Universidad de Navarra

The Tale of the Splitting Ruby: Unlocking Its Greatest Value

PRESENTED BY:

Miguel Diaz and Dante Schrantz

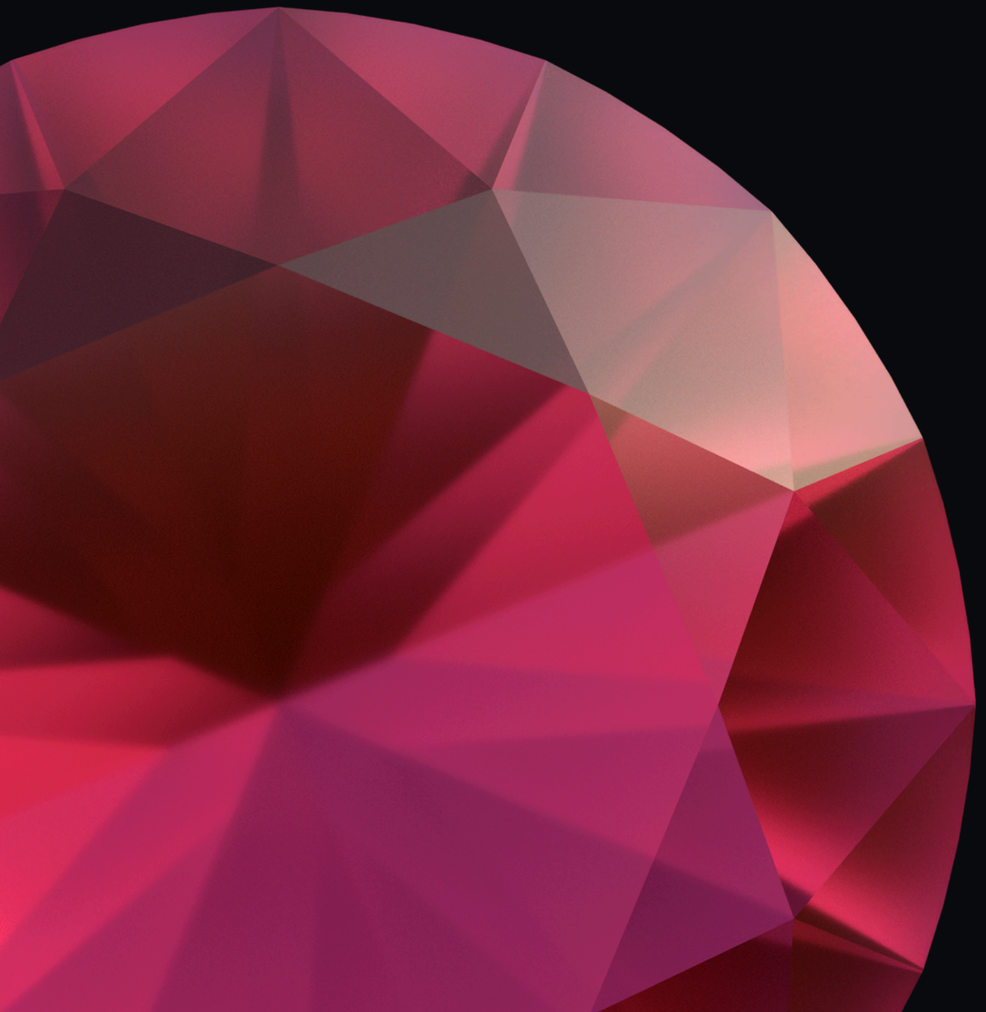
20 December, 2024

You can access the code in our personal Github profiles

The Old Man's Ruby

Once upon a time, an old man possessed a magnificent ruby. Wishing to gift his grandsons an equal treasure, he faced a curious dilemma: how to divide the ruby to ensure its value remained as magical and fair as possible.

Problem statement



An old man possesses a precious ruby weighing 2 grams. He wishes to divide it between his two grandsons, ensuring that the total value of the divided ruby is maximized. The value of each piece depends on its weight, but the division process introduces complexity: a penalty is applied if the two pieces differ too much in weight, and fairness in their distribution must also be considered.

- Value Function

The value of each piece is proportional to the square of its weight. If one piece weighs x grams, the other weighs 2−x grams. The total value of the two pieces can be expressed as:

$$V(x) = k \times x^2 + k \times (2 - x)^2$$

(where k is the proportionality constant)

- Penalty Term

To encourage fairness, a penalty is applied based on the difference in weights between the two pieces. This penalty is given by:

$$P(x) = c \times |x - (2 - x)| = c|2x - 2|$$

Problem statement

- Total Function

The total value after considering the penalty is given by:

$$T(x) = V(x) - P(x) = k \cdot x^2 + k \times (2 - x)^2 - c \times |2x - 2|$$

- Constraints. To ensure fairness and practicality, the following constraints are applied:

-Weight Non-Negativity:

$$x \geq 0$$

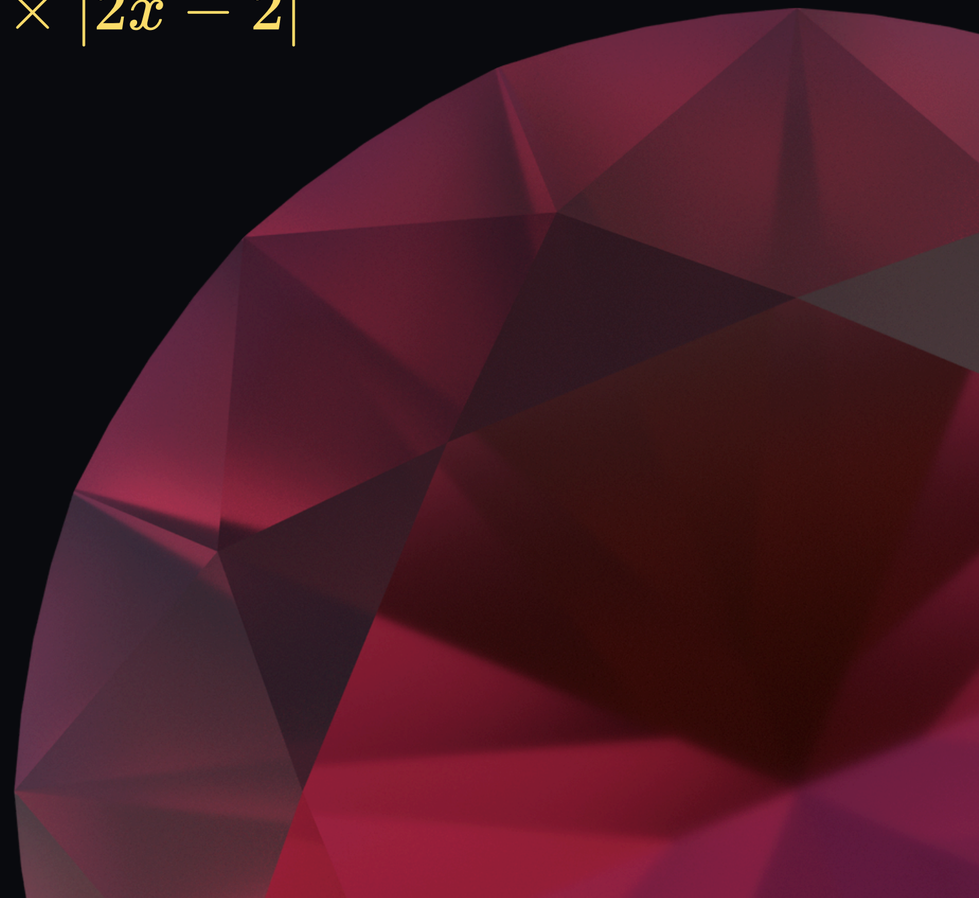
-Total Weight: (as the ruby weighs 2 grams in total)

$$x \leq 2$$

-Fairness Constraint:

The weights of the two pieces must not differ by more than a small threshold . This is expressed as:

$$|x - (2 - x)| \leqslant \varepsilon$$



The goal is to determine the weight x of one piece that maximizes the total value $T(x)$ while satisfying the fairness and practical constraints.

This ensures the grandsons receive a fair and magical gift, with the ruby's value maximized and the cutting process accounted for.

