Homework 4 - Solving the Knapsack problem by an advanced iterative method

Problem Statement

The 0/1 Knapsack problem is a classic combinatorial optimization problem where the goal is to select a subset of items to maximize total value while ensuring that the total weight does not exceed a given limit. Due to its computational complexity, particularly for larger problem instances, heuristic methods like Genetic Algorithms are well-suited for finding near-optimal solutions efficiently.

Description of the algorithm

For this homework I choose Genetic Algorithm as an advanced iterative method. I choose Genetic Algorithm because it is inspired by the process of natural selection, and it is interesting to see how examples from real world can influence algorithms and combinatorial problems. It is interesting to see how with each iteration algorithm crosses its population and goes closer to the optimal solution with each new population.

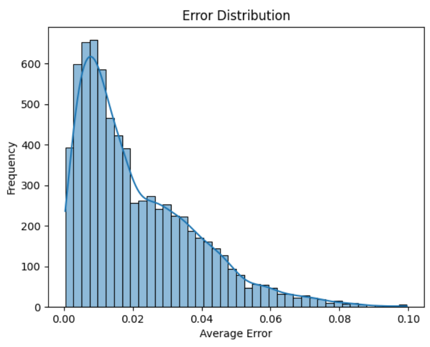
For the parameters to analyse I chose: n(30,32,35,37 or 40), population size (30, 70, or 100), mutation rate (0.01, 0.05, 0.1), crossover rate(0.8, 0.9, 0.95), tournament size(2, 4, 6), number of elites to carry to new population (1, 3, 5), crossover type(One Point or Two Point) and number of generations(30, 50, 100). I focused on analysing the influence of various Genetic Algorithm parameters on the runtime and solution quality for the Knapsack problem. I permutated all possible combinations because I wanted to have huge number of parameter configurations and to see easier what are the best parameters and is there any correlation between them.

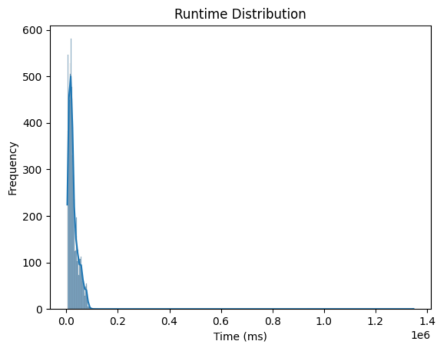
The algorithm works in a way that it iterates “number of generations” times through the algorithm and every time generates totally new population with which it works. Initially it creates a random genes population and repairs it if it’s not valid solution. Repairing it means to remove least valuable (cost/weight ratio) item one by one from the knapsack until the solution becomes valid. I did that only in the beginning because it is normally a bit too difficult computationally and with this, I am sure that my algorithm will find and keep at least one valid solution through the whole algorithm because sometimes my algorithm would not be able to find valid solutions at all depending on the example.

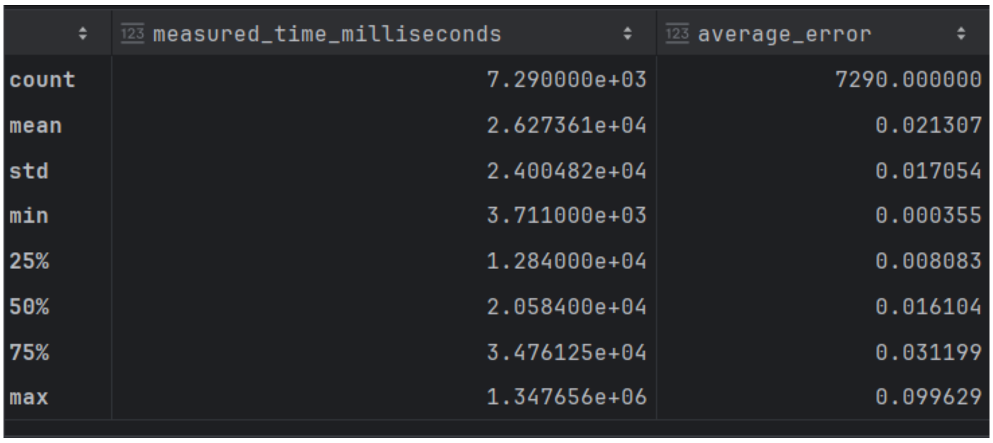
After initialising it goes into iteration part where it generates new chromosomes two by two. First it takes two chromosomes from the old population by tournament selection. Then, it does crossover and mutation with its respectable chances of being taken. And at the end, it takes best old chromosomes and puts them also in the new population so that it always has record of the best generated chromosomes and never losing it.

