Global Optimization in Multi-Dimensional Search Spaces

University of Johannesburg Tshepiso

Clearance Mahoko – 220015607

Problem

Optimization is essential in science and engineering to find the best solution under constraints. Gradient Descent (GD) uses derivatives to find local minima but struggles with complex, multi-modal functions. Evolutionary Algorithms (GA and DE) provide population-based global search without gradients. This study compares GD, GA, and DE on benchmark n-dimensional functions.

Problem:

- Finding global minima in high-dimensional, multi-modal functions is difficult.
- Gradient-based methods (e.g., Gradient Descent) often get stuck in local minima.
- Evolutionary Algorithms (GA and DE) offer population-based global search

benchmark functions:

- Rastrigin,
- Ackley,
- Schwefel,
- Six-Hump Camel

Goal:

- Evaluate GD, GA, and DE to identify the most reliable method for complex landscapes.

Minimize a function:

$$\min_{x \in \mathbb{R}^n} f(x)$$

Gradient Descent

- Deterministic
- Step size
- Local minima risk
  Gradient approximation

Genetic Algorithm

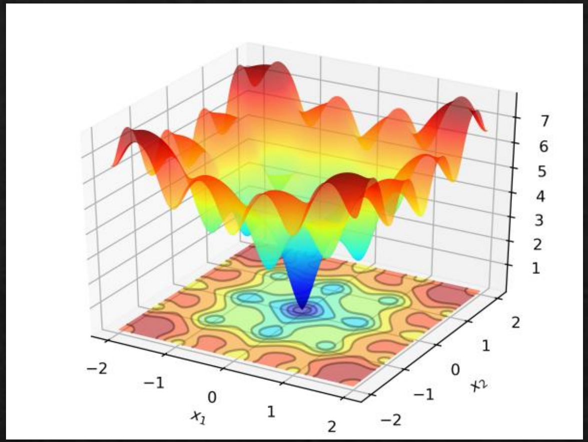
- Population-based
- Stochastic
- Crossover
  Mutation

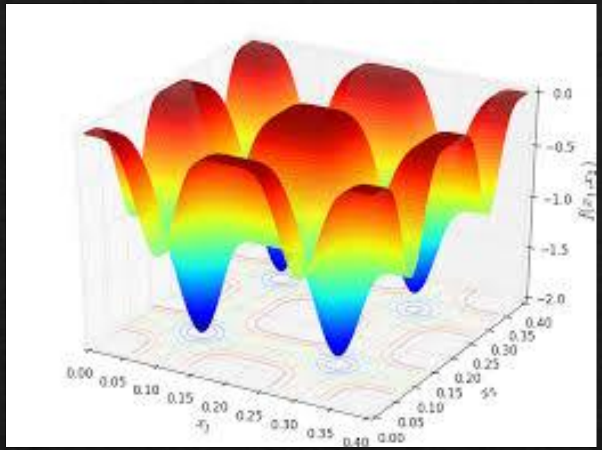
Differential Evolution

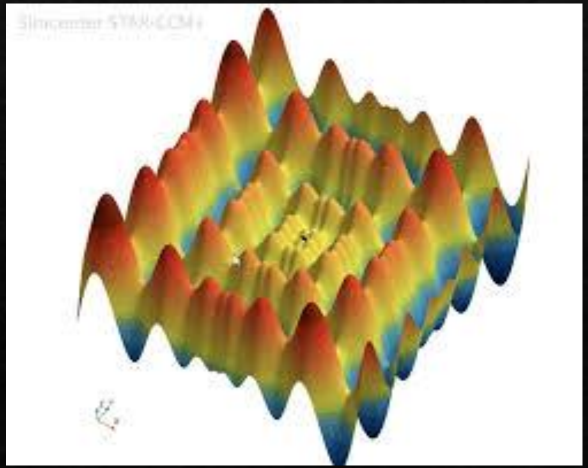
- Vector-based Mutation.
- Use of Difference vector
- Crossover : trial and target
- Global search (more diversity)

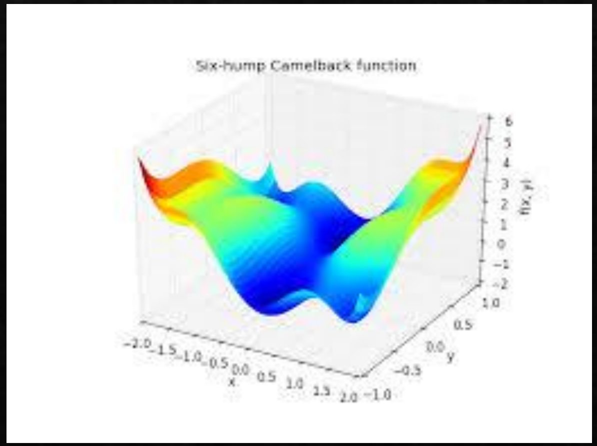
Benchmark Functions

Rastrigin, Ackley, Schwefel, Six-Hump Camel









Gradient Descent

For each component $i = 1, 2, ..., n$:

$$\frac{\partial f(x)}{\partial x_i} \approx \frac{f(x + he_i) - f(x - he_i)}{2h}$$

where

- e_i is the unit vector in the i-th direction,
- h is a small constant (e.g. 10^{-5}).