Algorithms

+ Ant Colony Optimization Algorithm

01 Blind Search

02 Grid Search

03 Monte Carlo Search

04 Hill Climbing

05 Simmulated Annealing Search

06 Tabu Search

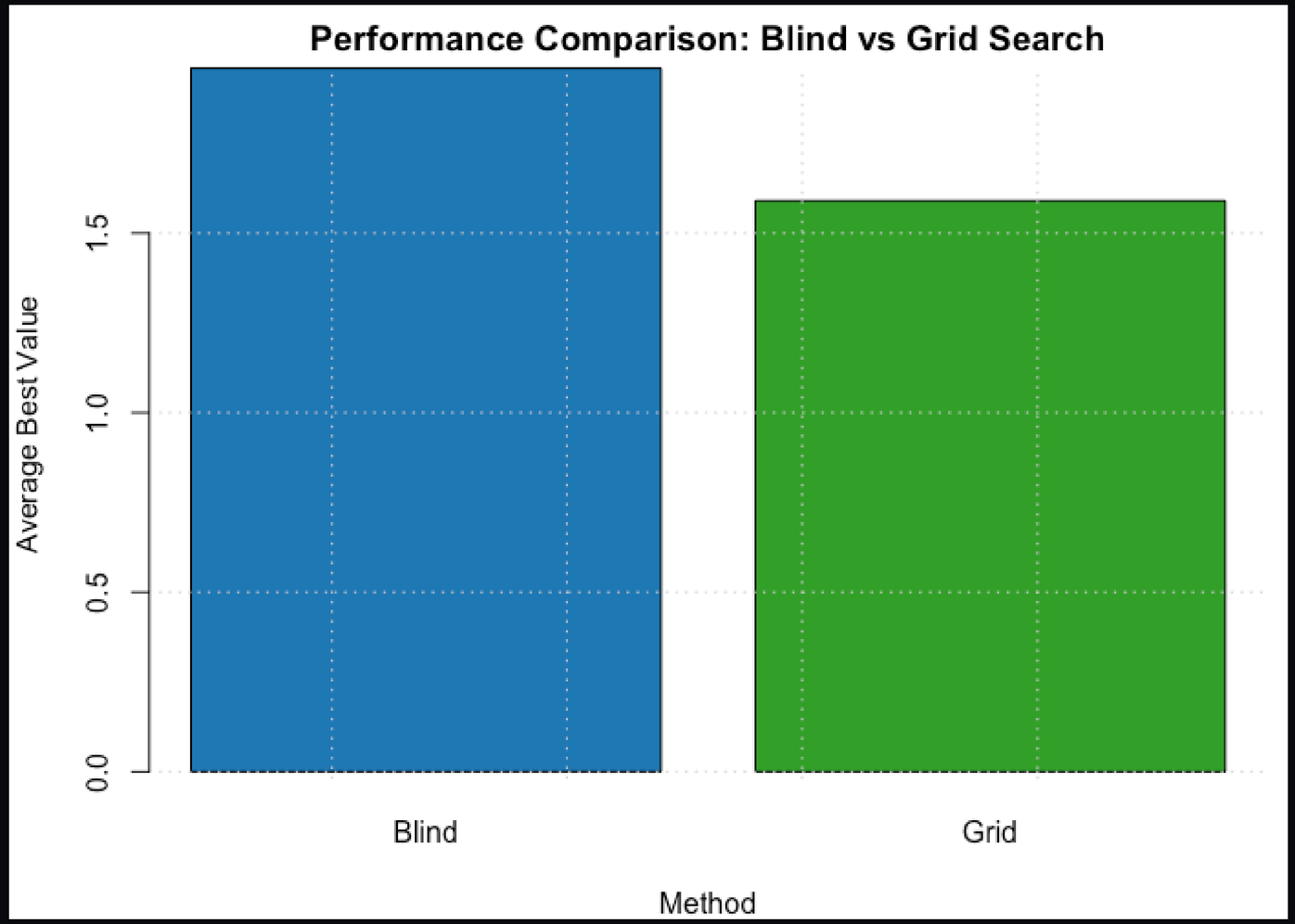
07 Genetic and Evolutionary Algorithms

08 Diferential Evolution

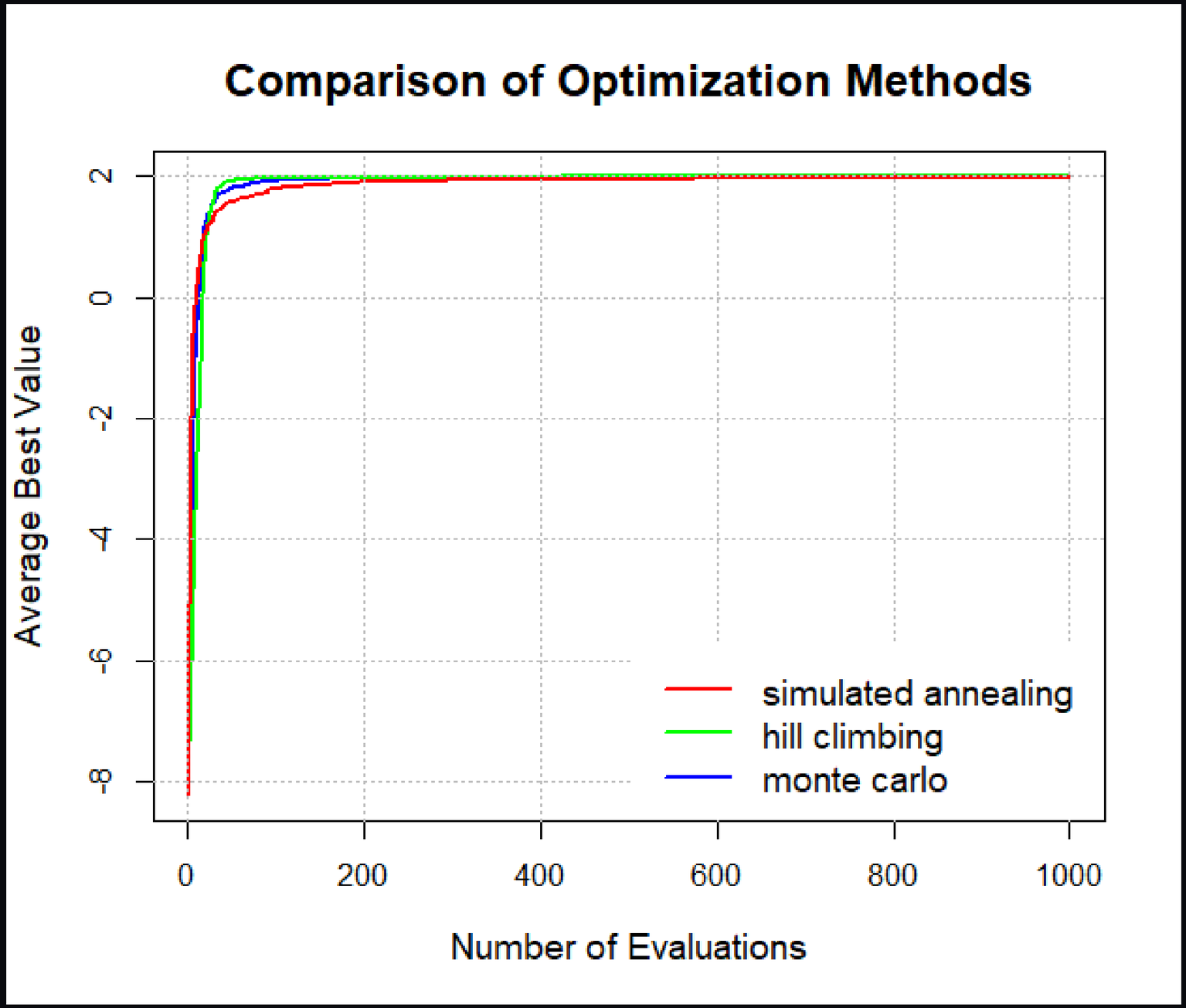
09 Particle Swarm

Grid & Blind

Method	Avg_Best	Std_Dev	Success_Rate
Blind	1.9615	0.0364	100
Grid	1.5927	0.0000	0

