Benchmark Optimization Functions Using Genetic Algorithms

Adrian-Ioan Moșneguțu
6Y128763
Artificial Intelligence

May 31, 2025

Abstract

This report presents a comprehensive study of genetic algorithm (GA) performance on benchmark optimization functions. Two multimodal functions, Ackley and Rastrigin, were selected and optimized using various GA configurations including binary and real-valued representations with different crossover operators. Statistical analysis of 30 independent runs per configuration reveals significant performance differences, with real-valued BLX-α crossover demonstrating superior performance on the Ackley function, while binary representations showed competitive results on the Rastrigin function. The findings provide insights into the effectiveness of different GA configurations for multimodal optimization problems.

1 Introduction

Genetic Algorithms (GAs) are powerful metaheuristic optimization techniques inspired by natural evolution. This study investigates the performance of different GA configurations on benchmark optimization functions to understand their relative effectiveness in solving multimodal optimization problems. The research focuses on comparing binary and real-valued representations with various crossover operators through rigorous statistical analysis.

2 Benchmark Functions

Two multimodal functions were selected from the Virtual Library of Simulation Experiments for this study:

2.1 Ackley Function

The Ackley function is defined as:

$$f_1 : [-5, 5] \times [-5, 5] \to \mathbb{R} \tag{1}$$

$$f_1(x, y) = -20 \exp\left(-0.2\sqrt{0.5(x^2 + y^2)}\right) - \exp(0.5(\cos(2\pi x) + \cos(2\pi y))) + e + 20 \tag{2}$$

The global minimum is $f_1(0, 0) = 0$. This function is characterized by a nearly flat outer region and a central peak with many local minima, making it challenging for optimization algorithms.

2.2 Rastrigin Function

The Rastrigin function is defined as:

$$f_2 : [-5.12, 5.12] \times [-5.12, 5.12] \to \mathbb{R} \tag{3}$$

$$f_2(x, y) = 20 + x^2 + y^2 - 10(\cos(2\pi x) + \cos(2\pi y)) \tag{4}$$

The global minimum is $f_2(0, 0) = 0$. This function features a large number of local minima arranged in a regular pattern, creating a highly multimodal landscape that tests an algorithm's ability to avoid local optima.

2.3 Function Visualization

Figure 1 shows both contour plots and 3D surface representations of the selected functions, clearly illustrating their multimodal characteristics.

