Randomized Optimization

Cody Phrampus
cphrampus3@gatech.edu

1 Problem 1 - KColor

This problem is a minimization problem for coloring a graph such that no two connected nodes have the same color. Fitness is a measure of the number of errors and the problem was generated as allowing each node up to 4 connections.

1.1 Tuning

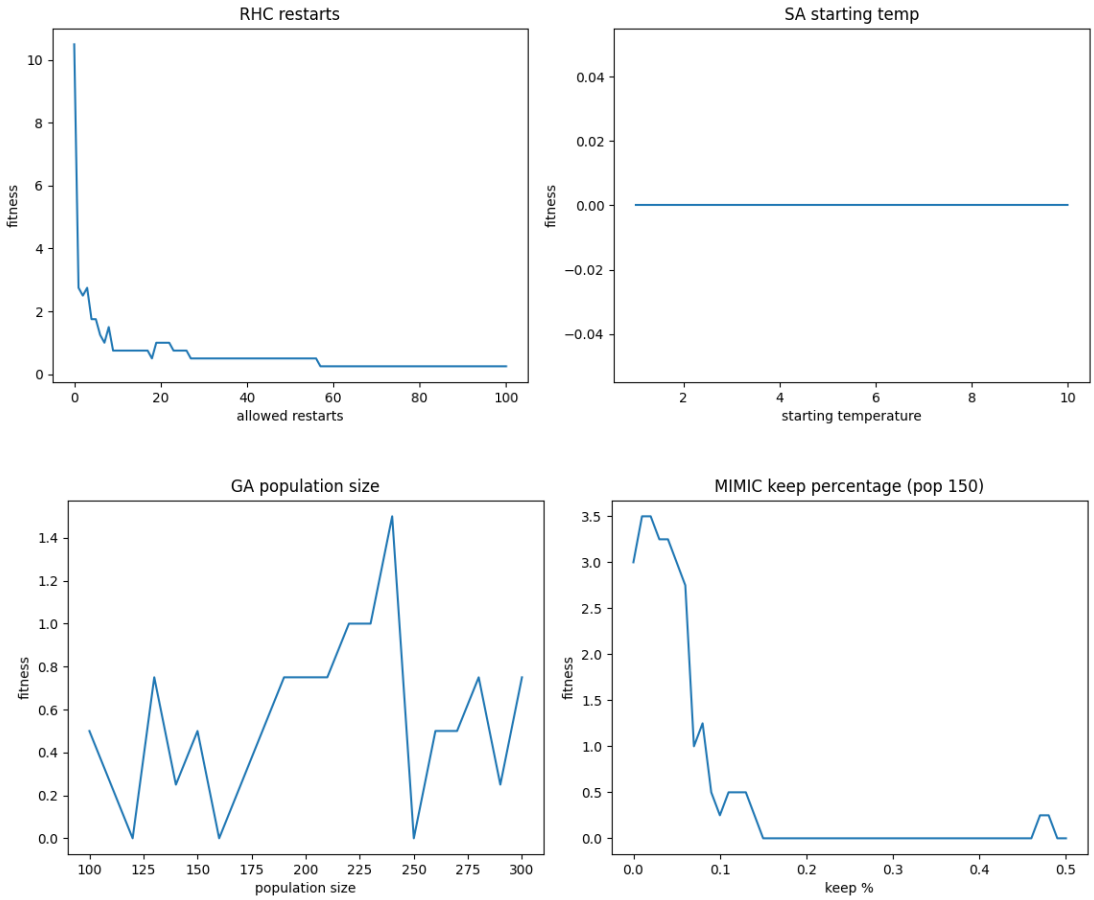

Figure 1, 2, 3, 4—tuning for RHC (top left), SA (top right), GA (bottom left), and MIMIC (bottom right)