Results and interpretations

Runtime and error distribution







	measured_time_milliseconds	average_error
count	7.290000e+03	7290.000000
mean	2.627361e+04	0.021307
std	2.400482e+04	0.017054
min	3.711000e+03	0.000355
25%	1.284000e+04	0.008083
50%	2.058400e+04	0.016104
75%	3.476125e+04	0.031199
max	1.347656e+06	0.099629

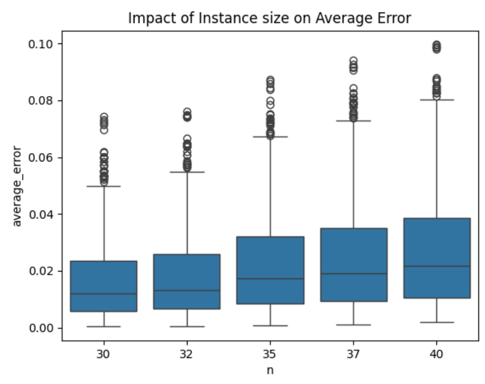
From the distributions we can see that distribution is somewhat good but for the runtime there exists an instance which has much bigger runtime that all others. We will get to it later when we analyse best and worst examples and its parameters.

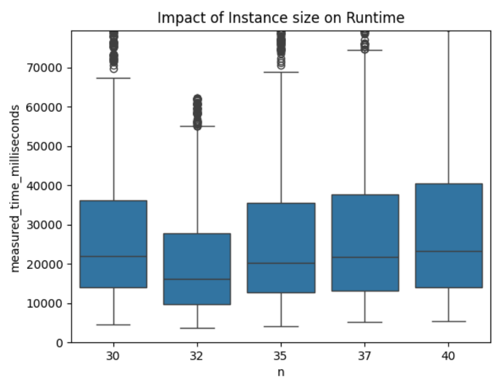
For runtime we can see that on average it takes 26 seconds to calculate, minimal value is 3 seconds while maximal value is 1347 seconds (22 minutes).

For average error, its mean is 0.0213 relative error with minimal value being 0.000355 and maximum being 0.099629. It corresponds to how much calculated value was from the real value in percentage.

Results from individual parameters

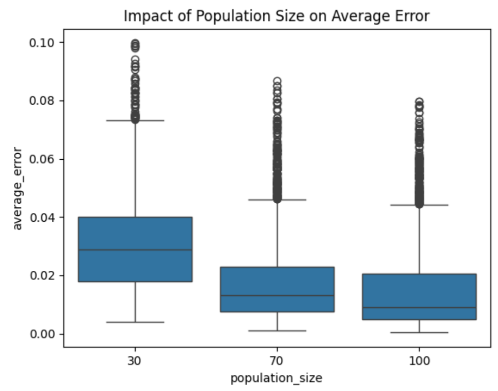
Instance size

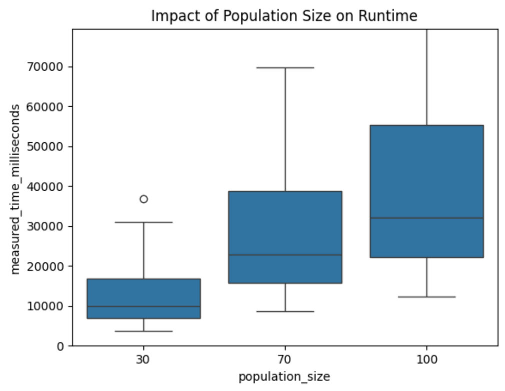




We can see that the average error is increased with increase of instance size. But interestingly, we cannot see any correlation between instance size change and runtime change.

