Benchmark Optimization Functions Using Genetic Algorithms

1. Introduction

This project implements a comparative study of Genetic Algorithm (GA) optimization techniques on two multimodal benchmark functions in two variables. The goal is to evaluate the performance of GA variants with different representations and crossover operators to identify configurations that yield superior optimization results on challenging test functions.

2. Selected Benchmark Functions

Two well-known multimodal functions from the benchmark set were chosen:

2.1 Cross-in-Tray Function f_1

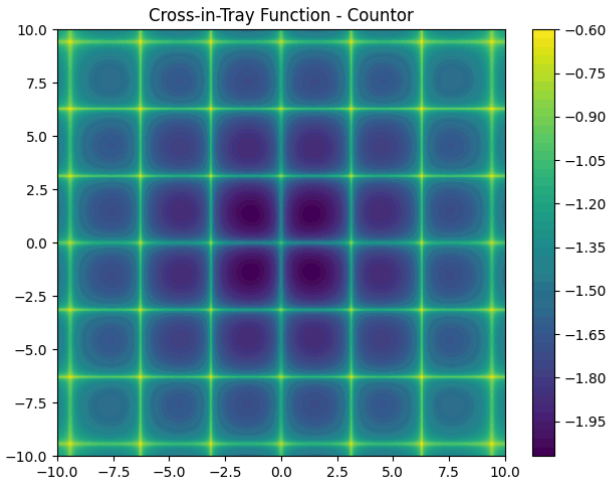
- **Domain:**

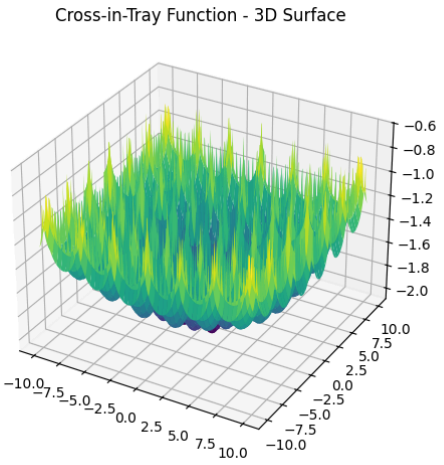
  $x,\ y \in [-\ 10,\ 10]$
- **Mathematical definition:**

$$f1(x,\ y)\ =\ -\ 0.0001\left(\left|sin(x)\ sin(y)\ exp\left(\left|100\ -\ \frac{\sqrt{x^2+y^2}}{\pi}\right|\right)\right|\ +\ 1\right)^{0.1}$$

- **Characteristics:**

  Highly multimodal with multiple global minima, posing a difficult landscape for optimization algorithms.





2.2 Bukin Function N.6 f_2

- **Domain:**

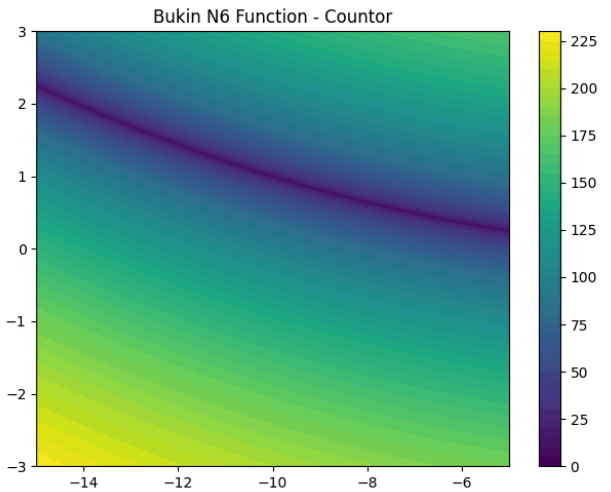
  $x \in [-15, 5], y \in [-3, 3]$

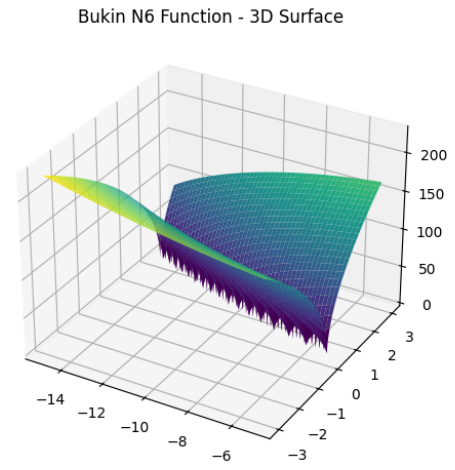
- **Mathematical definition:**

  $$f2(x, y) = 100\sqrt{|y - 0.01x^2|} + 0.01|x + 10|$$

- **Characteristics:**

  Contains steep ridges and a narrow search space, challenging convergence and exploration balance.