In tuning the algorithms, it was important to recall that the minimal value is desired. The parameters tuned are picked as they are the main driving forces in the algorithm's ability to learn. The number of allowed restarts is the essence of *random* hill climbing, providing some ability to avoid local optima, but the algorithm is still not very targeted in its approach, acting as

more of a baseline than anything else. Simulated annealing is mainly defined by the starting temperature and how long it takes to cool to give the algorithm time to wander the landscape before settling down in the best place in its area. Genetic algorithms and MIMIC are defined by the population of items they will work with and how frequently a mutation occurs when a new item is generated, in the case of GA, or the percentage of the population to continue with, in the case of MIMIC. There are other knobs that can be turned for these algorithms, particularly GA and MIMIC because they are more sophisticated, but the number looked at was kept relatively small in order to allow time for experiments to run. The final values used to tune the algorithms are as follows: 60 allowed restarts for random hill climbing, a starting temperature of 1 with a decay rate of 0.99 for simulated annealing, a population size of 120 with a mutation probability of 0.12 for the genetic algorithm, and a population size of 150 and keeping 15% for MIMIC.

1.2 Comparison

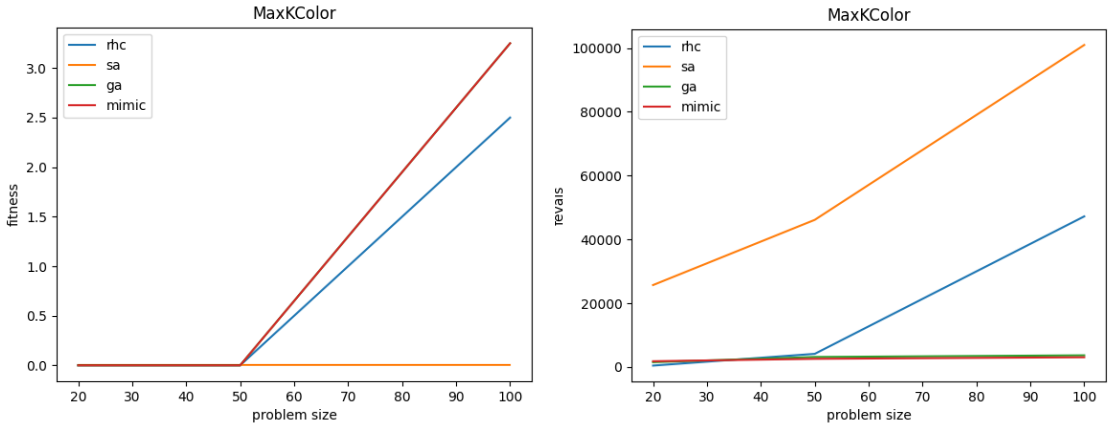

Figure 5, 6—fitness (left) and fevals (right) for algorithms

The algorithms were tuned for size 50 and all were able to get a fitness of 0 errors, after which the number of errors, averaged over 5 random seeds, began to grow as the problem exceeded the complexity tuned for by doubling in size. MIMIC was the best performing algorithm for this problem by measure of fitness function evaluations, with almost 20% fewer than the next best. This is not very surprising, the KColor problem has a solid underlying structure, so the structure based algorithms, GA and MIMIC, were able to perform best and find this structure with the fewest evaluations. MIMIC via finding the relationships between nodes and GA due to the fact that combining good solutions would also combine some representation of the structure of the graph. This problem is significantly closer in terms of evaluations than the original MIMIC paper, as the problem was not limited to graphs with a unique solution and only up to 4 connections were allowed (De Bonet). Because of this, the problem was significantly simpler which made it quicker to run, but also presented the interesting fact that simulated annealing managed to find the solution without the need for tuning and managed to keep 0 errors even at a problem size of 100, due to simply being able to brute force the graph coloring. However, for

the tuned size and the scoring parameter, fevals in this case, it was still not considered the "best." Given more time a more complicated problem could have been crafted to prevent this, however, MIMIC would still be the leader in this case as the number of connections grew until a unique solution would exist for certain graph configurations, as detailed in the original MIMIC paper (De Bonet).

2 Problem 2 - Six Peaks

This problem is similar to the Four Peaks problem, wherein there are global maxima on the outskirts of the space with local maxima within the space, with two additional global maxima (De Bonet). This is a maximization problem with a strong underlying structure where the algorithms were tuned for a problem size of 30.

