# Rice Classi ication Report

Hridoy Rahman
February 16, 2024

## 1 Summary

The rice classification problem aims to predict the class of Cammeo rice and Osmancik rice, which is being analyzed based on a dataset with summary measurements of rice derived from image data. The dataset's attributes are Area, Perimeter, Major and Minor Axis length, Eccentricity, Convex Area, Extent, and which class it belongs to. The dataset contains 3,810 records. Eight attributes denote each row, and its class denotes the type of rice. The rice classification problem using GP involves evolving mathematical expressions using a given gp language consisting of functions (Arithmetic and other required functions) and terminals provided within the dataset (7 attributes), and the end goal is to evolve an S-expression that predicts the rice class. Overall, the rice classification problem using GP involves using the dataset's attributes as inputs to evolve mathematical expressions that can classify the rice type.

| | |
|---|---|
| Objective | Search for a function that correctly classifies Cammeo Rice and Osmancik Rice based on the given data inputs. |
| Terminal Set | Area, Perimeter, Major and Minor Axis Length, Eccentricity, Convex_Area, Extent |
| Function Set | +, -, *, (Protected Division) %, Exp, Log, Ephemeral |
| Fitness Cases | provided dataset with 3,810 data points. Random seed training sets vary like 0.6% of the 3810 is used in one case. |
| Raw Fitness | The number of training cases classified into the correct class. |
| Standardized Fitness | The total number of training cases - (minus) raw fitness. |
| Hits | number of hits are the same as raw fitness. |
| Success Predicate | An S-expressions that score close to the dataset length's hits. |

Table 1: Table of Rice Classification

### 1.1 Experiment Details

### 1.2 Table of GP Parameters

Here are the GP parameters used for the Rice Classification problem.

| Parameter | Value |
|---|---|
| Population Size | 1024 |
| Generation Size | 51 |
| Crossover & Mutation Rate ( Variation 1 without Elitism) | 90% & 10% |
| Crossover & Mutation Rate ( Variation 2 without Elitism ) | 100% & 100% |
| Crossover & Mutation Rate ( Variation 3 with Elitism) | 90% & 10% |
| Crossover & Mutation Rate ( Variation 4 with Elitism ) | 100% & 100% |
| Selection Method | Tournament Selection |
| Tournament Size | 3 |
| Elites | 10 |
| Runs (Jobs) | 10 |

Table 2: Table of Parameters Used

### 1.3 Fitness Function

Fitness Cases: A dataset with 3,810 data points was provided. Random seed training sets vary like 0.6% of the 3810 is used in one case.

Raw Fitness: The number of training cases classified into the correct class.
Standardized Fitness The total number of training cases - (minus) raw fitness.

```
if (gp_ans >= 0.0 && currentAns.equals("Cammeo"))
    hits++;
else if (gp_ans < 0.0 && currentAns.equals("Osmancik"))
    hits++;
sum = hits;
```

The fitness function takes an answer from the evaluated GP tree for the provided individual and checks if the $gp\_ans >= 0.0$, and if the current individual is classed as Cammeo, then it increases the hit number. Otherwise, if the $gp\_ans < 0.0$ equals the Osmancik rice class, then it also increments the hit number.

### 1.4 GP Language

Terminal Set: Area, Perimeter, Major and Minor Axis Length, Eccentricity, Convex_Area, Extent
Function Set: +, -, *, (Protected Division) %, Exp, Log, Ephemeral

### 1.5 Data Format

- RELATION Rice_Cammeo_Osmancik
- ATTRIBUTE Area Integer
- ATTRIBUTE Perimeter Real
- ATTRIBUTE Major_Axis_Length Real

- ATTRIBUTE Minor_Axis_Length Real
- ATTRIBUTE Eccentricity Real
- ATTRIBUTE Convex_Area Integer
- ATTRIBUTE Extent Real
- ATTRIBUTE Class {Cammeo, Osmancik}