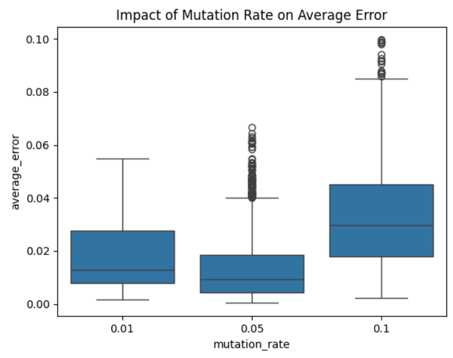
Population size

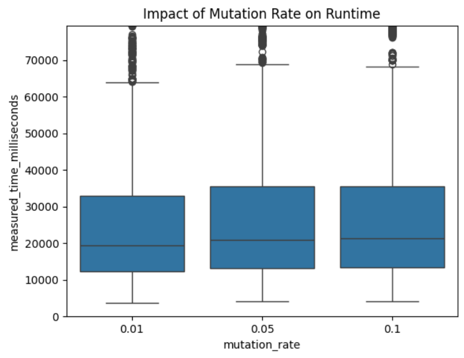




We can see that with increase in population size, error decreases while the runtime increases. For that reason we need to find a optimal value in between which will find values with at least error as possible while not being so computationally difficult to calculate.

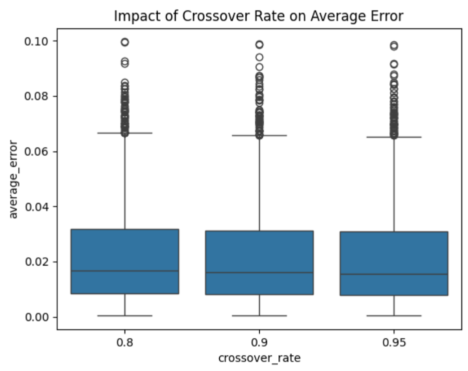
Mutation rate

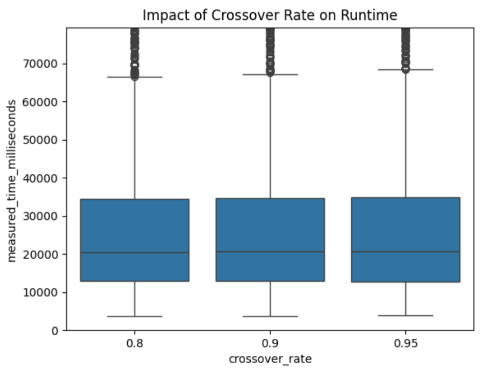




For the mutation rate we can see that the best values is 0.05 which is in between because it calculates with the least error on average while not affecting the runtime.

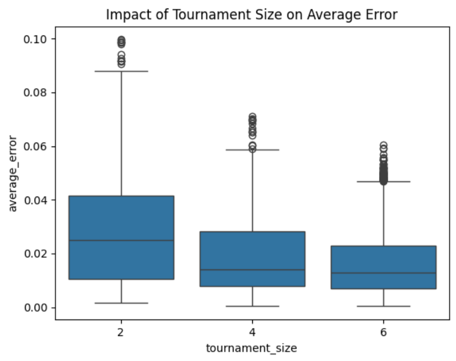
Crossover rate

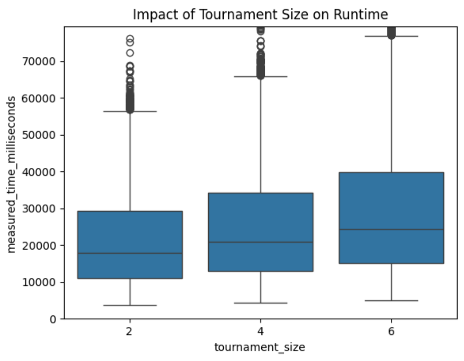




Crossover rate does not affect much either runtime nor average error.