2.1 Tuning

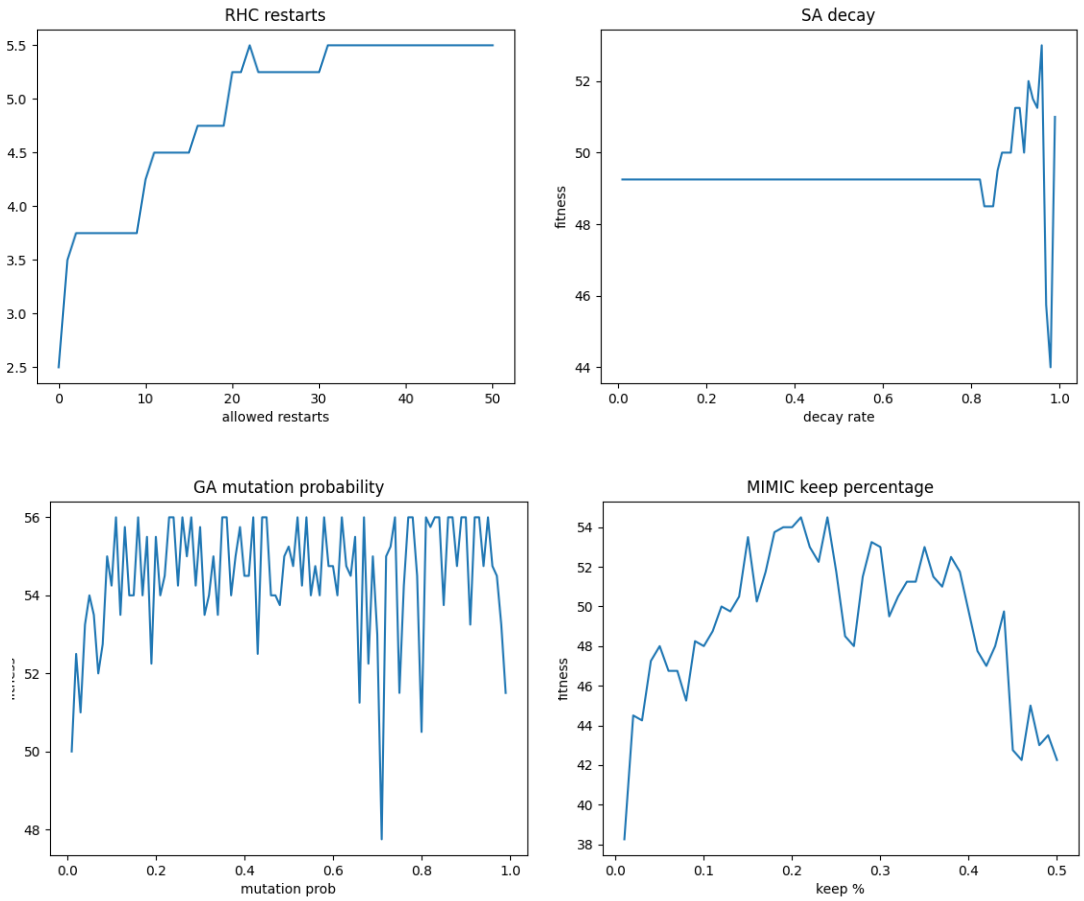

Figure 7, 8, 9, 10—tuning for RHC (top left), SA (top right), GA (bottom left), and MIMIC (bottom right)

The tuning methodology was the same as for problem 1, leading to the final tuned values for the algorithms, from the plots above with previous plots over different parameters, to be: RHC has

35 allowed restarts, GA has a population size of 210 with a mutation probability of 0.23, MIMIC has a population size of 300 and keeps 20%, and SA has an initial temperature of 11 with a decay of 0.96.