### 1.6 Variations of Experiments

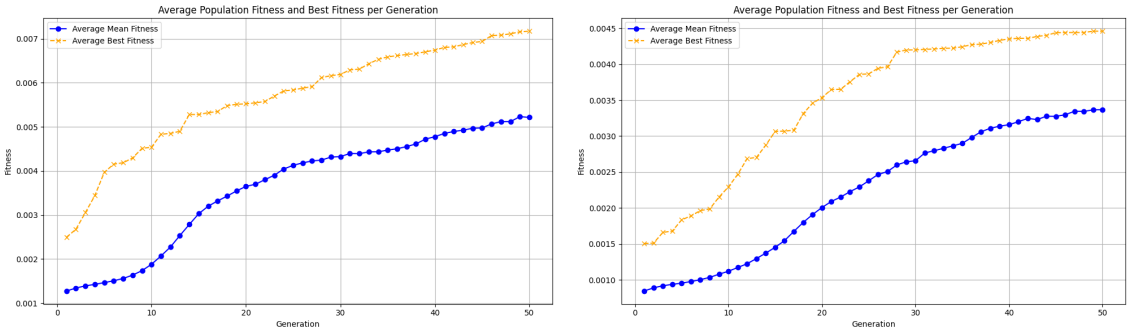
| | |
|---|---|
| Training Set & Testing Set | 0.6% & 0.4% of the actual dataset |
| Crossover & Mutation Rate ( Variation 1 without Elitism) | 90% & 10% |
| Crossover & Mutation Rate ( Variation 2 without Elitism ) | 100% & 100% |
| Crossover & Mutation Rate ( Variation 3 with Elitism) | 90% & 10% with 10 elites |
| Crossover & Mutation Rate ( Variation 4 with Elitism ) | 100% & 100% with 10 elites |

Table 3: Table of Parameters Used

## 2 Results

### 2.1 Tables of Result

The following data table is the data gathered from the testing set. The testing set performs better with variation 3, parameters with 90% crossover rate, 10% mutation rate, and 10 elites. The testing set performs better as it is clearly visible in the graph.




(a) Testing Set: (Parameters) 90% & 10% with Elitism

(b) Training Set: (Parameters) 90% & 10% with Elitism

Figure 1: Average of Mean and best Fitness plotted in a graph across 10 runs

Table 4: Testing Set: Average Mean and Best Fitness per Generation Across 10 Runs

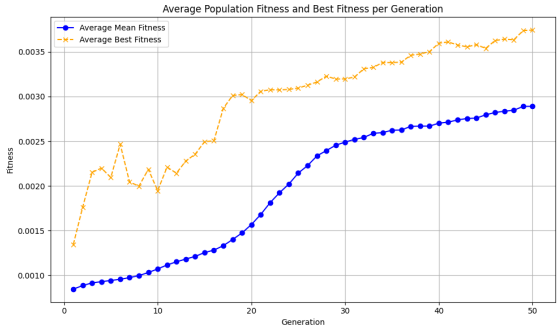
| Generation | Average Mean Fitness | Average Best Fitness |
|---|---|---|
| 0 | 0.001273 | 0.002493 |
| 1 | 0.001339 | 0.002674 |
| 2 | 0.001389 | 0.003052 |
| 3 | 0.001423 | 0.003438 |
| 4 | 0.001462 | 0.003965 |
| 5 | 0.001505 | 0.004152 |
| 6 | 0.001557 | 0.004184 |
| 7 | 0.001633 | 0.004293 |
| 8 | 0.001735 | 0.004511 |
| 9 | 0.001879 | 0.004535 |
| 10 | 0.002067 | 0.004829 |
| 11 | 0.002271 | 0.004849 |
| 12 | 0.002531 | 0.004898 |
| 13 | 0.002782 | 0.005277 |
| 14 | 0.003026 | 0.005280 |
| 15 | 0.003198 | 0.005319 |
| 16 | 0.003317 | 0.005347 |
| 17 | 0.003426 | 0.005480 |
| 18 | 0.003544 | 0.005510 |
| 19 | 0.003647 | 0.005521 |
| 20 | 0.003695 | 0.005543 |
| 21 | 0.003799 | 0.005573 |
| 22 | 0.003900 | 0.005697 |
| 23 | 0.004042 | 0.005814 |
| 24 | 0.004125 | 0.005836 |
| 25 | 0.004181 | 0.005877 |
| 26 | 0.004224 | 0.005912 |
| 27 | 0.004245 | 0.006118 |
| 28 | 0.004313 | 0.006160 |
| 29 | 0.004322 | 0.006188 |
| 30 | 0.004394 | 0.006292 |
| 31 | 0.004390 | 0.006309 |
| 32 | 0.004430 | 0.006430 |
| 33 | 0.004433 | 0.006531 |
| 34 | 0.004470 | 0.006586 |
| 35 | 0.004502 | 0.006615 |
| 36 | 0.004550 | 0.006641 |
| 37 | 0.004612 | 0.006663 |
| 38 | 0.004720 | 0.006702 |
| 39 | 0.004775 | 0.006741 |
| 40 | 0.004848 | 0.006801 |
| 41 | 0.004894 | 0.006815 |
| 42 | 0.004922 | 0.006865 |
| 43 | 0.004968 | 0.006912 |
| 44 | 0.004973 | 0.006937 |
| 45 | 0.005066 | 0.007069 |
| 46 | 0.005117 | 0.007082 |
| 47 | 0.005118 | 0.007110 |
| 48 | 0.005230 | 0.007156 |
| 49 | 0.005213 | 0.007171 |