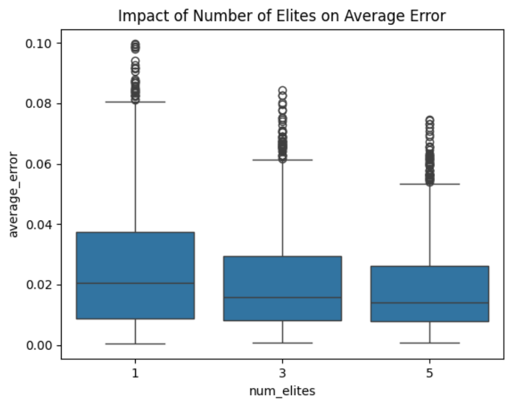
Tournament size

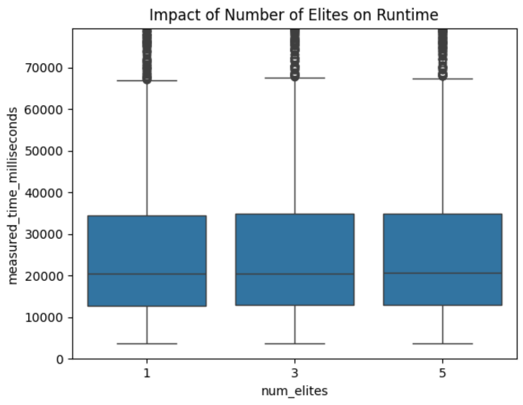




We can see that tournament size also affects error and runtime like population size but with less influence. For this value some optimal value would be 4 because it is a perfect balance with error and runtime even though 6 could also be used for a bit more precision.

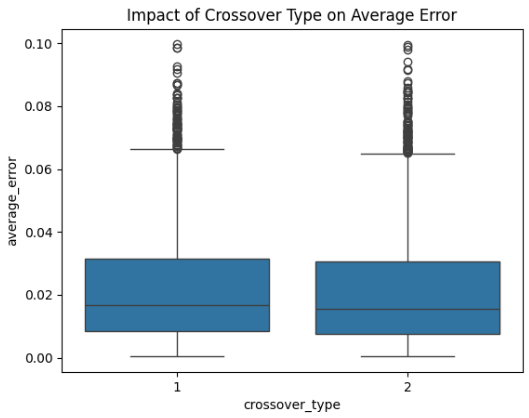
Number of elites

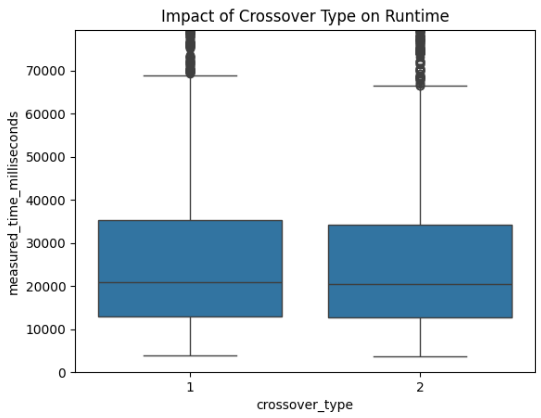




We can see that number of elites transferred from the old to the new population does not affect runtime at all but improves our algorithm for average error. That is why I would take value 5 as the most optimal one. If I had tested for a greater number of elites maybe I would see difference because it is not good to have too large number of elites because then our algorithm does not produce much new chromosomes but for now this number seems fine.

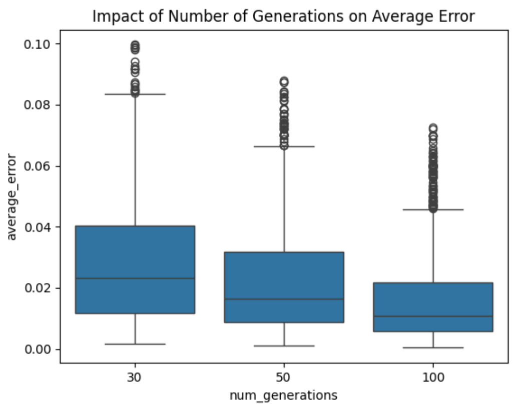
Crossover type

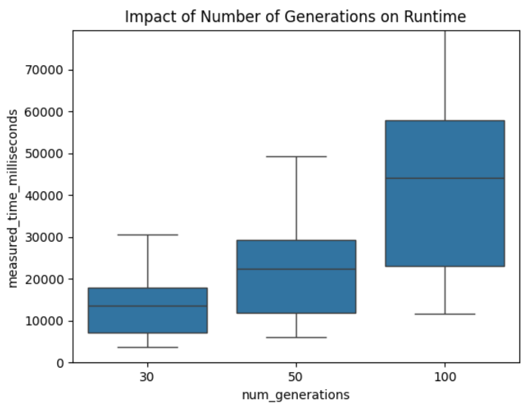




There does not seem to be any big impact on whether we choose one or two point crossover.