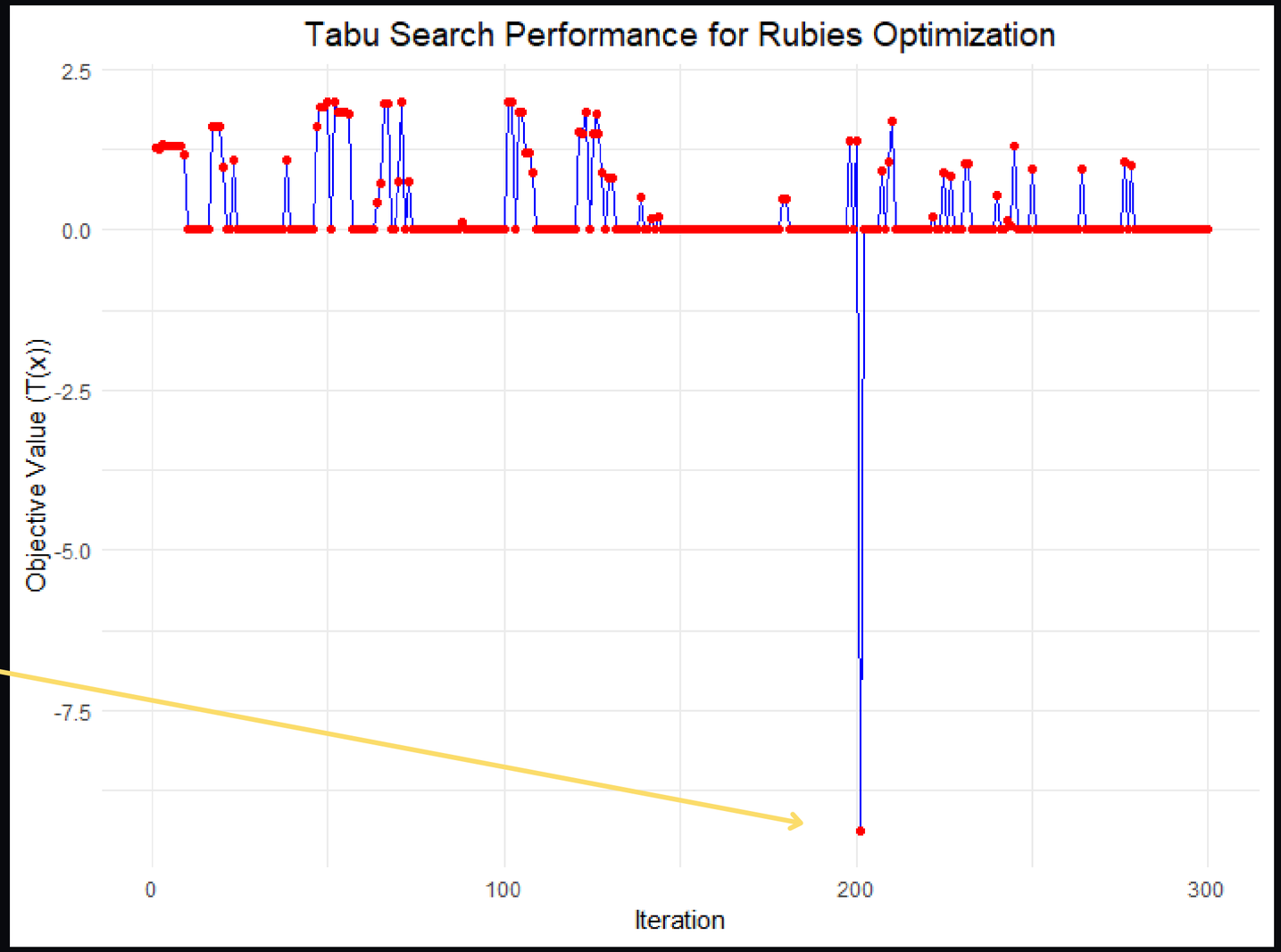


```
> print(max_values)
[1] 1.979451 1.997118 1.995546

> print(avg_iterations)
[1] 505.33 481.29 567.98

> print(fastest_method)
[1] "hill climbing"
```



```
> cat("Best configuration:", best_config, "\n")
Best configuration: 1.000107
> cat("Maximum value:", best_value, "\n")
Maximum value: 1.997864
```

The value of -7.5 observed at iteration 200 is a natural consequence of the operation of Tabu Search. It occasionally visits suboptimal solutions to escape local minima. Near the boundary of [0, 2], the penalty term P=c|2x-2| grows large, causing T(x) to drop to -7.5. This dip is temporary, and the algorithm continues exploring to find better solutions afterward.

