Josue Santana Robledo Corona 325073061

Maestría en Ciencias de la Robótica e Inteligencia Artificial

Algoritmos Bio-Inspirados

Centro Universitario de Ciencias Exactas e Ingeniería.

Mtro. Carlos Alberto López Franco

1. **Introducción**

El Algoritmo Genético (AG) es un método de optimización inspirado en la evolución biológica. Utiliza operadores como selección, cruza y mutación para evolucionar una población de soluciones hacia el óptimo de una función objetivo.

2. **Descripción del código**

Inicializar la población aleatoriamente.

Evaluar cada individuo con la función objetivo.

Seleccionar padres mediante torneo y ruleta.

Realizar cruzamiento de un punto y mutación bit a bit.

Registrar el historial del mejor individuo a lo largo de las iteraciones.

3. **Componentes del Código**

Variables y parámetros iniciales

dimension: número de variables del problema.

limites: rango permitido para cada variable.

num_bits: cantidad de bits para codificar cada variable.

tam_poblacion: número de individuos.

max_iter: número máximo de generaciones.

prob_cruza: probabilidad de cruza entre padres.

prob_mutacion: probabilidad de mutar un bit.

minimizar: indica si se busca minimizar o maximizar la función.

porc_ruleta y porc_torneo: porcentaje de padres seleccionados por ruleta y torneo.

Función Objetivo

```
def esfera(x):
  return np.sum(x**2)
```

4. **Algoritmo Principal**

Inicialización: se genera una población de tam_poblacion individuos codificados en binario.

Decodificación: cada individuo se transforma a valores reales según num_bits y limites.

Evaluación: se calcula la función objetivo para cada individuo.

Selección de padres: se usan torneo y ruleta para elegir los padres.

Cruzamiento: se realiza cruzamiento de un punto entre pares de padres según la probabilidad prob_cruza.

Mutación: cada bit del individuo puede invertirse con probabilidad prob_mutacion.

Reemplazo y registro: se guarda el mejor individuo de la generación y se repite hasta max_iter.

Historial: se almacena la evolución del mejor fitness para graficar.

5. Resultados Obtenidos

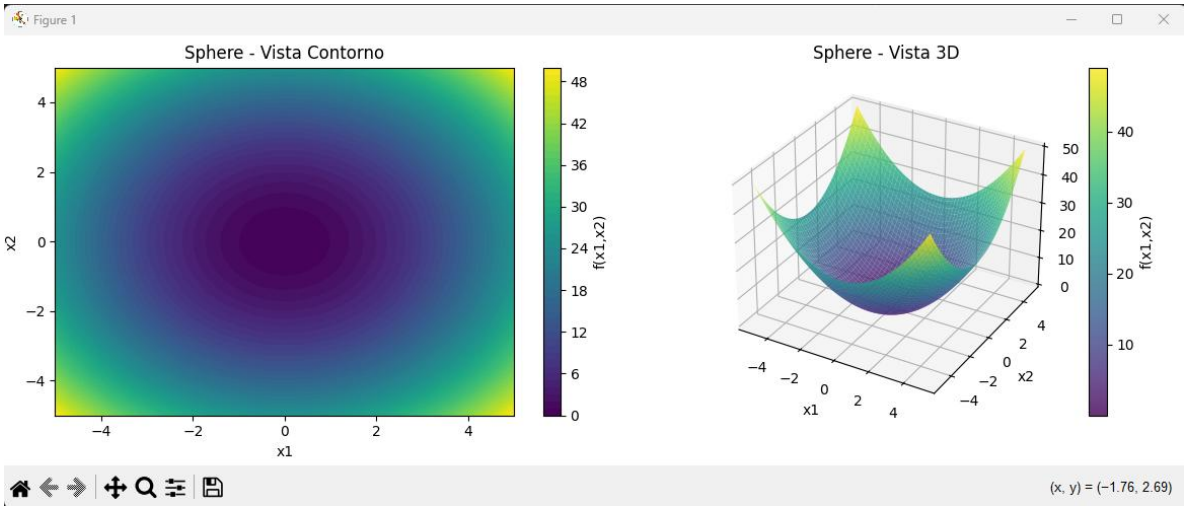

Imagen 1. Gráfica de Esfera

Primero vamos con el descendente para la función Esfera

Iteración 48: Mejor fitness = 0.000048, Mejor solución = [-0.00488759 -0.00488759]
Iteración 49: Mejor fitness = 0.000048, Mejor solución = [-0.00488759 -0.00488759]
Iteración 50: Mejor fitness = 0.000048, Mejor solución = [-0.00488759 -0.00488759]