Number of generations





Similar to population size, increasing number of generations decreases average error but has much impact on the runtime of our algorithm so here we should think about some optimal value which get as least as possible error while maintaining low runtime.

Best and worst example

Minimal error parameters:

n	30
average_error	0.000355
population_size	100
mutation_rate	0.05
crossover_rate	0.95
tournament_size	4
num_elites	1
crossover_type	2
num_generations	100

Minimal runtime parameters:

n	32
measured_time_milliseconds	3711
population_size	30
mutation_rate	0.01
crossover_rate	0.8
tournament_size	2
num_elites	1

crossover_type	2
num_generations	30

Maximum error parameters:

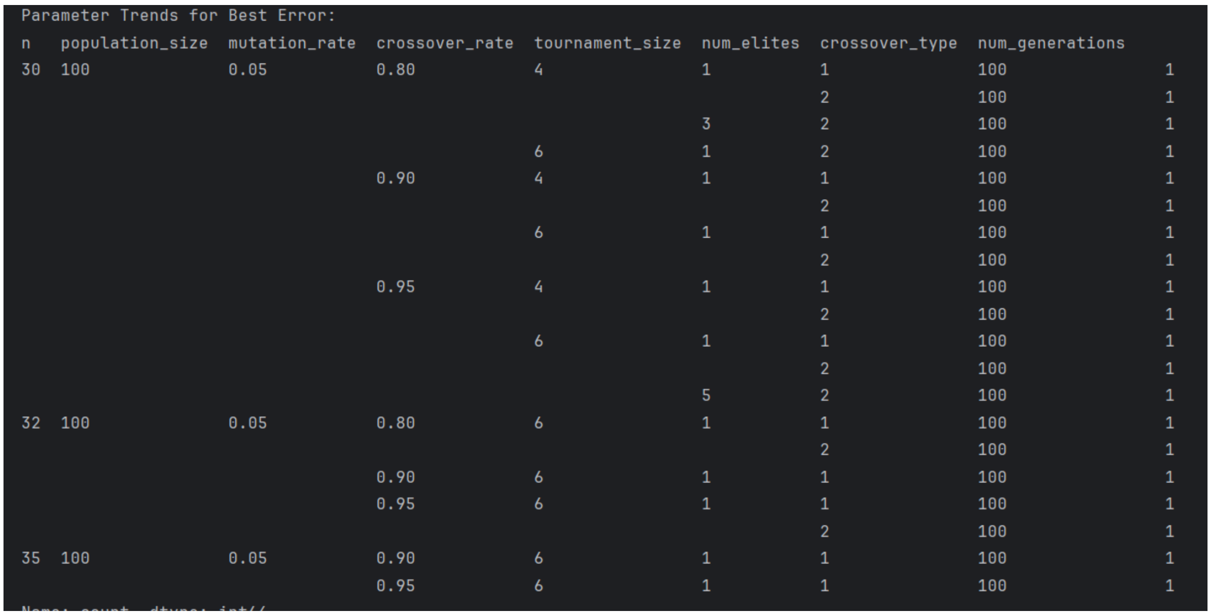
n	40
average_error	0.099629
population_size	30
mutation_rate	0.1
crossover_rate	0.8
tournament_size	2
num_elites	1
crossover_type	1
num_generations	30

Maximum runtime parameters:

n	40
measured_time_milliseconds	1347656 miliseconds
population_size	100
mutation_rate	0.05
crossover_rate	0.95
tournament_size	6
num_elites	3
crossover_type	1
num_generations	50