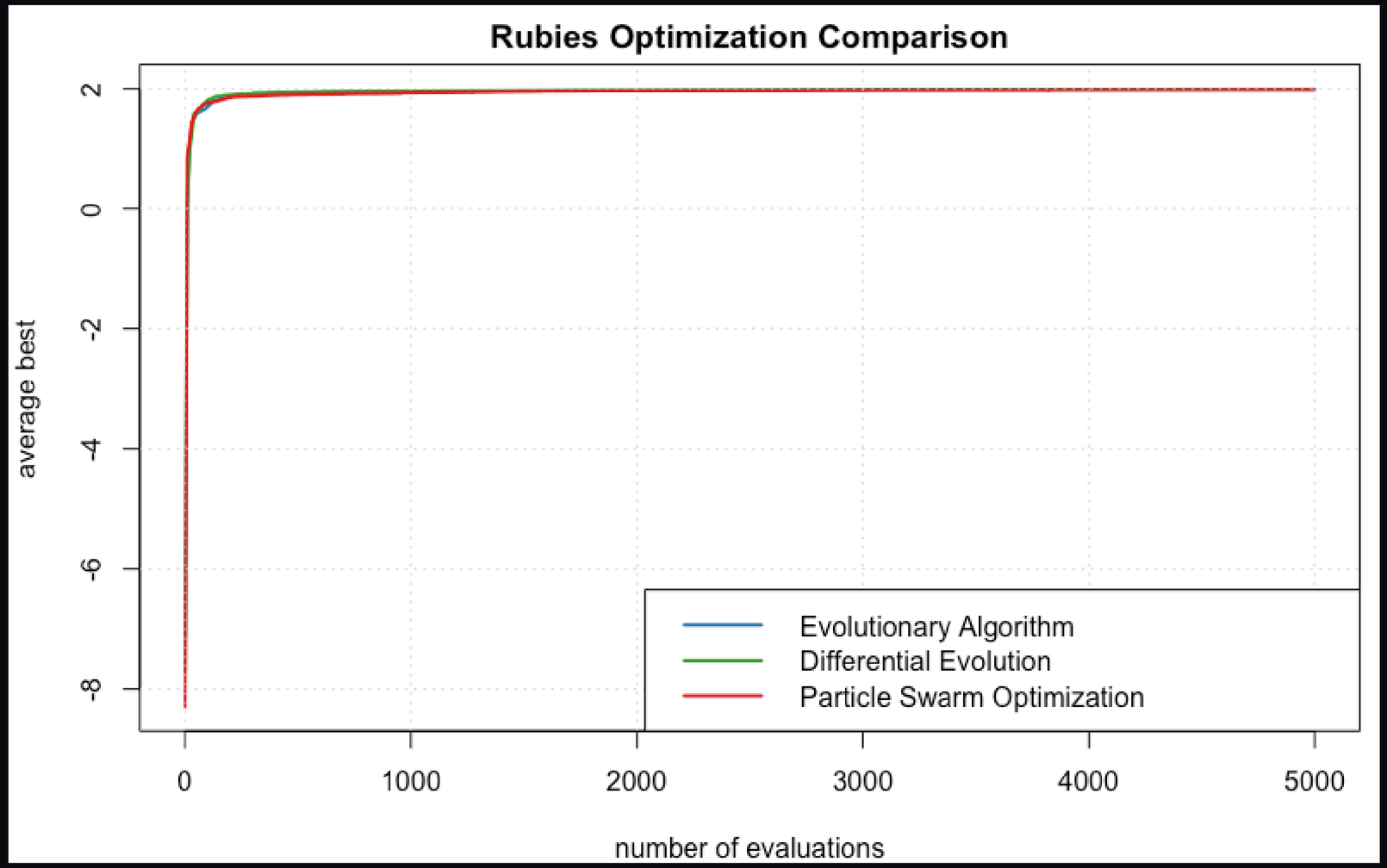


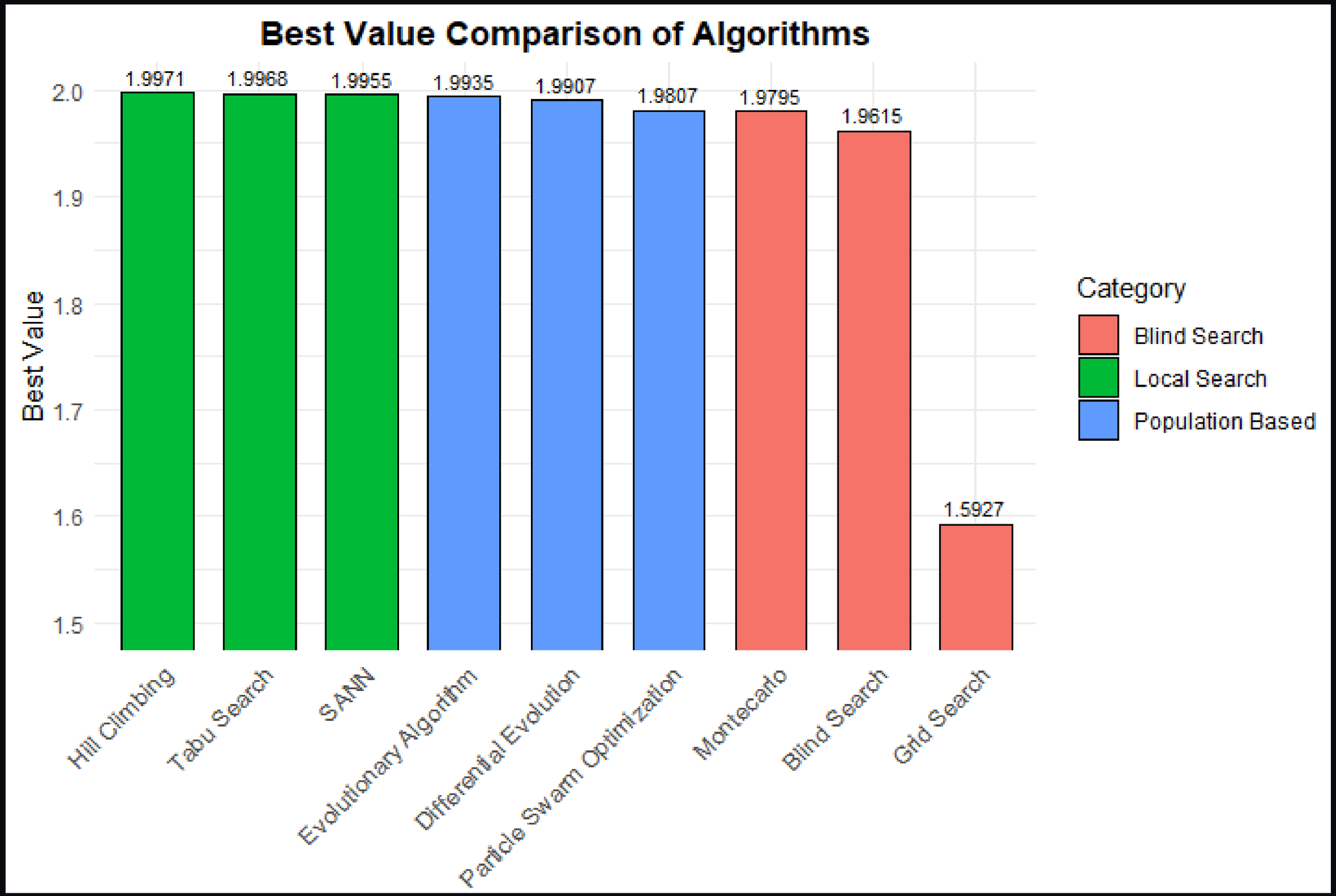
EA, DE and PSO

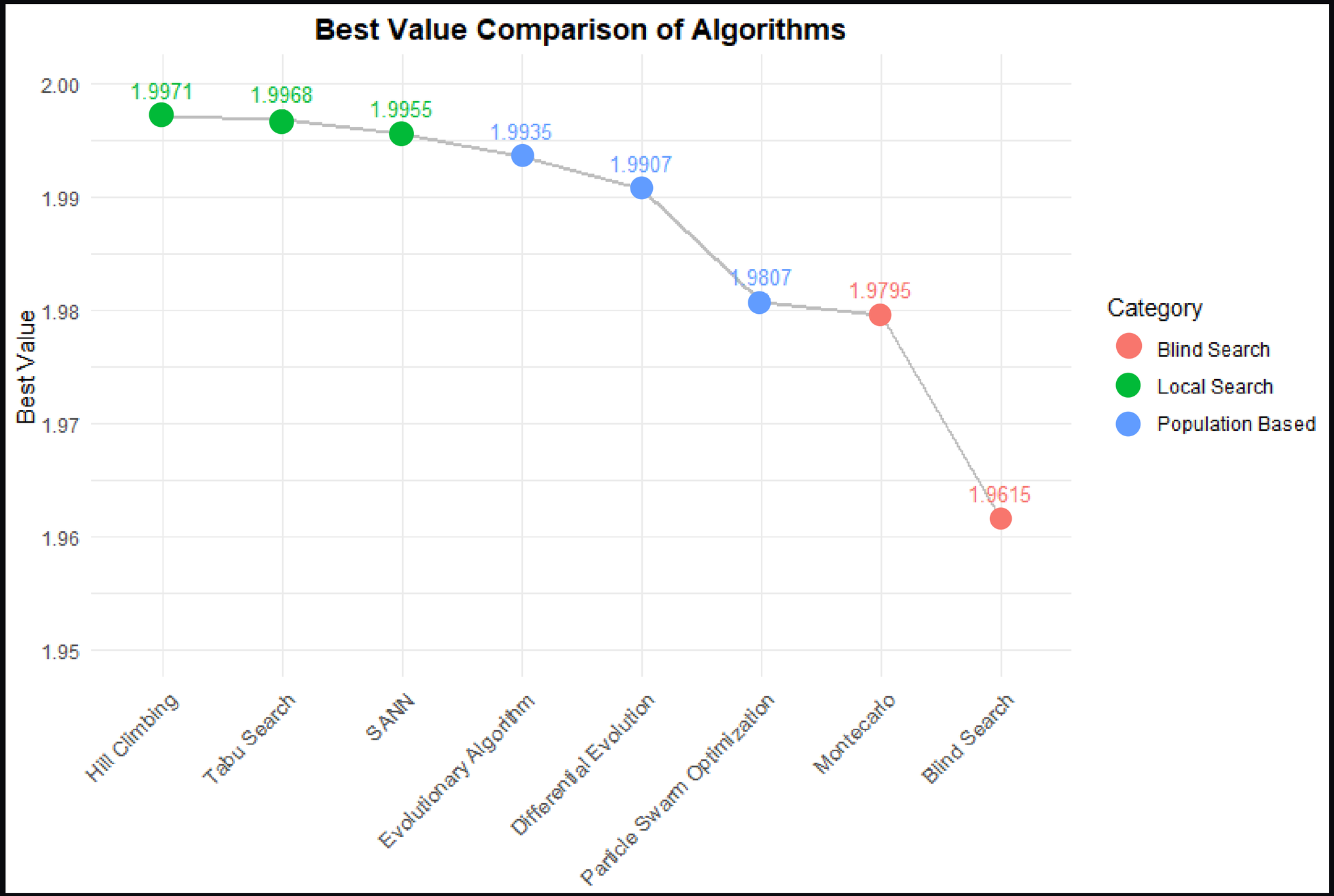
- Evolutionary Algorithm
  - Avg score - 1.9935
  - Avg Runtime - 0.03 seconds
  - Avg Iterations to Best - 2965
  - Success Rate - 100%

- Differential Evolution