2.2 Comparison

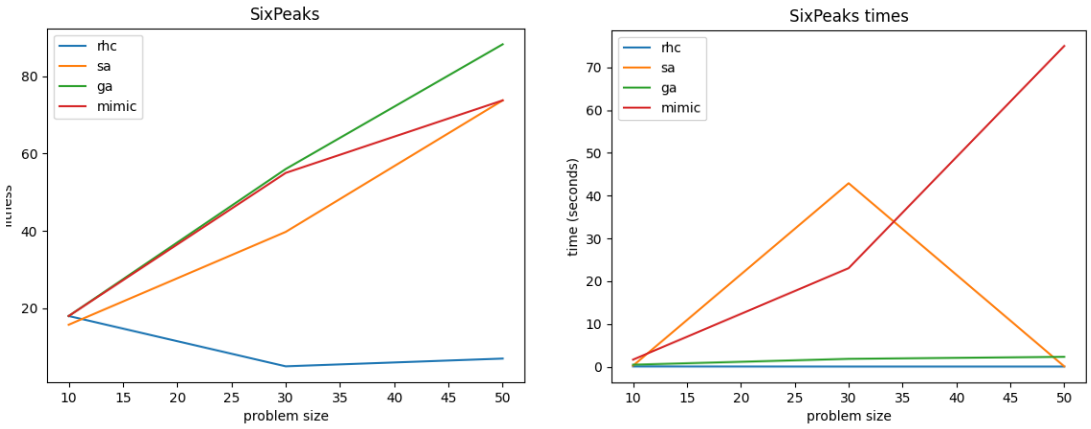

Figure 11, 12—fitness (left) and fevals (right) for algorithms

The algorithms performed about as expected given the problem structure. RHC performed quite poorly, getting stuck in one of the multiple local optima in the problem space. SA performed significantly better due to its ability to walk around the problem space more freely as it cooled before settling on an answer, avoiding the local optima better than pure random hill climbing. MIMIC performed well due to there being a strong underlying structure to the problem and where the global optima resided and being able to recognize that the beginning and ending chunks of the bit string, each of size T+1, should contain the same value and differ from one another. GA performed very well due to its ability to combine good answers to get to a better answer overall, thus avoiding being trapped in the local optima and finding the outlier global optima as subsequent generations moved in that direction. GA performed best on the tuned problem size of 30, both in fitness by 2 points, 56 to MIMIC's 54, and the fact that it was able to get this level of fitness in almost 1/20th the time of its competitor due to having a significantly quicker turnaround time between generations/iterations.

3 Problem 3 - KQueens

This problem involves placing k queens on a k by k chess board, one in each rank, so that the minimum number are being attacked. It is a minimization problem, since we are seeking to minimize the number of attacks, and tuning took place for a 36 queen version of the problem.