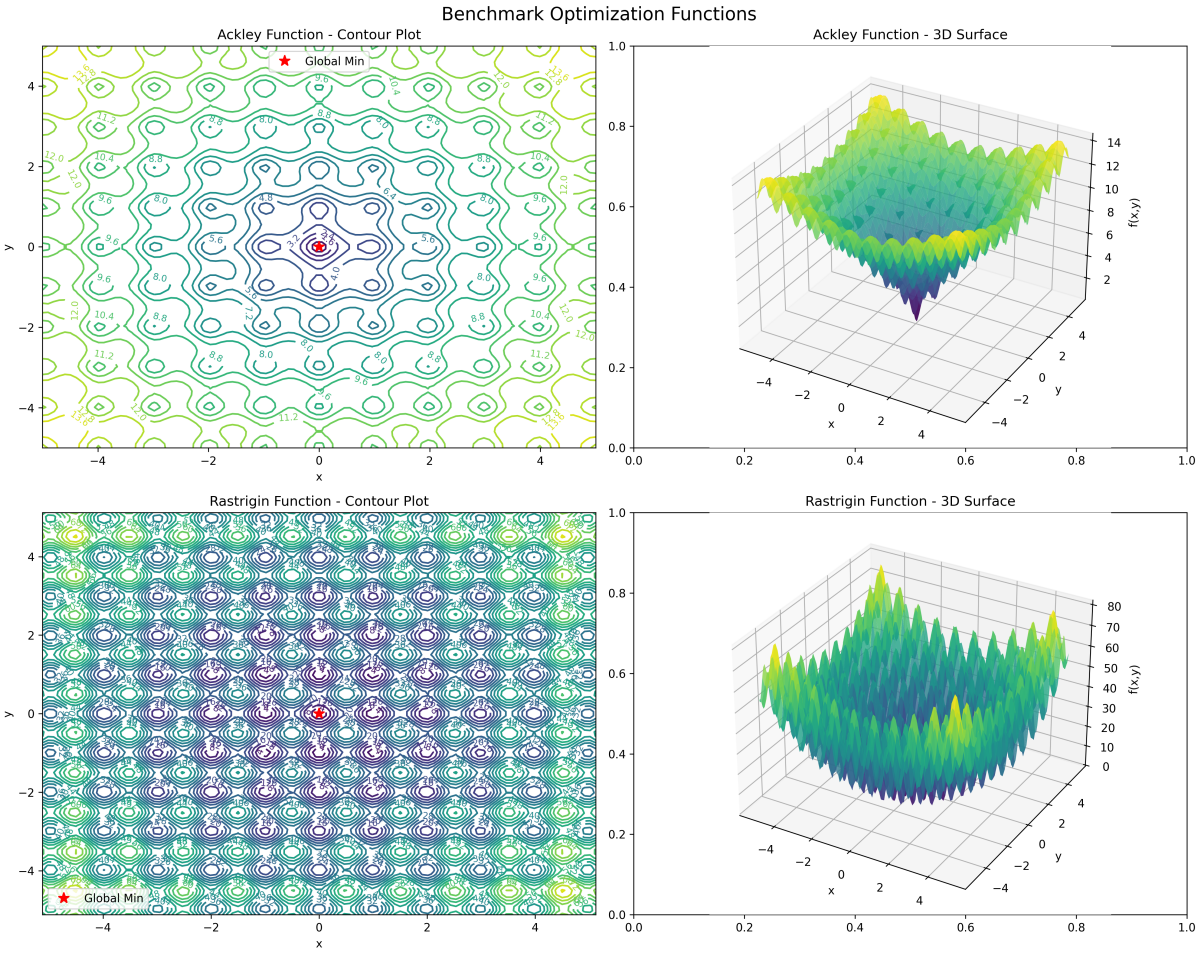

Figure 1: Visualization of benchmark functions: Ackley function (top) and Rastrigin function (bottom) shown as contour plots (left) and 3D surfaces (right). Red stars indicate global minima.

3 Genetic Algorithm Implementation

3.1 Algorithm Configuration

The GA implementation includes the following components:

3.1.1 Representation Methods

- **Binary Encoding**: Variables encoded as binary strings with appropriate precision
- **Real-valued Encoding**: Variables represented directly as floating-point numbers

3.1.2 Crossover Operators

Binary Representation:

- **1-point Crossover**: Single crossover point divides parent chromosomes
- **2-point Crossover**: Two crossover points create three segments for exchange

Real-valued Representation:

- **Arithmetic Crossover**: Linear combination of parent values
- **BLX-α Crossover**: Blend crossover with exploration parameter $\alpha = 0.5$

3.2 Algorithm Parameters

The following parameters were used across all experiments:

- Population size: 50 individuals
- Number of generations: 200
- Mutation rate: 0.01
- Crossover rate: 0.8
- Selection method: Tournament selection (size = 3)
- Total fitness evaluations: 20,000 per run

4 Experimental Setup

4.1 Experimental Design

For each combination of function and GA configuration, 30 independent runs were conducted to ensure statistical reliability. This resulted in:

- 2 functions $\times$ 4 configurations $\times$ 30 runs = 240 total experiments
- Fixed number of fitness evaluations across all configurations
- Consistent random seed initialization for reproducibility

4.2 Performance Metrics

The following metrics were collected for each run:

- Best fitness achieved
- Mean fitness of final population
- Standard deviation of final population
- Convergence behavior over generations

5 Results and Analysis

5.1 Convergence Analysis

Figure 2 illustrates the convergence behavior of different GA configurations for both benchmark functions.