  - Avg score - 1.9907
  - Avg Runtime - 0.01 seconds
  - Avg Iterations to Best - 2775
  - Success Rate - 100%

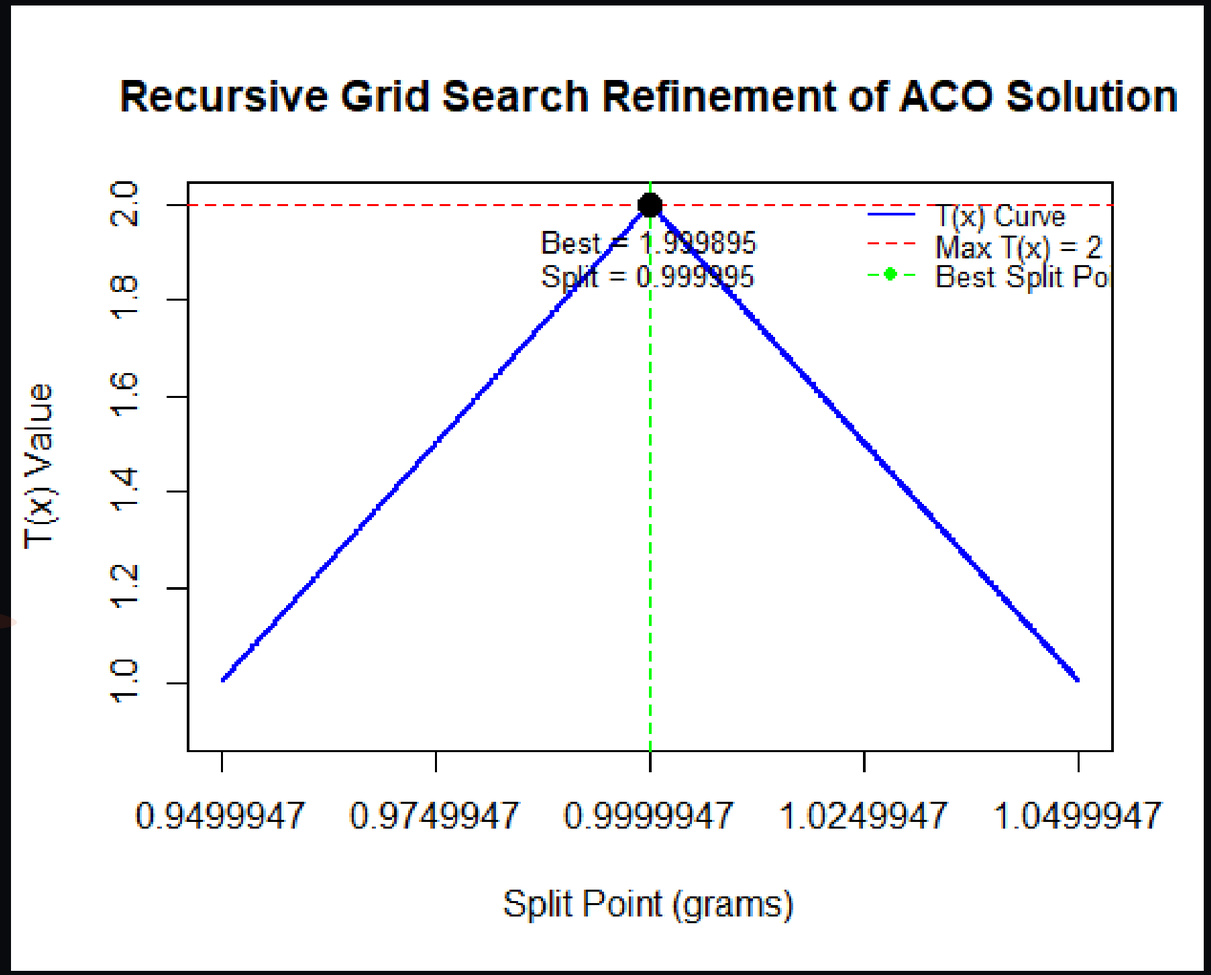
- Partical Swarm
  - Avg score - 1.9807
  - Avg Runtime - 0.05 seconds
  - Avg Iterations to Best - 2332
  - Success Rate - 90%