3.1 Tuning

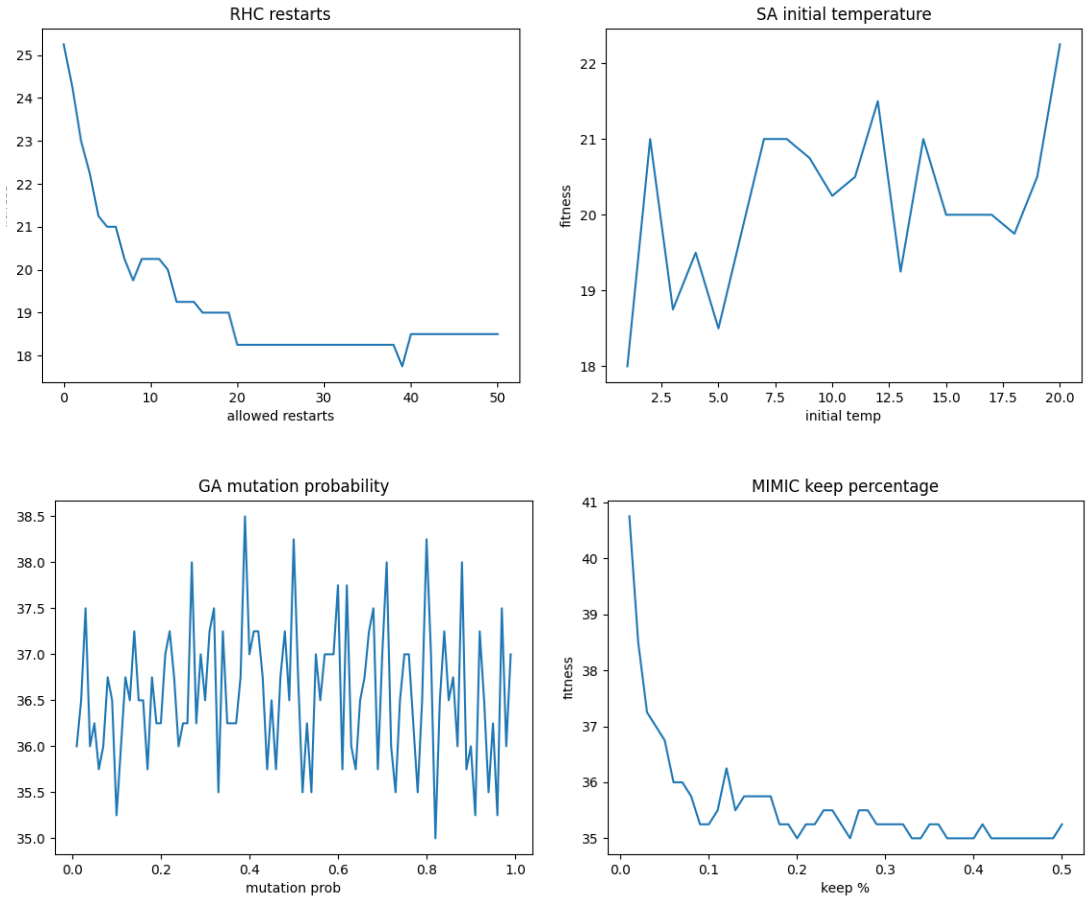

Figure 13, 14, 15, 16—tuning for RHC (top left), SA (top right), GA (bottom left), and MIMIC (bottom right)

After tuning from the plots above, the final values of the algorithms are: RHC has 39 available restarts, GA has a population size of 170 and a mutation probability of 0.82, MIMIC has a population size of 275 and keeps 20%, and SA has a starting temperature of 1 with a 0.94 decay.

3.2 Comparison

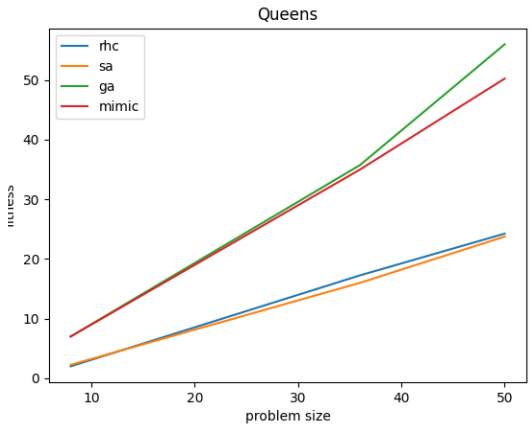

Figure 17—fitness for algorithms

It is clear from the graph that SA takes this handily in terms of minimizing fitness out of the main three being compared. This seems like a counterintuitive result since, as humans, this problem is heavily structured in terms of having k queens that can move horizontally and diagonally, vertically does not matter since for this problem it is a queen in each column precisely due to this attack option, in k columns. There is a structure to it and the pieces have defined relationships between them. However, the fact that the algorithms not concerned with structure greatly outperform the ones that are gives a hint as to the problem with the problem here. Despite the problem being one with underlying structure, this only matters if the *representation of the problem* has structure and rewards that structure being used. In this case, it is clear that it is the brute force approach that is being better rewarded. Given more time, an interesting next step of analysis would be across problem representation or simply a deep dive into the mlrose representation and how the structure could be represented to showcase GA and MIMIC.

4 ANN with Randomized Optimization weight finding

For the Neural Network comparison, the networks were trained and run over the UCI wine dataset to classify which of three regions in Italy the wine was produced (wine). This dataset contains 13 features for the algorithm to use to determine the region.