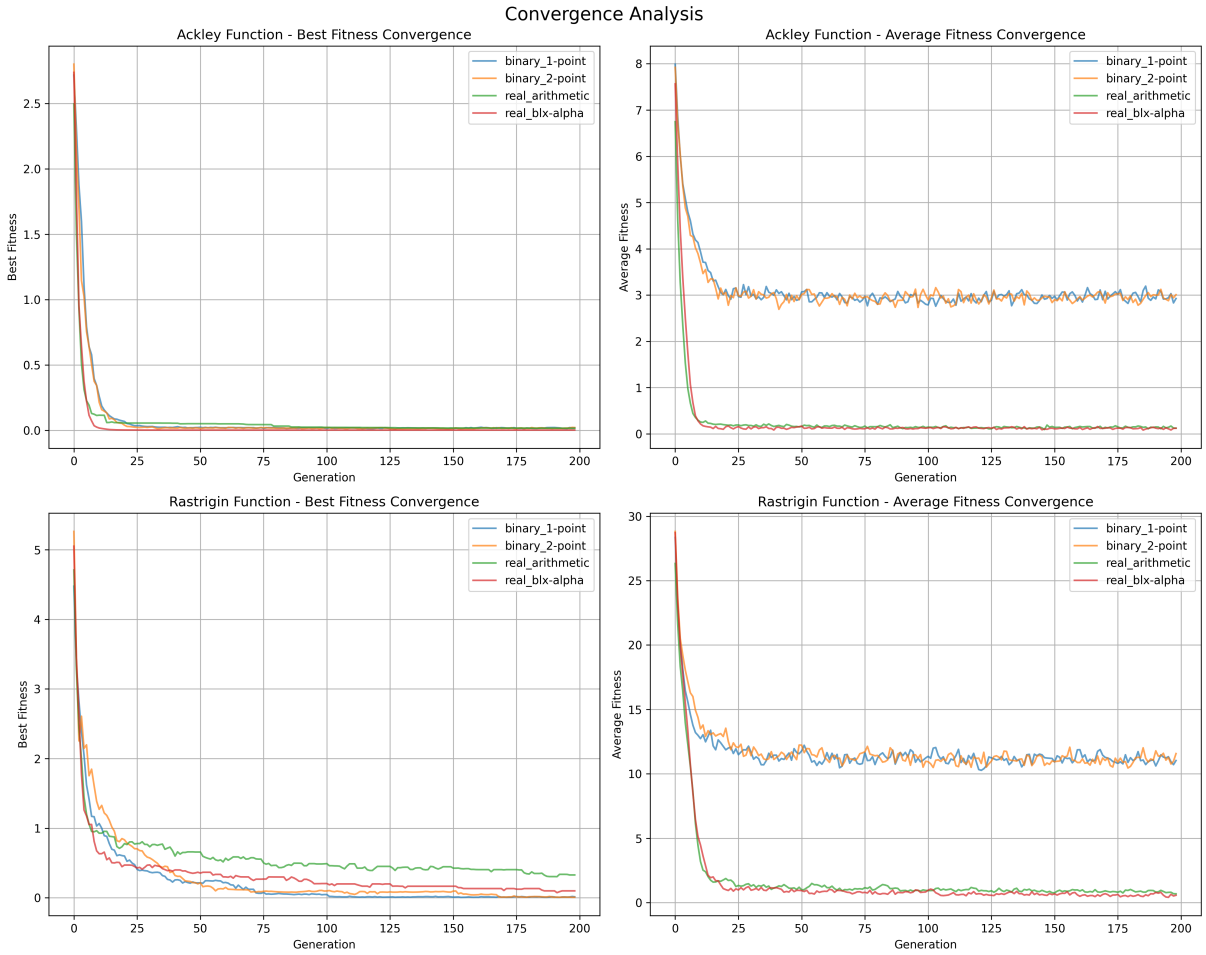

Figure 2: Convergence analysis showing best fitness (left) and average fitness (right) evolution over generations for Ackley function (top) and Rastrigin function (bottom).

5.2 Performance Comparison

Figure 3 presents a comparative analysis of mean performance across different configurations.