Best Value Comparison of Algorithms
Best Value
2.0
1.9
1.8
1.7
1.6
1.5
1.9971
1.9968
1.9955
1.9935
1.9907
1.9807
1.9795
1.9615
1.5927
Hill Climbing
Tabu Search
SANN
Evolutionary Algorithm
Differential Evolution
Particle Swarm Optimization
Montecarlo
Blind Search
Grid Search
Category
Blind Search
Local Search
Population Based

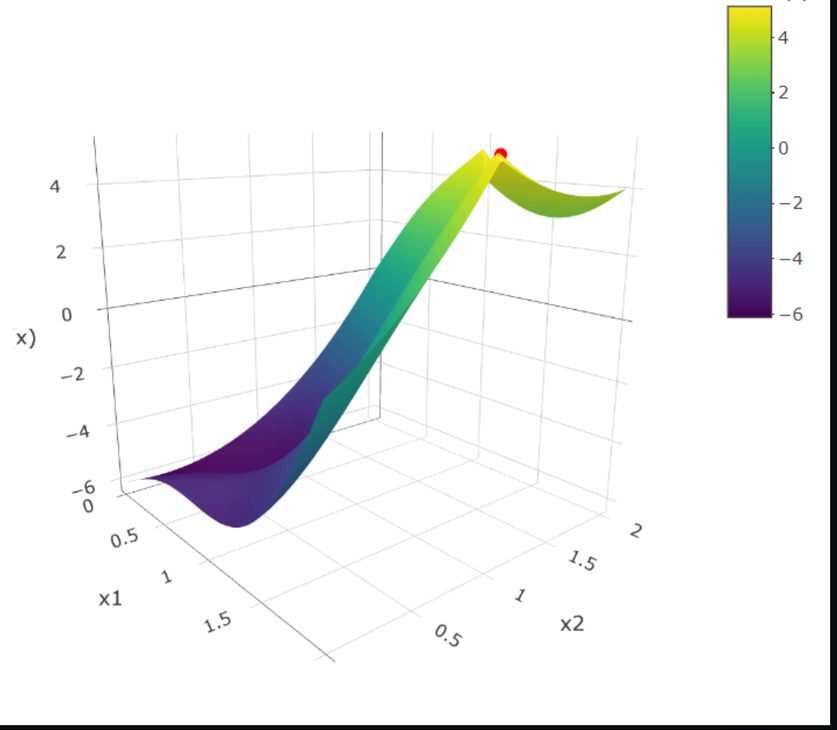

Best Value Comparison of Algorithms
Best Value
2.00
1.99
1.98
1.97
1.96
1.95
1.9971
1.9968
1.9955
1.9935
1.9907
1.9807
1.9795
1.9615
Hill Climbing
Tabu Search
SANN
Evolutionary Algorithm
Differential Evolution
Particle Swarm Optimization
Montecarlo
Blind Search
Category
Blind Search
Local Search
Population Based

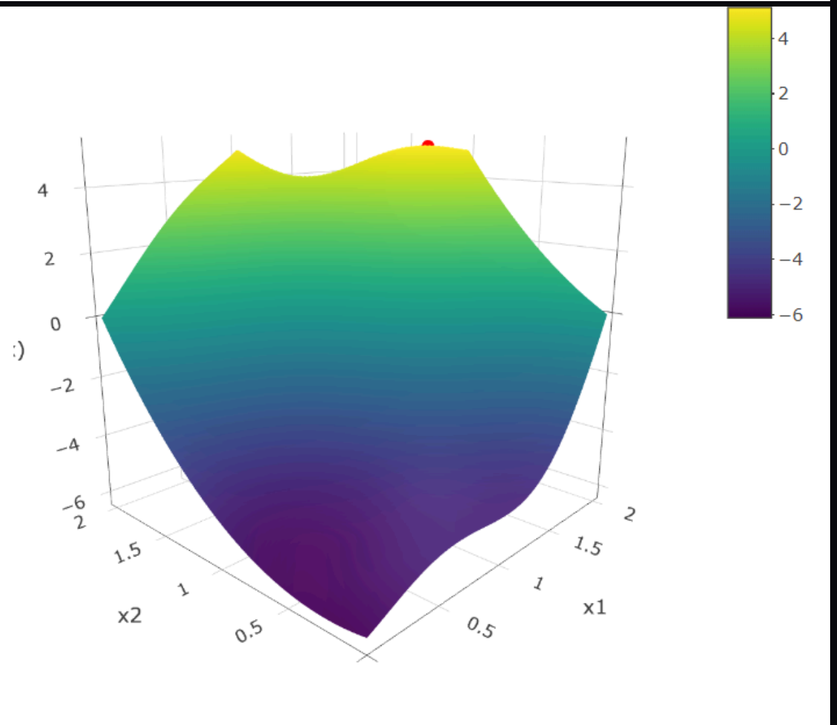
Ant Colony Optimization with Recursive Refinement



We need something more advanced, we want to look for a complex obejctive function

...