3. Function Implementation and Visualization

- Both functions were implemented in Python using vectorized operations via NumPy for efficiency.

- Visualization includes:

    - **2D contour plots** to illustrate function topography and minima distribution.

    - **3D surface plots** to demonstrate the shape and modality visually.

- These visualizations aid in understanding function complexity and inform parameter choices for optimization.

4. Genetic Algorithm Design

The GA framework supports:

4.1 Representation Types

- **Binary Encoding:**
  Each variable is encoded as a fixed-length bitstring, concatenated to form an individual genome. Enables discrete crossover and mutation operations at bit-level granularity.

- **Real-valued Encoding:**
  Individuals are represented as vectors of floating-point numbers corresponding directly to variable values, allowing continuous search space exploration.

4.2 Crossover Operators

- **Binary Encoding:**

    - *One-point crossover:* A random single crossover point splits parents; offspring are formed by exchanging segments.

    - *Two-point crossover:* Two crossover points define a middle segment exchanged between parents.

- **Real-valued Encoding:**

    - *Arithmetic crossover:* Offspring genes are convex combinations of parent genes weighted by a random factor.

    - *BLX-a crossover:* Offspring genes are sampled uniformly from an extended interval defined by parent gene values and a parameter α\alpha controlling exploration beyond parents.

4.3 Mutation Operators

- **Binary:** Bit-flip mutation with a specified probability per bit.

- **Real-valued:** Gaussian perturbation within variable bounds to maintain diversity.

4.4 Algorithm Parameters

- Population size

- Number of generations

- Mutation rate

- Crossover rate

- BLX-α parameter (for BLX crossover)

All parameters are adjustable to facilitate systematic experimentation.

5. Experimental Setup

- Fixed total number of fitness function evaluations for fair comparison across configurations.
- For each function f_1 and f_2, experiments run with all four combinations of representation and crossover:

    1. Binary + One-point crossover
    2. Binary + Two-point crossover
    3. Real-valued + Arithmetic crossover
    4. Real-valued + BLX-α crossover

- Each configuration was independently executed 30 times to gather robust statistical data.

6. Performance Metrics and Statistical Analysis

To evaluate the performance of the Genetic Algorithm (GA) configurations across the selected benchmark functions, we conducted a comprehensive analysis consisting of summary statistics and pairwise statistical hypothesis testing.

6.1 Summary Statistics

For each combination of:

- **Function** (f1, f2)
- **Encoding Type** (binary, real)
- **Crossover Method** (1-point, 2-point, arithmetic, BLX-α)

We performed **30 independent runs** and recorded the **best fitness value** obtained in each run.

The following descriptive statistics were computed per configuration:

- **Best Fitness**
- **Mean fitness**
- **Standard deviation**
- **Median fitness**
- **Number of runs (n = 30)**

6.2 Visualization of Distributions

To illustrate the performance distributions, we generated the following plots for each function:

- **Box plots**: Show median, quartiles, and potential outliers.
- **Violin plots**: Visualize the probability density of the results.
- **Strip plots**: Display individual fitness results with jitter to reveal clustering.

Each plot compares the four configurations:

- binary_1point
- binary_2point
- real_arithmetic
- real_blx

6.3 Statistical Hypothesis Testing

To assess whether observed performance differences between GA configurations are **statistically significant**, we conducted pairwise comparisons between all configuration pairs using:

- **Welch's t-test**: For unequal variances and sample sizes.
- **Wilcoxon signed-rank test**: Applied when both distributions had equal size and enough samples (n > 10), to account for non-normal distributions.