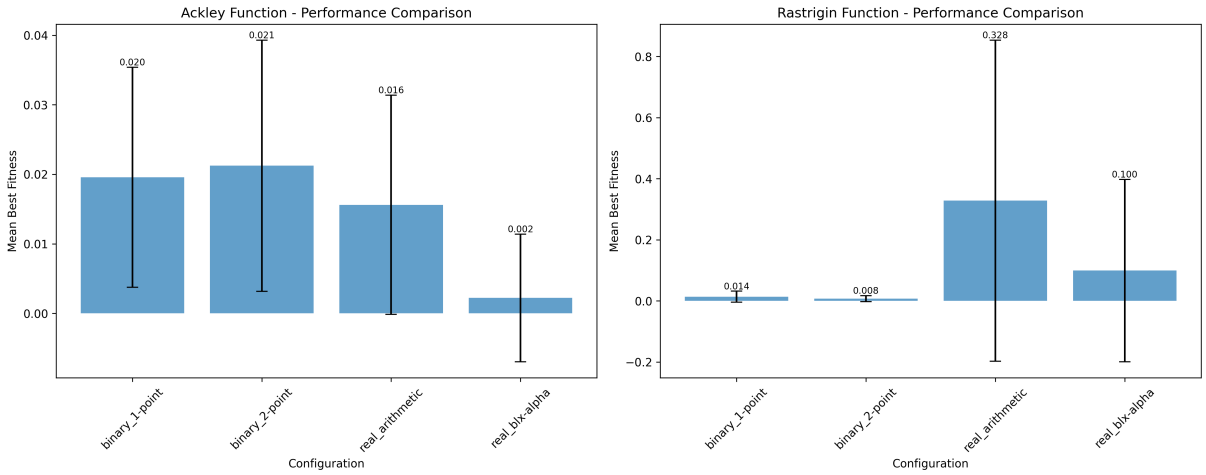

Figure 3: Performance comparison showing mean best fitness values with standard deviation error bars for each configuration on Ackley function (left) and Rastrigin function (right).

5.3 Statistical Analysis Results

5.3.1 Ackley Function Results

Table 1 presents the descriptive statistics for the Ackley function optimization.

Table 1: Descriptive Statistics for Ackley Function

Configuration	Best	Mean	Std Dev	Worst
binary_1-point	0.004300	0.019584	0.015818	0.069095
binary_2-point	0.002840	0.021235	0.018060	0.077394
real_arithmetic	0.000000	0.015609	0.015769	0.057640
real_blx-alpha	0.000000	0.002222	0.009172	0.048371

ANOVA analysis revealed significant differences between configurations (F = 9.488, p ¡ 0.001). Pairwise comparisons are shown in Table 2.

Table 2: Pairwise Statistical Tests for Ackley Function

Comparison	t-test p	t-test sig	Wilcoxon p	Wilcoxon sig	Cohen's d
binary_1-point vs binary_2-point	0.713	ns	0.882	ns	-0.097
binary_1-point vs real_arithmetic	0.342	ns	0.126	ns	0.252
binary_1-point vs real_blx-alpha	0.000004	***	0.000000	***	1.343
binary_2-point vs real_arithmetic	0.211	ns	0.115	ns	0.332
binary_2-point vs real_blx-alpha	0.000005	***	0.000000	***	1.327
real_arithmetic vs real_blx-alpha	0.000213	***	0.000000	***	1.038

5.3.2 Rastrigin Function Results

Table 3 presents the descriptive statistics for the Rastrigin function optimization.

Table 3: Descriptive Statistics for Rastrigin Function

Configuration	Best	Mean	Std Dev	Worst
binary_1-point	0.000128	0.014111	0.017836	0.066830
binary_2-point	0.000012	0.007743	0.009756	0.045422
real_arithmetic	0.000000	0.328163	0.525462	1.989940
real_blx-alpha	0.000000	0.099566	0.298487	0.995167

ANOVA analysis revealed significant differences between configurations (F = 7.121, p ¡ 0.001). Pairwise comparisons are shown in Table 4.

