de peces, donde la cooperación guía la búsqueda de alimento, PSO opera con un enjambre de soluciones candidatas, llamadas partículas.

Cada partícula i en este enjambre mantiene la siguiente información:

- Una posición actual x_i en el espacio de búsqueda, que representa una solución candidata.
- Una velocidad v_i que determina la dirección y magnitud de su movimiento en la siguiente iteración.
- Su mejor posición histórica personal, $pbest_i$, que es la posición donde la partícula i ha encontrado el mejor valor (fitness) de la función objetivo hasta el momento.

Adicionalmente, el enjambre, en su variante global, mantiene un registro de la mejor posición encontrada por cualquier partícula en la población hasta la iteración actual. Esta posición es el *gbest*, que corresponde al mínimo global actual conocido por el enjambre.

En cada iteración t, la velocidad y la posición de cada partícula se actualizan basándose en las siguientes ecuaciones de actualización:

$$v_i(t+1) = w \cdot v_i(t) + c_1 \cdot r_1 \cdot (pbest_i - x_i(t)) + c_2 \cdot r_2 \cdot (gbest - x_i(t))$$
$$x_i(t+1) = x_i(t) + v_i(t+1)$$

Este mecanismo permite un equilibrio dinámico entre la *diversificación*, facilitada por la inercia y el componente cognitivo, y la *intensificación*, principalmente influenciada por el componente social.

Para asegurar que las partículas permanezcan dentro de los límites del dominio de búsqueda definido para cada función objetivo, se aplica una estrategia de "clamping". Esto implica ajustar la posición de la partícula a los límites del dominio si su actualización la lleva fuera, y a menudo también limitar su velocidad máxima para controlar el tamaño de los pasos.

2.3. Pseudocódigo

procedure PSO(objective_func, grado, bounds, num_particulas, max_iter, w, c1, c2, max_vel_ratio)
  $lower\_bound, upper\_bound \leftarrow bounds$
  $max\_velocity \leftarrow max\_vel\_ratio \times (upper\_bound - lower\_bound)$
  Inicializar POSICIONES(num_particulas, grado, lower_bound, upper_bound) aleatoriamente
  Inicializar VELOCIDADES(num_particulas, grado, -max_velocity, max_velocity) aleatoriamente
  $pbest\_positions \leftarrow positions$
  Calcular $pbest\_values$ evaluando $objective\_func$ en cada $pbest\_positions$
  Encontrar el índice $gbest\_index$ de la mejor partícula en $pbest\_values$
  $gbest\_position \leftarrow pbest\_positions[gbest\_index]$
  $gbest\_value \leftarrow pbest\_values[gbest\_index]$
  $history \leftarrow [gbest\_value]$
  for $iter\_count \leftarrow 0$ to $max\_iter - 1$ **do**
    $current\_w \leftarrow w(iter\_count, max\_iter)$ si w es una función, sino w
    for $i \leftarrow 0$ to $num\_particulas - 1$ **do**
      $r_1 \leftarrow$ VECTORALEATORIO($grado$) (uniforme en [0,1])
      $r_2 \leftarrow$ VECTORALEATORIO($grado$) (uniforme en [0,1])
      $cognitive\_velocity \leftarrow c_1 \times r_1 \times (pbest\_positions[i] - positions[i])$
      $social\_velocity \leftarrow c_2 \times r_2 \times (gbest\_position - positions[i])$
      $velocities[i] \leftarrow current\_w \times velocities[i] + cognitive\_velocity + social\_velocity$
      Limitar $velocities[i]$ al rango $[-max\_velocity, max\_velocity]$ (Clamping de velocidad)
      $positions[i] \leftarrow positions[i] + velocities[i]$
      Limitar $positions[i]$ al rango $[lower\_bound, upper\_bound]$ (Clamping de posición)
      $current\_value \leftarrow objective\_func(positions[i])$
      if $current\_value < pbest\_values[i]$ **then**
        $pbest\_values[i] \leftarrow current\_value$
        $pbest\_positions[i] \leftarrow positions[i]$
      end if
    end for
    Encontrar el índice $best\_pbest\_index$ de la mejor partícula en $pbest\_values$
    if $pbest\_values[best\_pbest\_index] < gbest\_value$ **then**
      $gbest\_value \leftarrow pbest\_values[best\_pbest\_index]$
      $gbest\_position \leftarrow pbest\_positions[best\_pbest\_index]$
    end if
    Agregar $gbest\_value$ a $history$
  end for
  Return $gbest\_position, gbest\_value, history$
end procedure