Each test was conducted separately for both functions. The results include:

- **T-statistic** and **p-value** from Welch's t-test

- **Wilcoxon statistic** and **p-value**

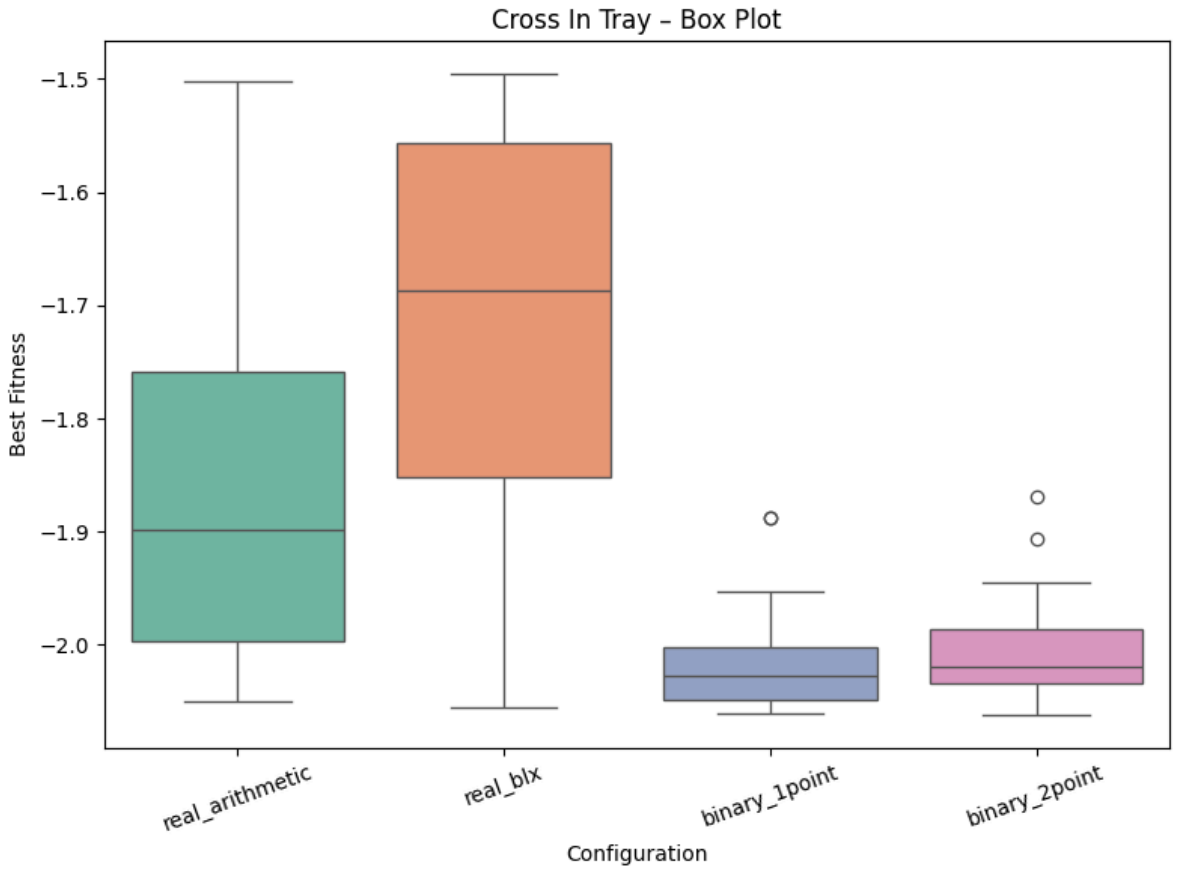
7. Results

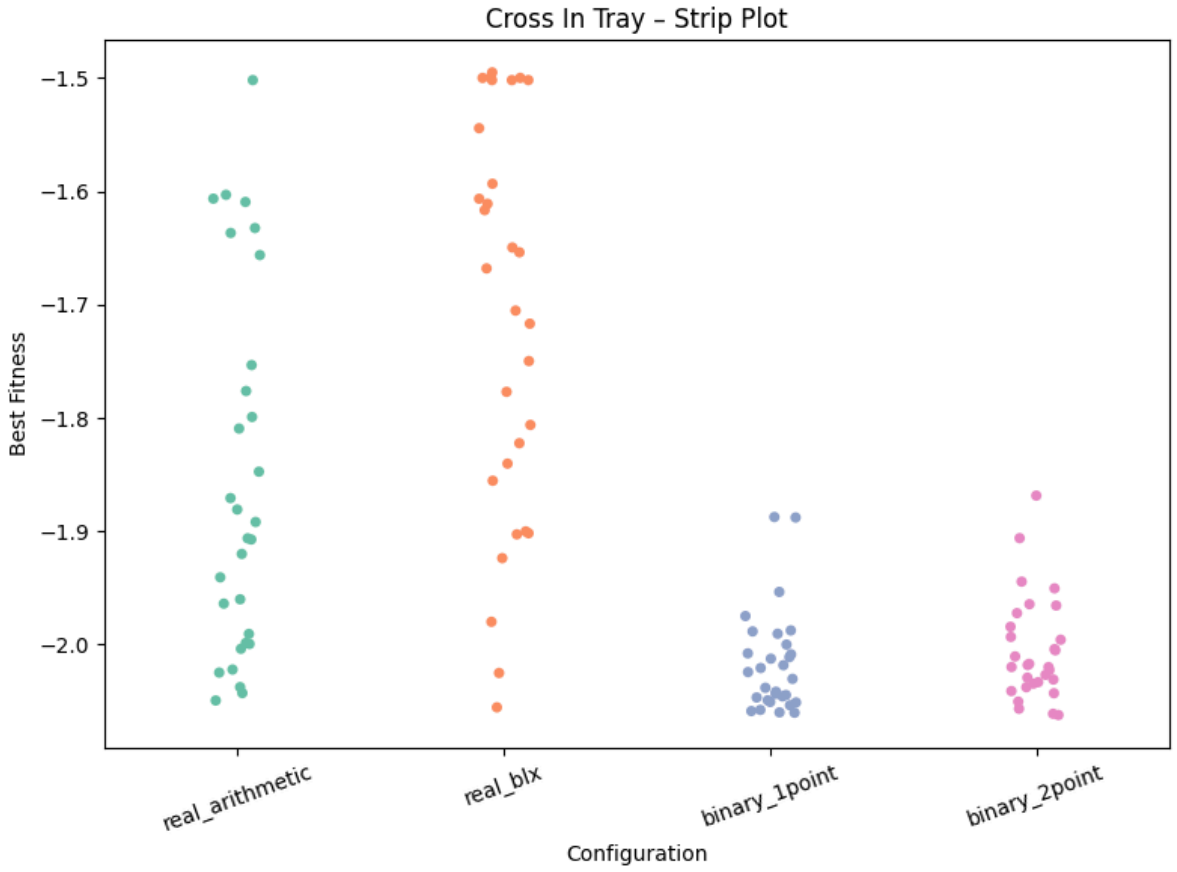
We analyzed the results using the following statistical metrics:

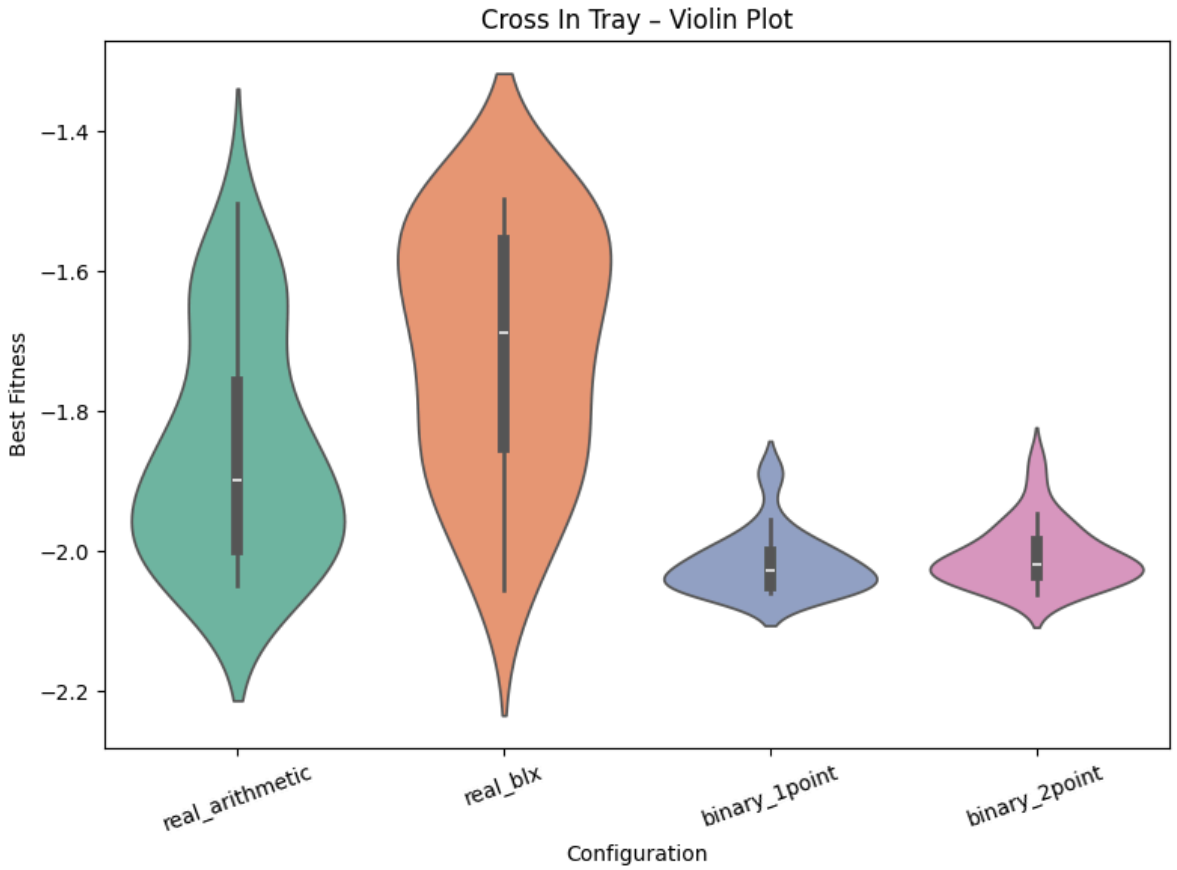
- **Best Fitness**: Measures the best performance across all runs.
- **Mean fitness**: Measures the average performance across all runs.
- **Standard deviation (Std)**: Captures consistency (lower is better).
- **Median**: The median value achieved among all runs.