Table 5: Training Set: Average Mean Fitness and Average Best Fitness per Generation

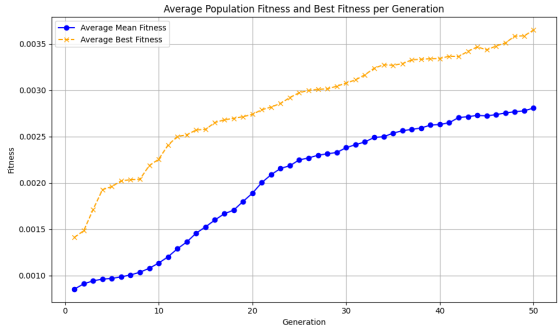
| Generation | Average Mean Fitness | Average Best Fitness |
|---|---|---|
| 0 | 0.000848 | 0.001508 |
| 1 | 0.000891 | 0.001513 |
| 2 | 0.000921 | 0.001662 |
| 3 | 0.000940 | 0.001680 |
| 4 | 0.000957 | 0.001836 |
| 5 | 0.000979 | 0.001892 |
| 6 | 0.001005 | 0.001965 |
| 7 | 0.001037 | 0.001988 |
| 8 | 0.001081 | 0.002153 |
| 9 | 0.001122 | 0.002293 |
| 10 | 0.001174 | 0.002469 |
| 11 | 0.001226 | 0.002687 |
| 12 | 0.001299 | 0.002704 |
| 13 | 0.001375 | 0.002874 |
| 14 | 0.001455 | 0.003066 |
| 15 | 0.001547 | 0.003070 |
| 16 | 0.001676 | 0.003082 |
| 17 | 0.001801 | 0.003314 |
| 18 | 0.001910 | 0.003458 |
| 19 | 0.002007 | 0.003535 |
| 20 | 0.002092 | 0.003647 |
| 21 | 0.002153 | 0.003651 |
| 22 | 0.002228 | 0.003756 |
| 23 | 0.002294 | 0.003858 |
| 24 | 0.002380 | 0.003865 |
| 25 | 0.002467 | 0.003944 |
| 26 | 0.002507 | 0.003963 |
| 27 | 0.002599 | 0.004167 |
| 28 | 0.002643 | 0.004197 |
| 29 | 0.002658 | 0.004198 |
| 30 | 0.002766 | 0.004205 |
| 31 | 0.002797 | 0.004212 |
| 32 | 0.002831 | 0.004220 |
| 33 | 0.002865 | 0.004223 |
| 34 | 0.002900 | 0.004242 |
| 35 | 0.002981 | 0.004271 |
| 36 | 0.003060 | 0.004278 |
| 37 | 0.003109 | 0.004301 |
| 38 | 0.003138 | 0.004330 |
| 39 | 0.003159 | 0.004351 |
| 40 | 0.003200 | 0.004359 |
| 41 | 0.003246 | 0.004359 |
| 42 | 0.003230 | 0.004385 |
| 43 | 0.003276 | 0.004401 |
| 44 | 0.003275 | 0.004437 |
| 45 | 0.003296 | 0.004440 |
| 46 | 0.003346 | 0.004440 |
| 47 | 0.003344 | 0.004440 |
| 48 | 0.003365 | 0.004458 |
| 49 | 0.003368 | 0.004459 |

## 2.2 Performance Plots

These are the plots of each run, including four variations, each with 10 runs. The fitness wasn't adjusted. The best hits are generated by the parameters 90% and 10% crossover, and mutation rate, with 10 elitism, provides 1409 as the best result with a testing set with 1524 data points. The training set consists of 2286 data points, for which the best hits gotten from running the evaluation tree is 2125 hits, which performs better as well. The following expression provides the best result of the testing set. [ Appendix A, 4.1 ] The following expression provides the best results in the training set. [ Appendix A, 4.4 ]