2.4. Funciones Objetivo

Se consideran las siguientes cuatro funciones:

1. $f_1(x) = 4 - 4x_1^3 - 4x_1 + x_2^2$
   Dominio: $x_i \in [-5, 5]$.

2. $f_2(x) = \frac{1}{899} \sum_{i=1}^{6} x_i^2 2^i - 1745$
   Dominio: $x_i \in [0, 1]$.

3. $f_3(x) = (x_1^6 + x_2^4 - 17)^2 + (2x_1 + x_2 - 4)^2$
   Dominio: $x_i \in [-500, 500]$.

4. $f_4(x) = \sum_{i=1}^{10} [(\ln(x_i - 2))^2 + (\ln(10 - x_i))^2] - \left(\prod_{i=1}^{10} x_i\right)^{0,2}$
   Dominio: $-2{,}001 < x_i \leq 10$.

3. Configuración Experimental

Para cada una de las cuatro funciones objetivo, se realizaron 10 ejecuciones independientes. Para evaluar la influencia de los parámetros de PSO, se definieron y probaron cuatro configuraciones distintas:

1. **Config 1 (Estándar):** $w = 0{,}8, c_1 = 2{,}0, c_2 = 2{,}0$. Parámetros comúnmente utilizados, buscan un balance entre exploración y explotación.
2. **Config 2 (Más Exploratoria):** $w = 0{,}9, c_1 = 2{,}5, c_2 = 1{,}5$. Mayor inercia y mayor peso en el componente cognitivo ($c_1 > c_2$), incentivando a las partículas a explorar más lejos de la mejor posición global conocida.
3. **Config 3 (Más Explotadora):** $w = 0{,}7, c_1 = 1{,}5, c_2 = 2{,}5$. Menor inercia y mayor peso en el componente social ($c_2 > c_1$), impulsando a las partículas a converger más rápidamente hacia la mejor posición global del enjambre.
4. **Config 4 (Inercia Decreciente):** $c_1 = 2{,}0, c_2 = 2{,}0$, con w decreciendo linealmente de 0.9 a 0.4 a lo largo de las iteraciones. Esta estrategia busca combinar una alta exploración al principio (con w alto) con una mayor explotación al final (con w bajo), manteniendo un balance constante entre c_1 y c_2.

Todas las configuraciones utilizaron 50 partículas y 1000 iteraciones.

Para cada función y configuración, se registró el mejor valor de fitness encontrado en cada una de las 10 ejecuciones, el valor medio de esos mejores fitness al final de las ejecuciones, su desviación estándar y el historial de convergencia promedio (promedio del *gbest_value* a través de las 10 ejecuciones en cada iteración).

4. Resultados Experimentales

A continuación, se presentan los resultados promedio y la desviación estándar del mejor valor encontrado para cada función tras 10 ejecuciones y bajo las diferentes configuraciones de parámetros.