7.1 Cross-in-Tray Function

Configuration	Best Fitness	Mean Fitness	Std Dev	Median
real - arithmetic	-2.0493051253067827	-1.8546932084234790	0.1614394252148623 8	-1.8989775461296607
real - blx	-2.0553623196168997	-1.7134337158388390	0.1762510351158741	-1.6866115637025805
binary - 1point	-2.0601139356280240	-2.0167667225597990	0.0448601234846519 25	-2.0272183836310145
binary - 2point	-2.0621352749873480	-2.0055596522528590	0.0452562369668541 64	-2.0188847284744966

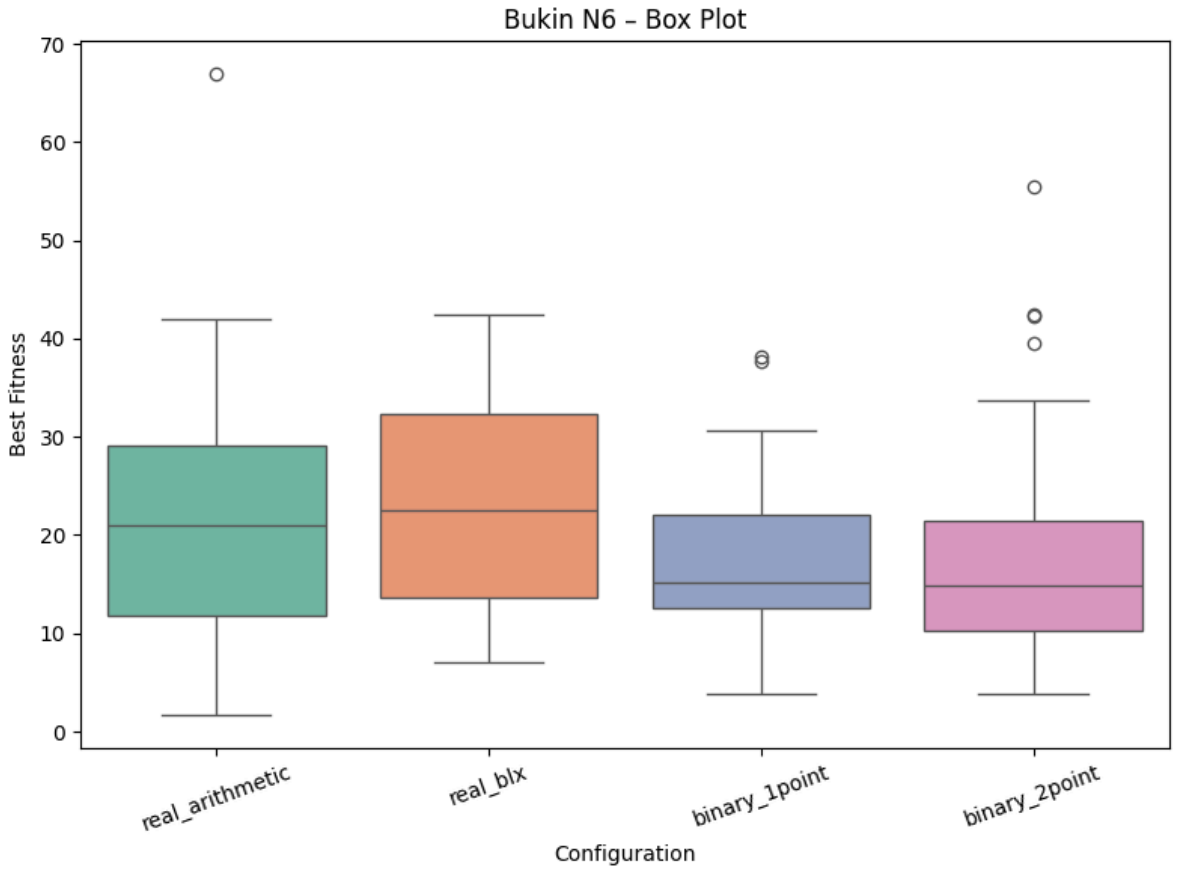

Cross In Tray – Box Plot
Best Fitness
−1.5
−1.6
−1.7
−1.8
−1.9
−2.0
real_arithmetic
real_blx
binary_1point
binary_2point
Configuration

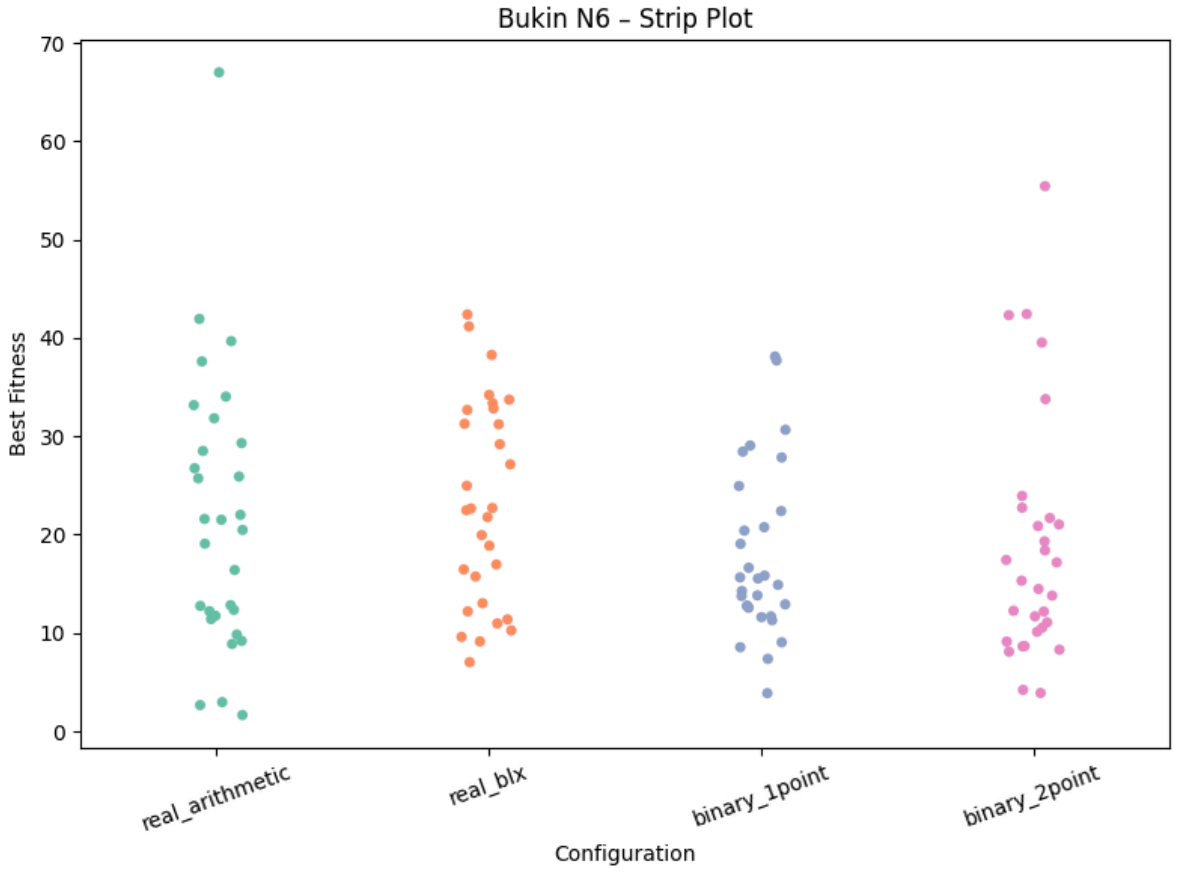

Cross In Tray – Strip Plot
Best Fitness
−1.5
−1.6
−1.7
−1.8
−1.9
−2.0
real_arithmetic
real_blx
binary_1point
binary_2point
Configuration