Mejor solución final encontrada: [-0.00488759 -0.00488759]
Mejor valor de la función: 4.777698467983201e-05

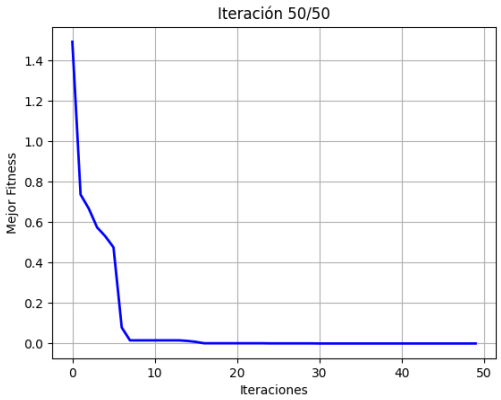

Imagen 2. Descendente Esfera

Ahora vamos con el ascendente para la función Esfera

Iteración 48: Mejor fitness = 50.000000, Mejor solución = [-5. -5.]
Iteración 49: Mejor fitness = 50.000000, Mejor solución = [-5. -5.]
Iteración 50: Mejor fitness = 50.000000, Mejor solución = [-5. -5.]

Mejor solución final encontrada: [-5. -5.]
Mejor valor de la función: 50.0

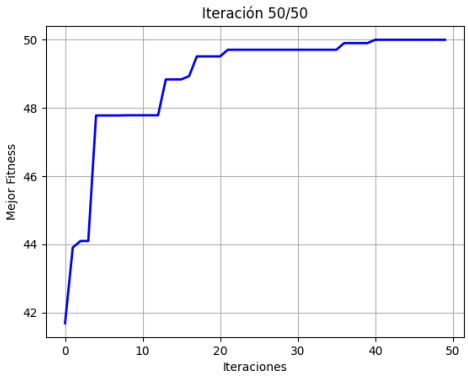

Imagen 3. Ascendente Esfera

Primero vamos con el descendente para la función Quadrico

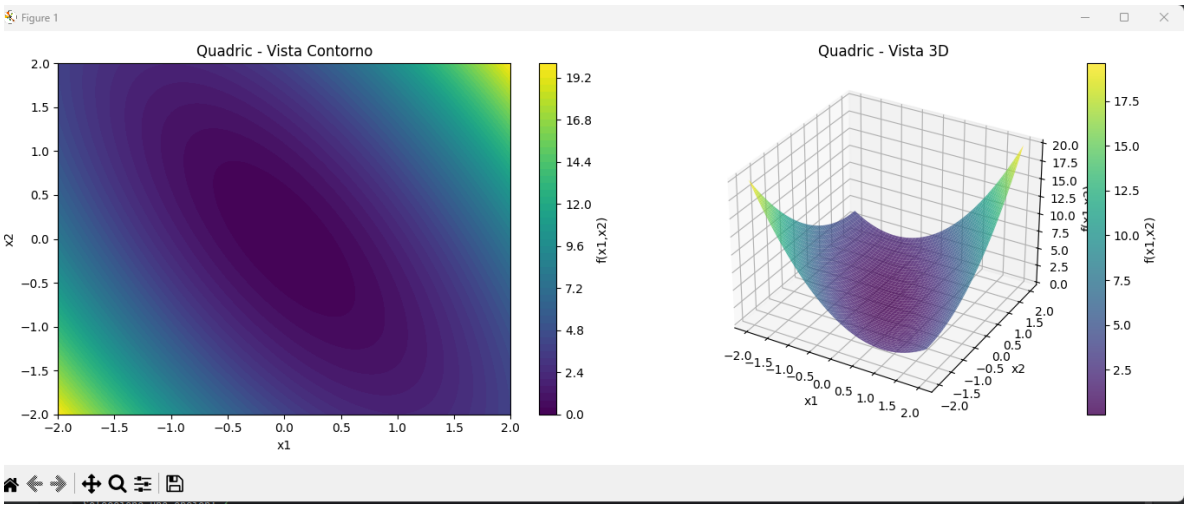

Imagen 4. Gráfica de Quadrico

Primero vamos con el descendente para la función Quadrico

Iteración 48: Mejor fitness = 0.000311, Mejor solución = [-0.01466276 0.02443793]
Iteración 49: Mejor fitness = 0.000311, Mejor solución = [-0.01466276 0.02443793]
Iteración 50: Mejor fitness = 0.000311, Mejor solución = [-0.01466276 0.02443793]