- **Path Toward the Global Minimum**
- Start with $x^{(0)}$randomly chosen.
- Compute the gradient direction $\nabla f\left(x^{(t)}\right)$.
- Estimate $\nabla f\left(x^{(t)}\right)\ using\ numerical\ gradient$
- Move opposite to the gradient (downhill) $x_{t+1} = x_t - \eta \nabla f(x_t)$
- Reduce the error $f\left(x^{(t)}\right)$iteratively.
- Stop when movement becomes very small (i.e., slope ≈ 0).
- The final $x^{(t)}$approximates x^* ,where $f\left(x^{(t)}\right)$ results in lowest value

Genetic Algorithm

- Random Initialization
- Evaluate fitness $f_i = f(x_i)$for all individuals i
- **Selection :** parents = best $\frac{\text{pop\_size}}{2}$ individuals 50% of best individual
- **Crossover (Recombination) :** child$_j = [p_1[:cp], p_2[cp:]]$ (cp is a randomly chosen crossover point) for each child , apply single crossover
- ***Mutation*** *:* Choose a random index idx and *add small random value* -1 and 1 on index idx value. child$[k] \leftarrow$ child$[k] + \delta, \delta \sim U(-1,1)$
- **next generation** : The best parents (elitism) and The newly created offspring
- Re-evaluate the final population and pick **best individual** and its **fitness value**

Differential Evolution

- Initialization
- Evaluate fitness for each individual on the population
- Mutation : Pick **three distinct individuals** $a, b, c \neq i$. $v_i = a + F \cdot (b - c)$

- Crossover (Recombination) :
- For each dimension j, **decide randomly** whether to take the gene from the mutant or the target individual x_i:

$$u_{ij} = \begin{cases} v_{ij} & \text{if } r_j < CR \\ x_{ij} & \text{otherwise} \end{cases}$$

where $r_j \sim U(0,1)$and $CR \in [0,1]$is the **crossover probability.**

- Selection
- If the **trial vector** is better (lower fitness) than the current individual:

$$x_i^{(t+1)} = u_i \text{ if } f(u_i) < f(x_i)$$

Else, retain the original individual:

$$x_i^{(t+1)} = x_i$$

- Repeat and return best indvidual

Experimental Results

- **Empirical Analysis**
- Each algorithm was run **50 times per benchmark function** to ensure robust comparison.
- **Differential Evolution (DE)** consistently outperformed others:
  - Lowest mean & median values, **highest success rates (86–100%)**.
  - Examples:
    - Rastrigin: mean 0.48737, success 86%
    - Ackley: mean 0.00000, success 100%
- **Gradient Descent (GD)**: high variability, low success, often trapped in local minima.
- **Genetic Algorithm (GA)**: moderate performance, better than GD but less reliable than DE.

Conclusion

- **Evolutionary Algorithms (GA & DE)** are more effective for **multi-dimensional, multi-modal optimization** than GD.
- **Differential Evolution (DE)** provides the best balance between **exploration and convergence speed**.
- **Gradient Descent** is suitable only for **simple, convex problems**.
- Future work: explore **hybrid or advanced evolutionary strategies** for very large or complex search spaces.

Prototype

- **Rastrigin :** $f(x) = 10 \cdot n + \sum_{i=1}^{n}[x_i^2 - 10\cos(2\pi x_i)]\,, x_i \in [-5.12,5.12]$
- **Ackley** : $f(x) = -20\exp(\,-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} \quad x_i^2}\,) - \exp(\frac{1}{n}\sum_{i=1}^{n} \quad \cos(2\pi x_i)) + 20 + e, x_i \in [-5,5]$
- **Schwefel :** $f(x) = 418.9829 \cdot n - \sum_{i=1}^{n} \quad x_i \sin(\sqrt{|\, x_i \,|}), x_i \in [-500,500]$
- **Six-Hump Camel :** $f(x_1, x_2) = (4 - 2.1x_1^2 + \frac{x_1^4}{3})x_1^2 + x_1x_2 + (-4 + 4x_2^2)x_2^2, x_1 \in [-3,3], x_2 \in [-2,2]$
- Evaluate all against the three optimisation methods