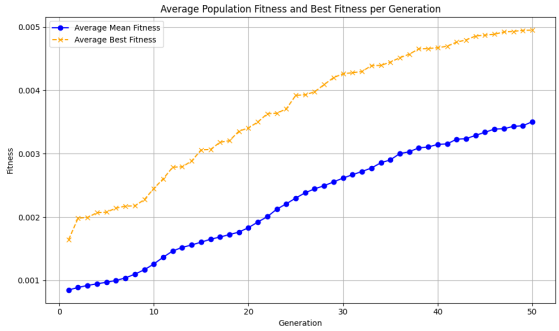



(a) Training Set: (Parameters) 90% & 10% without Elitism

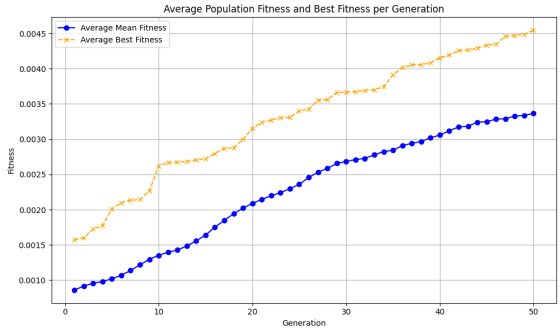



(b) Training Set: (Parameters) 100% & 100% without Elitism

Figure 2: Average of Mean and best Fitness plotted in a graph across 10 runs




(a) Training Set: (Parameters) 90% & 10% with Elitism




(b) Training Set: (Parameters) 100% & 100% with Elitism

Figure 3: Average of Mean and best Fitness plotted in a graph across 10 runs

## 2.3 Best GP Expressions

```
The following GP expression is the best one obtained from the testing set.
Best Individual of Run:
Subpopulation 0:
Evaluated: true
Fitness: Standardized=115.0 Adjusted=0.008620689655172414 Hits=1409
Tree 0:
 (/ 1.0 (+ (* (/ (* (/ (+ eccentricity perimeter)
     (* (rlog (exp (- (- minor_axis major_axis)
```

```
        (rlog major_axis)))) convex_area)) perimeter)
    (* eccentricity convex_area)) perimeter)
    (* (* (rlog minor_axis) (* (- (+ (/ area
        major_axis) area) (/ (+ eccentricity area)
        (rlog major_axis))) (/ (exp (- (- minor_axis
        major_axis) (* minor_axis (* extent (+ (-
        (rlog eccentricity) (+ area convex_area))
        (* (- (+ (/ area major_axis) area) (/ (+
            eccentricity area) (rlog major_axis))) perimeter))))))
        (rlog (exp (- minor_axis minor_axis))))))
        (/ (rlog major_axis) (rlog (exp (- (- minor_axis
            major_axis) (* (/ (rlog major_axis) convex_area)
            (* extent (+ (- (rlog (rlog eccentricity))
                (+ area convex_area)) (* (- (rlog (* (rlog
                minor_axis) (exp minor_axis))) (/ extent
                convex_area)) (/ (rlog major_axis) (rlog
                (exp (- minor_axis major_axis))))))))))))))
```

```
The best GP Expression obtained from the Training Set:
Best Individual of Run:
Subpopulation 0:
Evaluated: true
Fitness: Standardized=161.0 Adjusted=0.006172839506172839 Hits=2125
Tree 0:
 (* (rlog (/ (rlog (/ (- (- (- (- (rlog (-
     (- (exp (rlog area)) (rlog extent)) (/ minor_axis
     perimeter))) (* perimeter perimeter)) (exp
     (* eccentricity minor_axis))) (rlog area))
     (rlog (/ (- (- (exp (rlog area)) (rlog area))
         (exp (* eccentricity minor_axis))) (/ (*
         (exp eccentricity) (rlog eccentricity)) (rlog
         (/ (- (- (rlog 1.0) (* perimeter perimeter))
             (exp (/ (* major_axis major_axis) perimeter)))
             (/ (* extent (rlog (- (rlog major_axis) minor_axis)))
                 (exp (rlog area))))))))) (/ (* (exp eccentricity)
     (rlog (- (/ (* major_axis major_axis) perimeter)
         minor_axis))) (rlog (/ (- (- (* perimeter
     perimeter) (* perimeter perimeter)) (exp
     (/ (* major_axis major_axis) perimeter)))
     (/ (* extent (rlog (- (rlog major_axis) (/
         (* major_axis major_axis) perimeter)))) (exp
         (rlog area)))))))) (- (- (exp extent) minor_axis)
     (exp (rlog (- (rlog (- (- (exp (rlog extent))
         (rlog extent)) (rlog (rlog (exp (rlog area))))))
         (- minor_axis 1.0))))))) (exp (/ (* (exp
```

```
eccentricity) (rlog (- (/ (* extent (rlog
(- (rlog major_axis) minor_axis))) (/ (-
(- (* major_axis major_axis) eccentricity)
(exp (/ (* major_axis major_axis) perimeter)))
(/ (* extent (rlog (- (rlog major_axis) minor_axis)))
    (exp (rlog area))))) (rlog area)))) (rlog
(/ (- (- (rlog 1.0) (* perimeter perimeter))
    (exp (/ (* major_axis major_axis) perimeter)))
    (/ (* extent (rlog (exp (rlog area)))) (exp
        (rlog area)))))))
```