Optimal parameters

For deciding on what would be optimal parameter values and what would be worst ones to use, I took 20 of best/worst parameter permutations to see is there some values that are shown more than the others.



```
Parameter Trends for Best Error:
n   population_size  mutation_rate  crossover_rate  tournament_size  num_elites  crossover_type  num_generations
30  100              0.05           0.80            4                1           1               100              1
                                                                                 2               100              1
                                                                     3           2               100              1
                                                    6                1           2               100              1
                                    0.90            4                1           1               100              1
                                                                                 2               100              1
                                                    6                1           1               100              1
                                                                                 2               100              1
                                    0.95            4                1           1               100              1
                                                                                 2               100              1
                                                    6                1           1               100              1
                                                                                 2               100              1
                                                                     5           2               100              1
32  100              0.05           0.80            6                1           1               100              1
                                                                                 2               100              1
                                    0.90            6                1           1               100              1
                                    0.95            6                1           1               100              1
                                                                                 2               100              1
35  100              0.05           0.90            6                1           1               100              1
                                    0.95            6                1           1               100              1
```

Repeated values for best error:

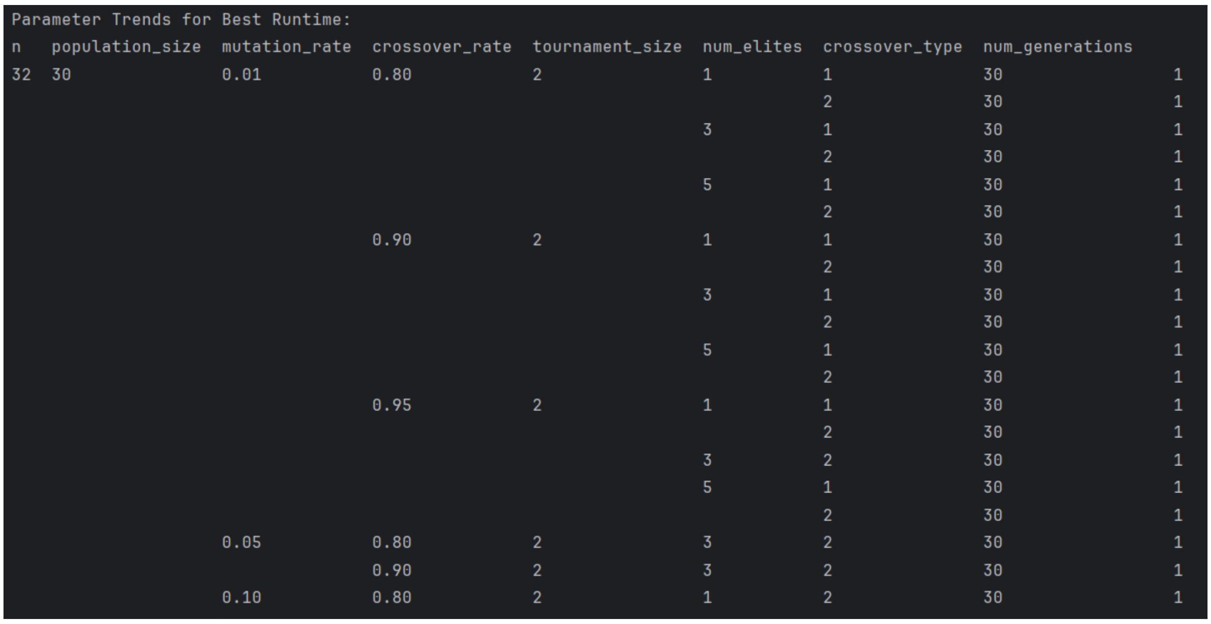
Population – 100

Mutation rate – 0.05

Tournament size – 4 and 6

Number of elites 1

Generations number - 100



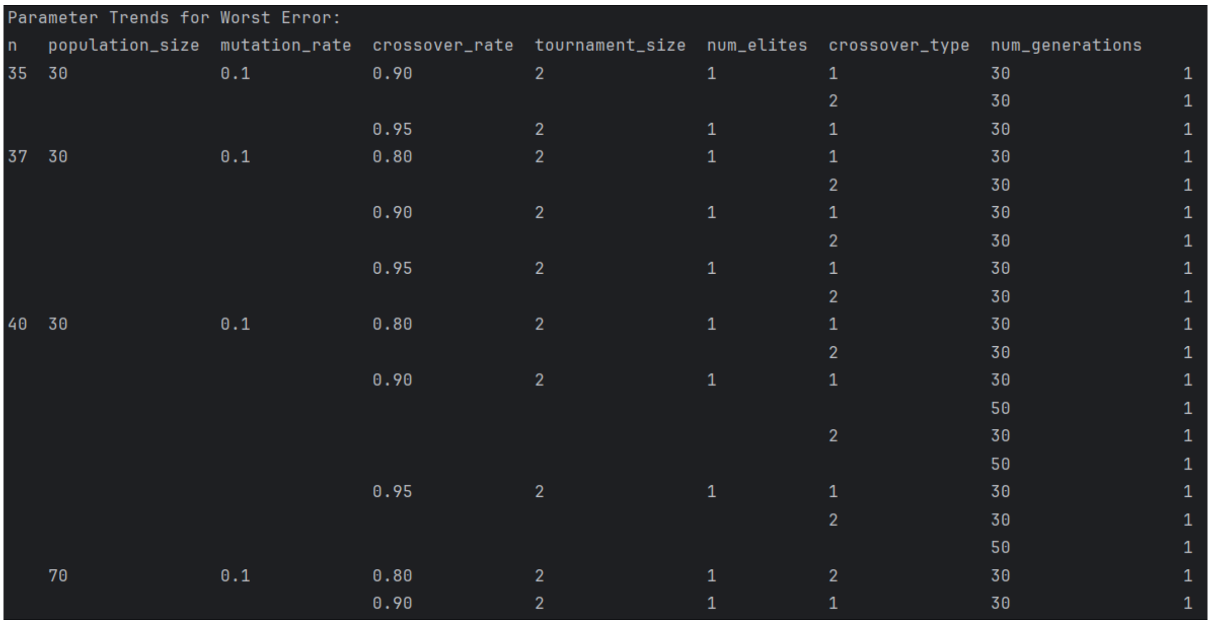
```
Parameter Trends for Best Runtime:
n   population_size  mutation_rate  crossover_rate  tournament_size  num_elites  crossover_type  num_generations
32  30               0.01           0.80            2                1           1               30               1
                                                                                 2               30               1
                                                                     3           1               30               1
                                                                                 2               30               1
                                                                     5           1               30               1
                                                                                 2               30               1
                                    0.90            2                1           1               30               1
                                                                                 2               30               1