4.1. Resultados

Función	Configuración	Mejor Valor Promedio	Desviación Estándar	Tiempo (10 ejec.)
f1	Estándar	-516.000000	0.000000	7.59s
	Más Exploratoria	-516.000000	0.000000	7.56s
	Más Explotadora	-516.000000	0.000000	7.53s
	W Decreciente	-516.000000	0.000000	7.56s
f2	Estándar	-1745.000000	0.000000	10.86s
	Más Exploratoria	-1745.000000	0.000000	10.79s
	Más Explotadora	-1745.000000	0.000000	10.81s
	W Decreciente	-1745.000000	0.000000	10.81s
f3	Estándar	0.000000	0.000000	7.66s
	Más Exploratoria	0.824449	0.943374	7.66s
	Más Explotadora	0.000000	0.000000	7.81s
	W Decreciente	0.000000	0.000000	8.20s
f4	Estándar	-45.773321	0.003848	16.70s
	Más Exploratoria	-45.311590	0.118431	17.06s
	Más Explotadora	-45.778470	0.000000	17.21s
	W Decreciente	-45.778470	0.000000	16.30s

Cuadro 1: Resumen de Resultados: Mejor Valor Promedio y Desviación Estándar (± Desv Est)

4.2. Gráficos de Convergencia

Los siguientes gráficos muestran el progreso promedio del mejor valor encontrado (gbest) a lo largo de las 1000 iteraciones para cada función y configuración paramétrica

Nota sobre la escala del eje Y: Debido a que los valores mínimos encontrados para f1, f2, f3 (mínimo 0) y f4 son negativos o cero, la escala logarítmica en el eje Y no es aplicable para la totalidad de los valores. Los gráficos presentados utilizan una escala lineal en el eje Y, lo cual nos sirve igualmente para visualizar la convergencia.