### 2.4 Variations of Experiments

The experiment is done with 4 different variations of parameters. One dataset is divided into a training set and a testing set, where the dataset is shuffled randomly and then divided into a training set and testing set on each run. The variation 1 and 2 parameter set from ***Table3*** are experimented without any elitism, and variation 3 and 4 parameter set uses elitism where the elite number is 10.

### 2.5 Analytical Discussion of Results

From the training set, Variation 3 with parameters 90% crossover rate and 10% mutation with 10 elites possess the best results. Also, in the training set, variation 3 parameters 90% crossover rate and 10% mutation with 10 elites possess the best results. Best results in the case of this Rice Classification problem means, the number of training cases classified into the correct class. In this case, variation 3 has the maximum number of hits out of all the parameters tested within both the training and testing sets.

## 3 Conclusions

The Rice Classification problem can be solved using the Genetic Programming approach. It is indeed a challenging task to predict the correct class. The implementation involved using GP language using function set and terminal set, which helped with address the classification problem. The fitness function helped with correctly classifying the rice from the dataset. However, with the GP approach, deducing the number of classes correctly is possible. After trying various parameter sets, I found that variation 3 has the best results with the maximum number of hits in the training and testing set.

***FutureWork :*** The terminal set can be increased. We could use neural networks for rice classification problem.

## 4 Appendix A

### 4.1

`Testing Set:`

```
Best Individual of Run:
Subpopulation 0:
Evaluated: true
Fitness: Standardized=115.0 Adjusted=0.008620689655172414 Hits=1409
Tree 0:
 (/ 1.0 (+ (* (/ (* (/ (+ eccentricity perimeter)
     (* (rlog (exp (- (- minor_axis major_axis)
         (rlog major_axis)))) convex_area)) perimeter)
     (* eccentricity convex_area)) perimeter)
     (* (* (rlog minor_axis) (* (- (+ (/ area
         major_axis) area) (/ (+ eccentricity area)
         (rlog major_axis))) (/ (exp (- (- minor_axis
         major_axis) (* minor_axis (* extent (+ (-
         (rlog eccentricity) (+ area convex_area))
         (* (- (+ (/ area major_axis) area) (/ (+
             eccentricity area) (rlog major_axis))) perimeter))))))
         (rlog (exp (- minor_axis minor_axis))))))
         (/ (rlog major_axis) (rlog (exp (- (- minor_axis
             major_axis) (* (/ (rlog major_axis) convex_area)
             (* extent (+ (- (rlog (rlog eccentricity))
                 (+ area convex_area)) (* (- (rlog (* (rlog
                 minor_axis) (exp minor_axis))) (/ extent
                 convex_area)) (/ (rlog major_axis) (rlog
                 (exp (- minor_axis major_axis))))))))))))))
```