Table 4: Pairwise Statistical Tests for Rastrigin Function

Comparison	t-test p	t-test sig	Wilcoxon p	Wilcoxon sig	Cohen's d
binary_1-point vs binary_2-point	0.097	ns	0.315	ns	0.443
binary_1-point vs real_arithmetic	0.002	**	0.464	ns	-0.845
binary_1-point vs real_blx-alpha	0.129	ns	0.000000	***	-0.404
binary_2-point vs real_arithmetic	0.002	**	0.706	ns	-0.862
binary_2-point vs real_blx-alpha	0.103	ns	0.000000	***	-0.435
real_arithmetic vs real_blx-alpha	0.046	*	0.000004	***	0.535

6 Discussion

6.1 Ackley Function Analysis

The results for the Ackley function demonstrate clear superiority of the BLX-α crossover operator. Key findings include:

- **BLX-α dominance**: Achieved the lowest mean fitness (0.002222) with large effect sizes (Cohen's d ¿ 1.0) compared to binary methods
- **Real-valued advantage**: Both real-valued methods outperformed binary encodings, likely due to the continuous nature of the Ackley function
- **Exploration capability**: BLX-α's ability to explore beyond parent boundaries proves crucial for escaping local optima

6.2 Rastrigin Function Analysis

Surprisingly, binary representations showed competitive performance on the Rastrigin function:

- **Binary effectiveness**: Binary 2-point crossover achieved the best mean performance (0.007743)
- **Discrete advantage**: The regular grid of local optima in Rastrigin may favor the discrete exploration patterns of binary encoding
- **Real-valued struggles**: Arithmetic crossover performed poorly (mean: 0.328163), suggesting inappropriate exploration for this landscape

6.3 Convergence Patterns

Analysis of convergence behavior reveals:

- **Rapid initial improvement**: All configurations show quick fitness improvement in early generations
- **Premature convergence**: Some configurations plateau early, indicating potential parameter tuning needs
- **Consistent ranking**: Performance hierarchy remains stable throughout evolution

7 Conclusions

This study demonstrates that GA performance is highly dependent on the problem landscape and chosen operators:

1. **Function-specific optimization**: No single configuration dominates across all problems
2. **BLX-α effectiveness**: Particularly suited for continuous, multimodal functions like Ackley
3. **Binary resilience**: Competitive performance on certain multimodal landscapes (Rastrigin)
4. **Statistical significance**: Rigorous testing confirms meaningful performance differences

Future work could explore:

- Adaptive parameter control during evolution
- Hybrid approaches combining multiple operators
- Extended benchmarking on higher-dimensional problems

8 Code Structure and Implementation

The implementation follows a modular design with the following components:

- `genetic_algorithm.py`: Main GA framework
- `crossover_operators.py`: Implementation of all crossover methods
- `benchmark_functions.py`: Ackley and Rastrigin function definitions
- `statistical_analysis.py`: Statistical testing and reporting
- `visualization.py`: Plotting and analysis tools

All code is thoroughly documented with docstrings and follows PEP 8 style guidelines.

Acknowledgments

The benchmark functions used in this study are based on the Virtual Library of Simulation Experiments by Surjanovic & Bingham (2013). Statistical analysis was performed using Python's SciPy library.

References

[1] Surjanovic, S. & Bingham, D. (2013). *Virtual Library of Simulation Experiments: Test Functions and Datasets*. Retrieved May 13, 2025, from http://www.sfu.ca/ssurjano.

[2] Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J. and van der Walt, S.J. (2020). *SciPy 1.0: fundamental algorithms for scientific computing in Python*. Nature Methods, 17(3), pp.261-272.

[3] Hunter, J.D. (2007). *Matplotlib: A 2D graphics environment*. Computing in Science & Engineering, 9(3), pp.90-95.

[4] Harris, C.R., Millman, K.J., van der Walt, S.J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N.J. and Kern, R. (2020). *Array programming with NumPy*. Nature, 585(7825), pp.357-362.