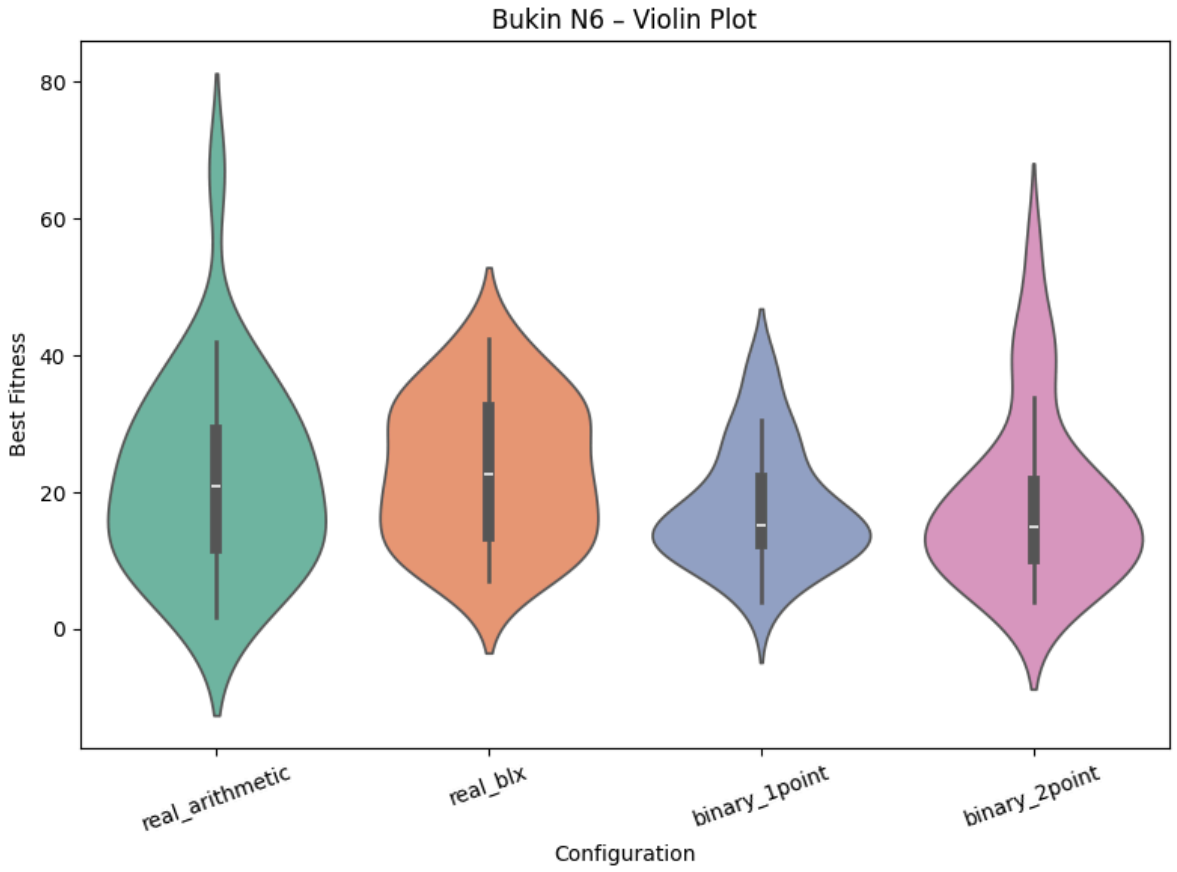


7.2 Buking Function N. 6

Configuration	Best Fitness	Mean Fitness	Std Dev	Median
real - arithmetic	1.664995490 0218895	21.70702951 657231	14.06865744 4125432	21.00047710 338049
real - blx	7.037006549 43594	23.12626967 942891	10.36681829 5309097	22.57402139 1750776
binary - 1point	3.885355973 1596863	17.81321449 408585	8.624039450 231948	15.21978869 519063
binary - 2point	3.907600094 299448	18.62044913 0184404	12.49164309 9112292	14.90172565 1693388


Bukin N6 – Box Plot
Best Fitness
0
10
20
30
40
50
60
70
real_arithmetic
real_blx
binary_1point
binary_2point
Configuration


Bukin N6 – Strip Plot
Best Fitness
0
10
20
30
40
50
60
70
real_arithmetic
real_blx
binary_1point
binary_2point
Configuration



Statistical Analysis of Crossover Configurations

To evaluate the performance differences among various crossover operators in the Genetic Algorithm, we conducted pairwise statistical tests using:

- **Student's t-test**: Assesses differences in mean performance.
- **Wilcoxon signed-rank test**: A non-parametric alternative that considers paired differences in medians.