                                                                     3           1               30               1
                                                                                 2               30               1
                                                                     5           1               30               1
                                                                                 2               30               1
                                    0.95            2                1           1               30               1
                                                                                 2               30               1
                                                                     3           2               30               1
                                                                     5           1               30               1
                                                                                 2               30               1
                     0.05           0.80            2                3           2               30               1
                                    0.90            2                3           2               30               1
                     0.10           0.80            2                1           2               30               1
```

Repeated values for best runtime:

Population size – 30

Tournament size – 2

Number of generations – 30



```
Parameter Trends for Worst Error:
n   population_size  mutation_rate  crossover_rate  tournament_size  num_elites  crossover_type  num_generations
35  30               0.1            0.90            2                1           1               30                  1
                                                                                 2               30                  1
                                    0.95            2                1           1               30                  1
37  30               0.1            0.80            2                1           1               30                  1
                                                                                 2               30                  1
                                    0.90            2                1           1               30                  1
                                                                                 2               30                  1
                                    0.95            2                1           1               30                  1
                                                                                 2               30                  1
40  30               0.1            0.80            2                1           1               30                  1
                                                                                 2               30                  1
                                    0.90            2                1           1               30                  1
                                                                                                 50                  1
                                                                                 2               30                  1
                                                                                                 50                  1
                                    0.95            2                1           1               30                  1
                                                                                 2               30                  1
                                                                                                 50                  1
    70               0.1            0.80            2                1           2               30                  1
                                    0.90            2                1           1               30                  1
```

Repeated values for worst error:

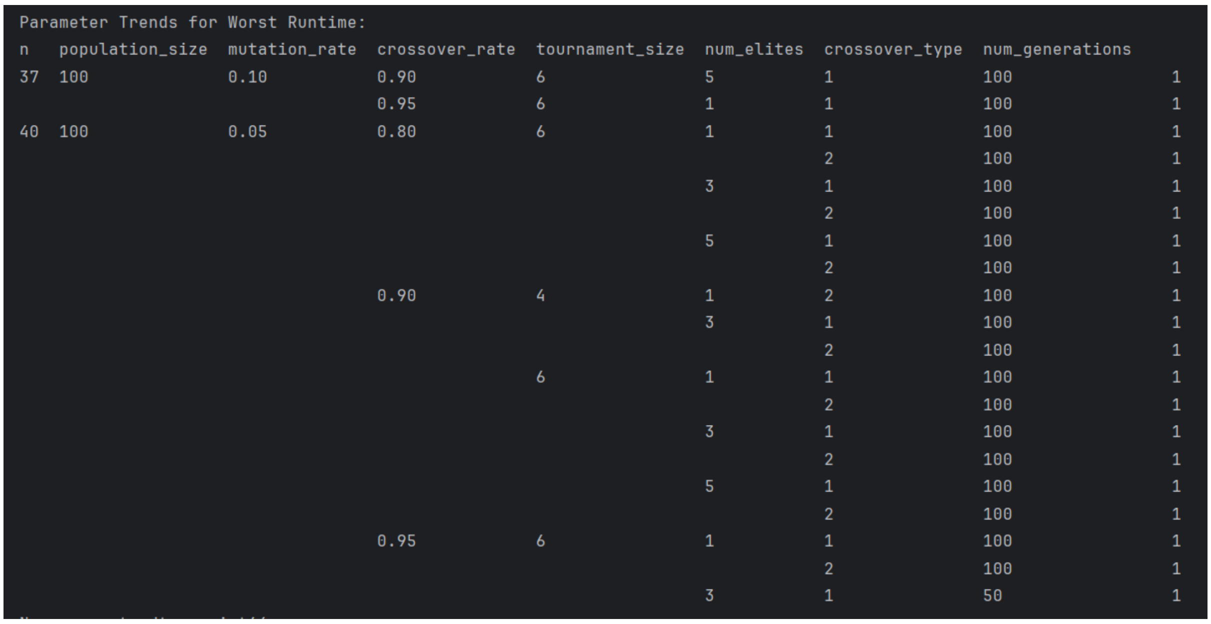
Population size – 30

Mutation rate – 0.1

Tournament size – 2

Number of elites – 1

Number of generations - 30



```
Parameter Trends for Worst Runtime:
n   population_size  mutation_rate  crossover_rate  tournament_size  num_elites  crossover_type  num_generations
37  100              0.10           0.90            6                5           1               100              1
                                    0.95            6                1           1               100              1
40  100              0.05           0.80            6                1           1               100              1
                                                                                 2               100              1
                                                                     3           1               100              1
                                                                                 2               100              1
                                                                     5           1               100              1
                                                                                 2               100              1
                                    0.90            4                1           2               100              1
                                                                     3           1               100              1
                                                                                 2               100              1
                                                    6                1           1               100              1
                                                                                 2               100              1
                                                                     3           1               100              1
                                                                                 2               100              1
                                                                     5           1               100              1
                                                                                 2               100              1
                                    0.95            6                1           1               100              1
                                                                                 2               100              1
                                                                     3           1               50               1
```

Repeated values for worst runtime:

Population size – 100

Mutation rate – 0.05

Tournament size – 6

Number of generations – 100

Conclusion

There does not seem to be optimal value for population size and number of generations, it only depends on do you want your result to be accurate or to be fast.

Most optimal value out of these three for mutation rate would be 0.05 because of how accurate it is on average which we saw on boxplots while still not hurting the runtime much.

Crossover rate and crossover type do not seem to affect much on the performance or quality of results.

Tournament size is like the population size in terms of cost of error for runtime so it is not crystal-clear which value would be the best but the most optimal seem to be 4.

For number of elites that are transferred to the new population, the best number seems to be 5 but that also depends on the problem. Bigger number of elites does not necessarily mean smaller error, just in this experiment we did not test bigger elites number so as not to affect quality of new population that is generated.