This red point represents the maximum solution identified using the ACO algorithm, showcasing its optimization capability.

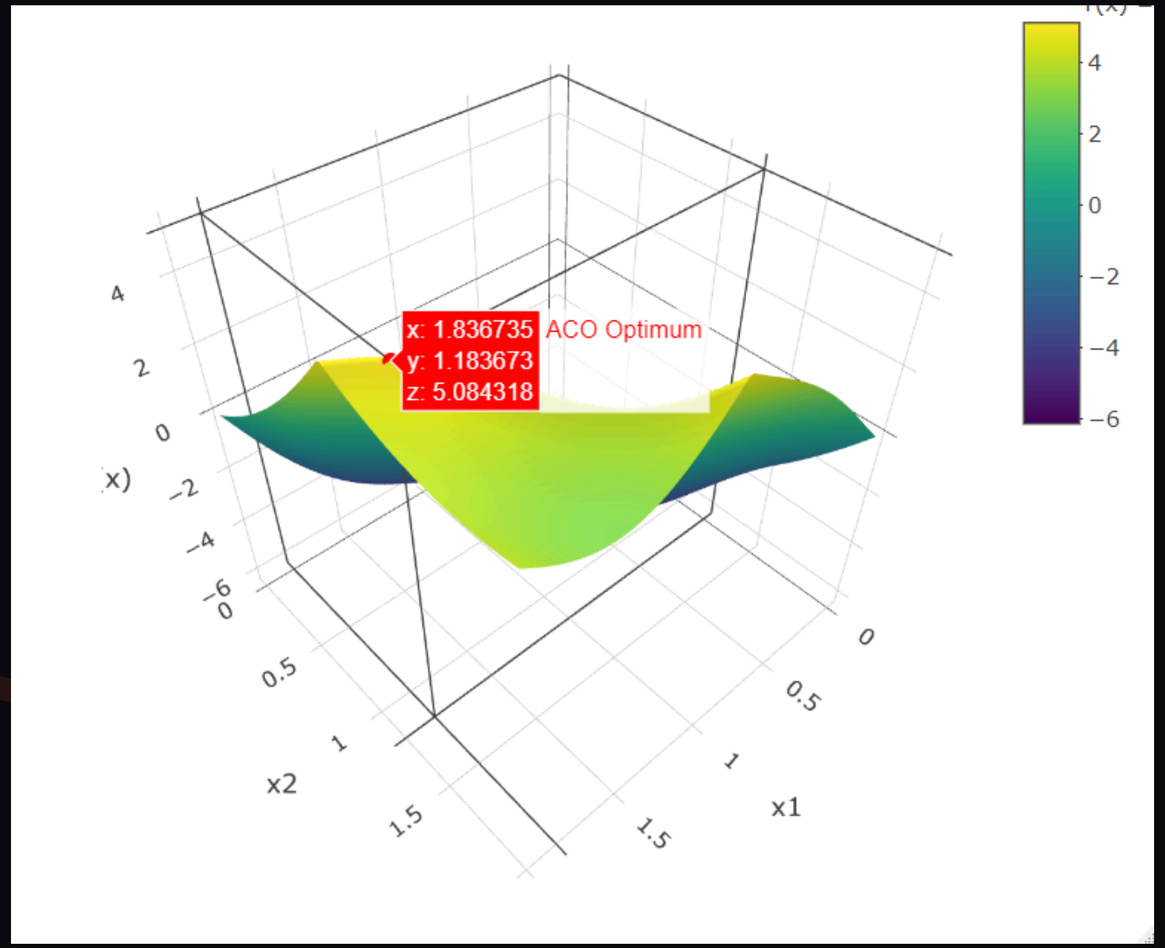
```r
rubies <- function(x) {
  k <- 1  # Value proportionality constant
  c <- 5  # Penalty constant

  # Total value
  V <- k * (x[1]^2 + x[2]^2 + (3 - x[1] - x[2])^2) +
    k * sin(pi * x[1]) * cos(pi * x[2])

  # Penalty
  P <- c * abs(3 - x[1] - x[2])

  # Objective function
  T <- V - P

  return(T)
}
```

Note: We recommend running the code in R and then downloading the file in HTML format to view it in a web browser. This will allow you to fully take advantage of the content's interactivity.

Ant Colony Optimization

```
> # Refinement using Grid Search
> refinement_range <- 0.1
> refined_result <- recursive_grid_search(ac
> cat("Refined Best Split Point:", refined_r
Refined Best Split Point: 1.816132 1.184176
> cat("Refined Maximum Value of T(x):", refi
Refined Maximum Value of T(x): 5.156234
```

```
> cat("ACO Best Split Point:", aco_re
ACO Best Split Point: 1.836735 1.1836
> cat("ACO Maximum Value of T(x):", a
ACO Maximum Value of T(x): 5.084317
```



CONCLUSION

The **combined ACO + Grid Search** approach worked effectively:
ACO provided a good initial solution near the optimal region.
Grid search refined it further to achieve a value essentially at the theoretical maximum.

- This demonstrates the strength of combining heuristic algorithms with precise local search methods!

Reach out to us for any questions.

MIGUEL DIAZ P.D.J.
github.com/migueldiazpdj

DANTE SCHRANTZ
github.com/DanteSc03