1. Cross-in-Tray Function

- **Real encodings (real_arithmetic vs. real_blx)**
  - **Significant difference** (t-test p ≈ 0.0020, Wilcoxon p ≈ 0.0106).
  - **Conclusion**: Real arithmetic outperforms real BLX.
- **Real vs. Binary crossover (1-point or 2-point)**:
  - **Real_arithmetic vs. binary_1point**: **Highly significant** ($p < 0.00001$).

    - **Real_arithmetic vs. binary_2point**: **Highly significant** ($p < 0.00005$).
    - **Real_blx vs. binary_1point**: **Extremely significant** ($p < 1e-08$).
    - **Real_blx vs. binary_2point**: **Extremely significant** ($p < 1e-08$).
    - **Conclusion**: Real-valued crossovers significantly outperform binary encodings.
- **Binary 1-point vs. 2-point**:
    - No significant difference ($p > 0.33$).
    - **Conclusion**: Binary crossover types perform similarly.

2. Bukin Function N.6

- **Real encodings (real_arithmetic vs. real_blx)**:
    - No significant difference ($p \approx 0.66$).
    - **Conclusion**: Real arithmetic and real BLX perform similarly.
- **Real vs. Binary**:
    - **Real_arithmetic vs. binary_1point and binary_2point**: No significant difference ($p > 0.2$).
    - **Real_blx vs. binary_1point**: **Statistically significant** (t-test $p \approx 0.035$, Wilcoxon $p \approx 0.012$).
    - **Real_blx vs. binary_2point**: No significant difference ($p > 0.13$).
    - **Conclusion**: Real BLX significantly outperforms binary 1-point crossover but not binary 2-point.
- **Binary 1-point vs. 2-point**:
    - No significant difference ($p \approx 0.77$).
    - **Conclusion**: Binary crossover types perform similarly.

Interpretation Guidelines

- Results with **p-values < 0.05** are considered **statistically significant**.

- For highly significant results (e.g., $p < 0.00001$), the evidence is strong that the two methods differ in performance.

- When **both t-test and Wilcoxon test agree**, the conclusion is more robust.

Function	Config1	Config2	T_STAT	T_P_VALUE	WILCOXON_STAT	WILCOXON_P_VALUE
cross_in_tray	real_arithmetic	real_blx	-3.237107497903747	0.00200509357032528 2	110.0	0.01059826090931892 4
cross_in_tray	real_arithmetic	binary_1point	5.297999044220535	7.372179094577632 5e-06	34.0	6.91786408424374e-06
cross_in_tray	real_arithmetic	binary_2point	4.928520442786390 5	2.194694830384268 5e-05	41.0	1.82446092367172 24e-05
cross_in_tray	real_blx	binary_1point	9.135202368371779	1.595414408614059 6e-10	4.0	1.303851604461667e-08
cross_in_tray	real_blx	binary_2point	8.792945933134888	3.850019415318220 4e-10	3.0	9.313225746154785e-09
cross_in_tray	binary_1point	binary_2point	-0.963296279007 3933	0.339398357567 20615	188.0	0.37074060738086 7
bukin_n6	real_arithmetic	real_blx	-0.444819066724 6721	0.658250939188 1557	192.0	0.41612989827990 53

bukin_n6	real_arithmetic	binary_1point	1.2924423581472373	0.2023788122455 0183	185.0	0.3387417849153 28
bukin_n6	real_arithmetic	binary_2point	0.8985785619954539	0.3726443250497 6697	187.0	0.3598779272288 084
bukin_n6	real_blx	binary_1point	2.1580142014128 91	0.0352190455831 2329	112.0	0.0120476223528 38516
bukin_n6	real_blx	binary_2point	1.5203173905435 854	0.1340483470076 9908	155.0	0.1141766738146 5435
bukin_n6	binary_1point	binary_2point	-0.2912760987510 4493	0.7720095142591 508	230.0	0.9676735773682 594

8. Bibliography

- numpy: NumPy
- pandas: Pandas
- matplotlib: Matplotlib
- seaborn: Seaborn
- scipy: SciPy
- optimization functions: Optimization Test Functions and Datasets