### 4.2

```
Training Set:
Best Individual of Run:
Subpopulation 0:
Evaluated: true
Fitness: Standardized=195.0 Adjusted=0.00510204081632653 Hits=2091
Tree 0:
 (/ eccentricity (/ (+ (/ eccentricity (rlog
     major_axis)) (/ area (* eccentricity area)))
     (- (- (- eccentricity (- (- (* (rlog (- perimeter
         (+ (/ eccentricity (rlog major_axis)) (/
             (/ (* area convex_area) (* eccentricity area))
             extent)))) (- major_axis minor_axis)) (exp
         (rlog perimeter))) (- (- (* extent perimeter)
         (/ (- extent area) major_axis)) (- (- (*
         (rlog (- perimeter (+ (- eccentricity (/
             area (* eccentricity area))) (- 1.0 area))))
         (- major_axis minor_axis)) (exp (rlog perimeter)))
         (exp (rlog perimeter)))))) (/ (- minor_axis
```

```
        perimeter) eccentricity)) (- convex_area
        1.0))))
```

### 4.3

```
Training Set:
Best Individual of Run:
Subpopulation 0:
Evaluated: true
Fitness: Standardized=177.0 Adjusted=0.0056179775280898875 Hits=2109
Tree 0:
 (* (exp (+ (/ (/ (exp (+ (/ (/ minor_axis
     convex_area) (- (+ (rlog eccentricity) minor_axis)
     (rlog eccentricity))) (/ (+ (/ (/ (+ convex_area
     minor_axis) eccentricity) (/ minor_axis major_axis))
     (* (exp (+ (/ minor_axis convex_area) (-
         minor_axis major_axis))) minor_axis)) convex_area)))
     convex_area) (/ (/ minor_axis convex_area)
     (- (+ (rlog eccentricity) (+ minor_axis (/
         0.0 eccentricity))) (+ (rlog (- (+ (rlog
         eccentricity) (- (+ (rlog eccentricity) (+
         minor_axis (/ (* (- major_axis extent) (*
         0.0 0.0)) eccentricity))) (+ (rlog (- 0.0
         (- convex_area minor_axis))) (rlog area))))
         (- convex_area minor_axis))) (rlog area)))))
     (- minor_axis major_axis))) (rlog eccentricity))
```

### 4.4

```
Training Set:
Best Individual of Run:
Subpopulation 0:
Evaluated: true
Fitness: Standardized=161.0 Adjusted=0.006172839506172839 Hits=2125
Tree 0:
 (* (rlog (/ (rlog (/ (- (- (- (- (rlog (-
     (- (exp (rlog area)) (rlog extent)) (/ minor_axis
     perimeter))) (* perimeter perimeter)) (exp
     (* eccentricity minor_axis))) (rlog area))
     (rlog (/ (- (- (exp (rlog area)) (rlog area))
         (exp (* eccentricity minor_axis))) (/ (*
         (exp eccentricity) (rlog eccentricity)) (rlog
```

```
    (/ (- (- (rlog 1.0) (* perimeter perimeter))
        (exp (/ (* major_axis major_axis) perimeter)))
        (/ (* extent (rlog (- (rlog major_axis) minor_axis)))
            (exp (rlog area))))))))) (/ (* (exp eccentricity)
(rlog (- (/ (* major_axis major_axis) perimeter)
    minor_axis))) (rlog (/ (- (- (* perimeter
perimeter) (* perimeter perimeter)) (exp
(/ (* major_axis major_axis) perimeter)))
(/ (* extent (rlog (- (rlog major_axis) (/
    (* major_axis major_axis) perimeter)))) (exp
    (rlog area)))))))) (- (- (exp extent) minor_axis)
(exp (rlog (- (rlog (- (- (exp (rlog extent))
    (rlog extent)) (rlog (rlog (exp (rlog area))))))
    (- minor_axis 1.0))))))) (exp (/ (* (exp
eccentricity) (rlog (- (/ (* extent (rlog
(- (rlog major_axis) minor_axis))) (/ (-
(- (* major_axis major_axis) eccentricity)
(exp (/ (* major_axis major_axis) perimeter)))
(/ (* extent (rlog (- (rlog major_axis) minor_axis)))
    (exp (rlog area))))) (rlog area)))) (rlog
(/ (- (- (rlog 1.0) (* perimeter perimeter))
    (exp (/ (* major_axis major_axis) perimeter)))
    (/ (* extent (rlog (exp (rlog area)))) (exp
        (rlog area))))))))
```

# 5 Bibliography

## References


[1] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press, 1992.

[2] R. Poli, W. B. Langdon, N. F. McPhee, and J. R. Koza, *A Field Guide to Genetic Programming*, Lulu.com, 2008.

[3] Sean Luke et al., *ECJ: The Evolutionary Computation Toolkit User's Guide*, https://cs.gmu.edu/~eclab/projects/ecj/, Accessed on February 16, 2024.