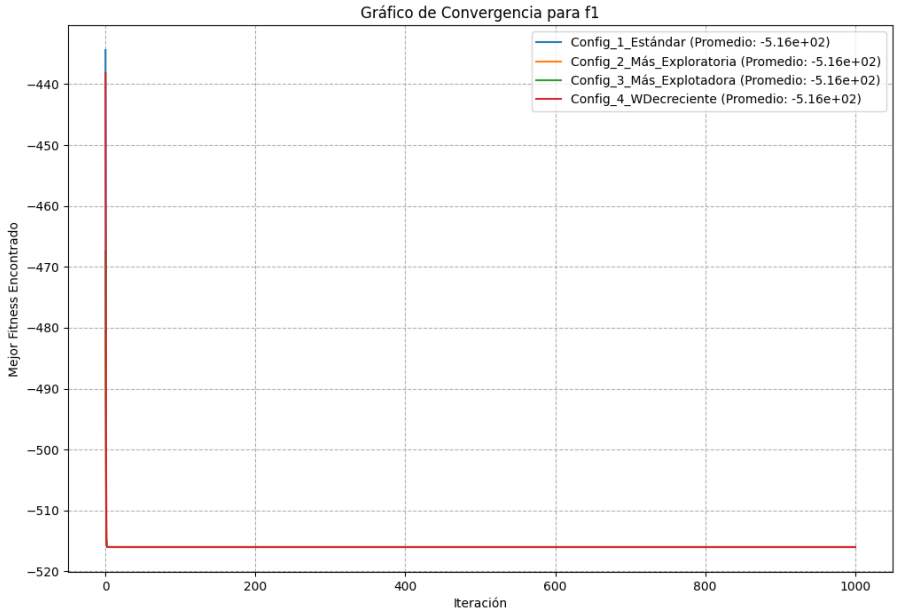

Figura 1: Gráfico de Convergencia para f1

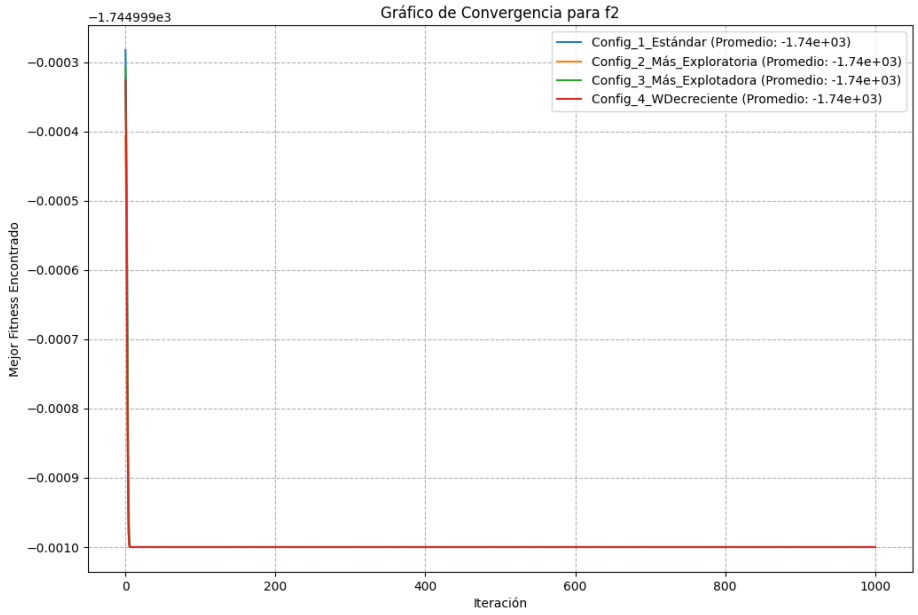

Figura 2: Gráfico de Convergencia para f2

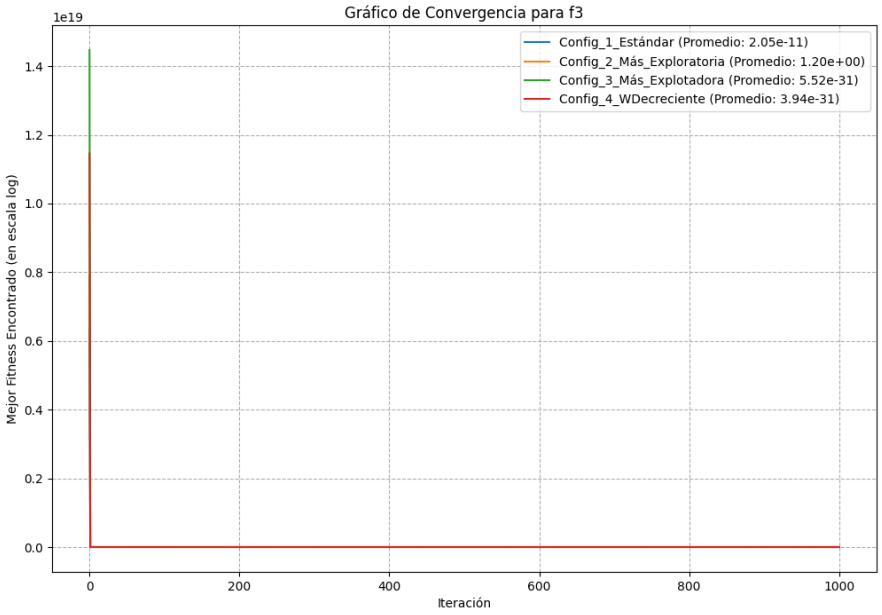

Figura 3: Gráfico de Convergencia para f3

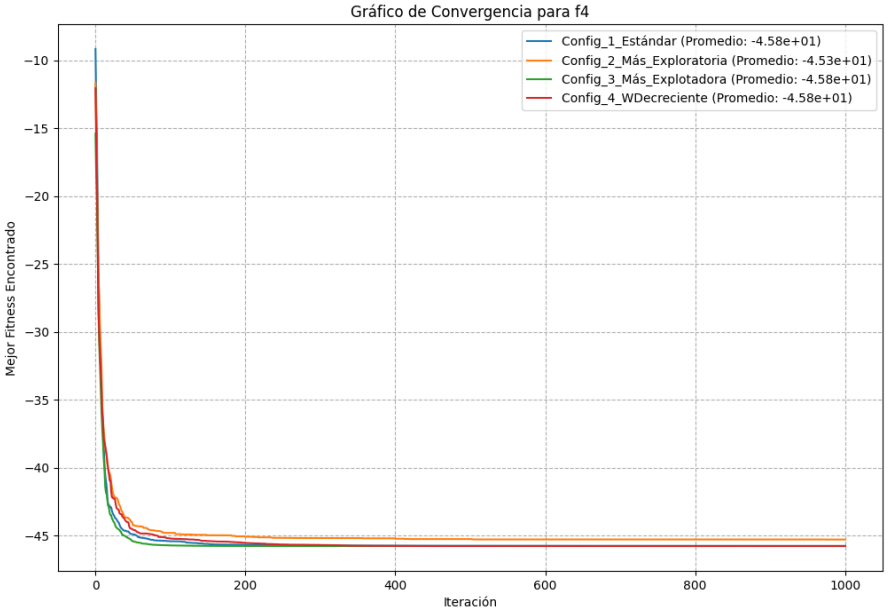

Figura 4: Gráfico de Convergencia para f4

5. Análisis de Resultados

Los resultados experimentales muestran diferencias significativas en el rendimiento de las distintas configuraciones de PSO dependiendo de la función objetivo.

1. **Funciones f1 y f2:** Para ambas funciones, PSO logró encontrar el mínimo global exacto (-516 para f1 y -1745 para f2) en las 10 ejecuciones y con las cuatro configuraciones probadas. La desviación estándar de 0.000000 confirma una consistencia perfecta. Los gráficos de convergencia (1, 2) muestran una caída muy rápida hacia el mínimo en las primeras iteraciones, con poca diferencia visible entre las configuraciones en la escala mostrada.

2. **Función f3:** El mínimo global (0) fue alcanzado de forma perfecta por las configuraciones 1, 3 y 4 en todas las ejecuciones, con desviación estándar de 0.000000. La Configuración 2 (Más Exploratoria) mostró un desempeño deficiente, con promedio 0.824 y desviación de 0.943, evidenciando dificultad para converger. El gráfico de convergencia (3) muestra cómo Config 2 se estanca, mientras las demás convergen rápidamente al mínimo, destacando la efectividad de estrategias explotadoras.

3. **Función f4:** Se logró obtener un valor mínimo consistente de aproximadamente -45.778470 con Configuraciones 3 y 4 (desviación 0.000000). Config 1 también fue efectiva, aunque con leve variabilidad. Config 2 fue la menos eficaz, con peor promedio (-45.311590) y mayor desviación (0.118). El gráfico (4) confirma que estrategias más explotadoras superan a enfoques altamente exploratorios en funciones complejas.

