Mathematical Reasoning Notes

Mohamad SAMMAN

April 2025

1 Differential Evolution Algorithm

Differential Evolution (DE) is a stochastic, population-based optimization method designed for continuous-valued, nonlinear, and possibly non-differentiable objective functions. It belongs to the class of evolutionary algorithms and is particularly effective for global optimization over bounded domains.

1.1 Overview

Given an objective function $f : \mathbb{R}^n \to \mathbb{R}$, the DE algorithm seeks to find the global minimum over a bounded domain. The algorithm maintains a population of candidate solutions $\{x_i^g\}_{i=1}^{N_p}$ at generation g, where each $x_i^g \in \mathbb{R}^n$, and N_p is the population size.

1.2 Algorithm Steps

1. Initialization Each individual is initialized randomly within the search bounds:

$$x_i^0 = x_{\min} + r_i \cdot (x_{\max} - x_{\min}), \quad r_i \sim \mathcal{U}(0,1)^n$$

2. Mutation For each individual x_i^g, a mutant vector v_i^{g+1} is created using the `DE/rand/1` strategy:

$$v_i^{g+1} = x_{r1}^g + F \cdot (x_{r2}^g - x_{r3}^g)$$

where $r1, r2, r3 \in \{1, \dots, N_p\}$, are mutually distinct indices and $\neq i$, and $F \in [0, 2]$ is the mutation factor.

3. Crossover A trial vector u_i^{g+1} is formed by mixing the target vector x_i^g and the donor vector v_i^{g+1}:

$$u_{i,j}^{g+1} = \begin{cases} v_{i,j}^{g+1} & \text{if } r_j \leq CR \text{ or } j = j_{\text{rand}} \\ x_{i,j}^g & \text{otherwise} \end{cases}$$

where $r_j \sim \mathcal{U}(0,1)$, $CR \in [0,1]$ is the crossover probability, and $j_{\text{rand}} \in \{1, \dots, n\}$ is a randomly chosen index to ensure at least one component is inherited from v_i^{g+1}.

4. Selection The new individual for the next generation is chosen based on the objective function:

$$x_i^{g+1} = \begin{cases} u_i^{g+1} & \text{if } f(u_i^{g+1}) \leq f(x_i^g) \\ x_i^g & \text{otherwise} \end{cases}$$

1.3 Convergence and Characteristics

DE is robust to local minima and does not require gradient information, making it suitable for noisy or discontinuous objective functions. The algorithm balances exploration and exploitation through parameters F, CR, and the chosen mutation strategy. Though convergence guarantees are limited to specific theoretical settings, DE performs well empirically in a wide range of practical optimization problems.

1.4 SciPy Implementation Notes

In `scipy.optimize.differential_evolution`, the default strategy is `DE/best/1/bin`, using the current best individual as the base for mutation. The implementation supports box constraints via the `bounds` parameter and optionally handles nonlinear constraints using penalty-based or trust-region methods.