Mejor solución final encontrada: [-0.01466276 0.02443793]
Mejor valor de la función: 0.00031055040041890803

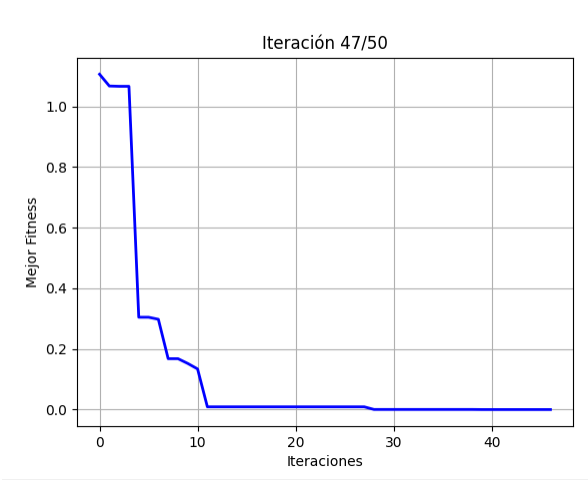

Imagen 5. Descendente Quadrico

Iteración 48: Mejor fitness = 125.000000, Mejor solución = [-5. -5.]

Iteración 49: Mejor fitness = 125.000000, Mejor solución = [-5. -5.]

Iteración 50: Mejor fitness = 125.000000, Mejor solución = [-5. -5.]

Mejor solución final encontrada: [-5. -5.]

Mejor valor de la función: 125.0

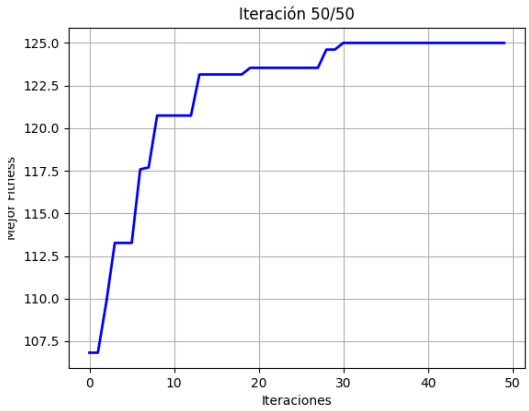

Imagen 6. Ascendente Quadrico

Con esto concluimos ejemplos en ascendente y descendente en dos funciones

6. Ventajas del Método

Algoritmo robusto y flexible para diferentes tipos de funciones.

Puede manejar espacios de búsqueda grandes y no lineales.

No requiere derivadas de la función objetivo.

Permite combinar diferentes criterios de selección y operadores genéticos.

7. Limitaciones

Puede estancarse en óptimos locales.

Mayor costo computacional en problemas de alta dimensión.

8. Conclusiones

El Algoritmo Genético es una técnica poderosa para optimización global. Su uso permite aproximarse a soluciones óptimas en problemas de minimización y maximización. Personalmente me parece curioso como la forma en la que se comporta una población haya inspirado un algoritmo. En el caso de la función de esfera que utilicé si llegaba al máximo, pero no al mínimo global de la función, hasta que cambie parte de los parámetros, con más iteraciones o cambiando el porcentaje de selección, pero creo que es un algoritmo muy parametrizable, creo que tiene más parámetros que puedes alterar y modificar. La implementación también se me llegó a complicar, pero con un poco de investigación salió. También como tema aparte, este es el programa en el que aprendí a hacer animaciones.

9. Código

https://github.com/santana-robledo/Algorithm-Lab/blob/main/17.%20Algoritmo%20Genético%20n%20dimensiones.py

Aquí se encuentra el repositorio