6. Conclusión

La implementación del algoritmo PSO demostró ser efectiva para encontrar los mínimos de las 4 funciones objetivo. Para las funciones de menor complejidad (f1, f2 y f3), este fue capaz de encontrar el mínimo global exacto o muy cercano en la mayoría de las configuraciones paramétricas, mostrando una alta fiabilidad (desviación estándar cero en muchos casos).

Los resultados destacan la importancia de la correcta sintonización de los parámetros, especialmente en funciones más desafiantes como f3 (donde una configuración falló en la consistencia) y f4 (donde hubo claras diferencias de rendimiento entre configuraciones). La configuración más exploratoria (Config 2) consistentemente tuvo el peor desempeño en f3 y f4 en términos de calidad del mínimo encontrado, sugiriendo que un exceso de exploración sin suficiente capacidad de explotación limita la convergencia a los mejores mínimos. Las configuraciones con mayor enfoque en la explotación (Config 3) o con una inercia decreciente que equilibra exploración y explotación a lo largo del tiempo (Config 4) mostraron ser las más exitosas para encontrar los mejores mínimos de manera consistente en las funciones f3 y f4.

Para resumir, PSO es una metaheurística potente para la optimización continua, y su rendimiento en problemas complejos y multimodales está fuertemente influenciado por sus parámetros, donde un balance adecuado entre exploración y explotación es fundamental para la convergencia a óptimos de alta calidad.