4.1 Tuning

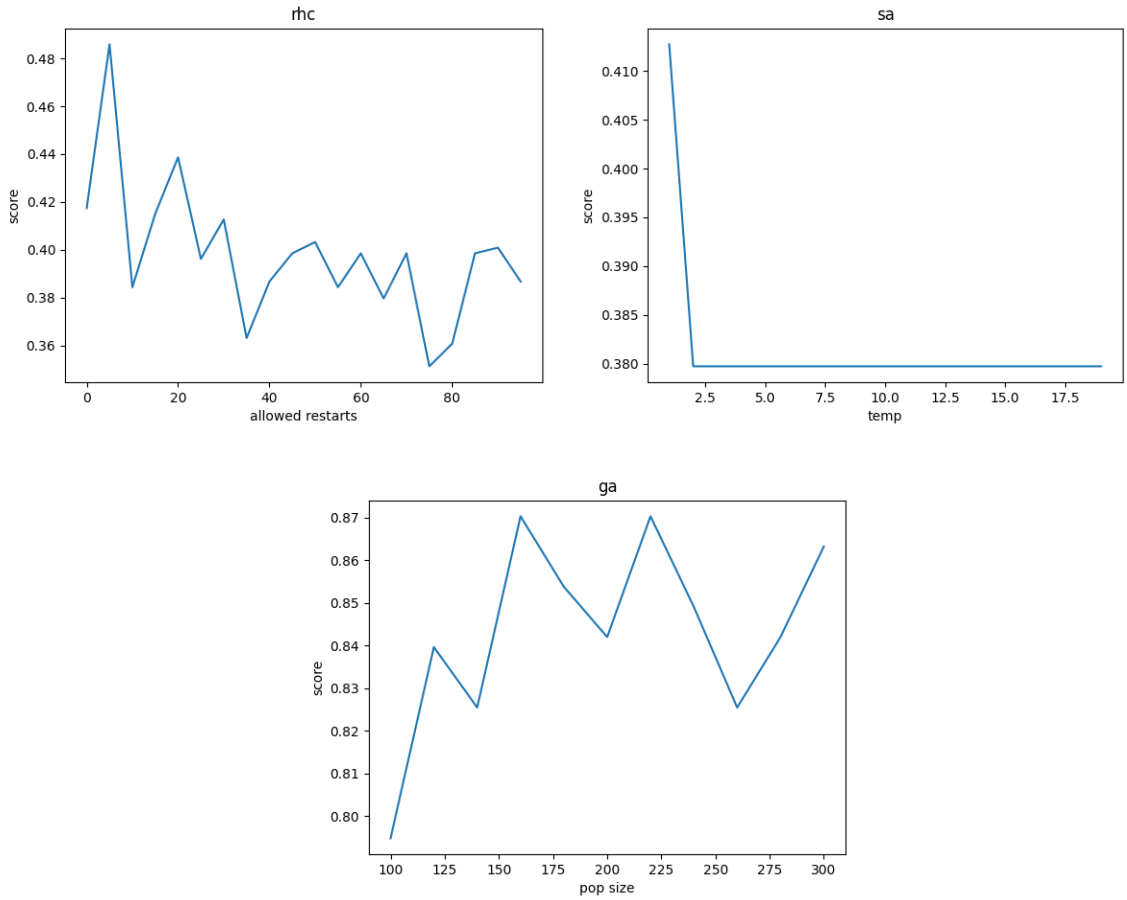

Figure 18, 19, 20, 21—tuning for RHC (top left), SA (top right), GA (bottom)

Similar to the tuning done in the optimization problem steps, the final values found for the algorithms are: RHC was allowed 5 restarts, SA was given a starting temperature of 1 and a decay rate of 0.99, and GA was given a population size of 160. The gradient descent approach was ported into mlrose from sklearn with no architectural changes from P1: a single hidden layer of 290 nodes and a learning rate of 0.001. It was discovered upon final review that the architecture was changed from P1, since the use of 290 nodes in the hidden layer was for the second problem, wine quality, and this number was not adapted to the correct value of 85 when the problem used in this comparison was changed and there was not sufficient time to retrain and analyze. This, however, should not affect the comparison, as the "incorrect" architecture was used consistently and performed comparably in the gradient descent case to the correct one. An interesting feature to note is the steep dropoff in the SA graph as the temperature increased from 1 to 2, indicating that having more ability to roam is detrimental to the algorithm. All algorithms were given 110 max iterations following testing, this was purely for

plotting purposes as even given many more iterations, they changed so little as to have functionally converged.

4.2 Comparison

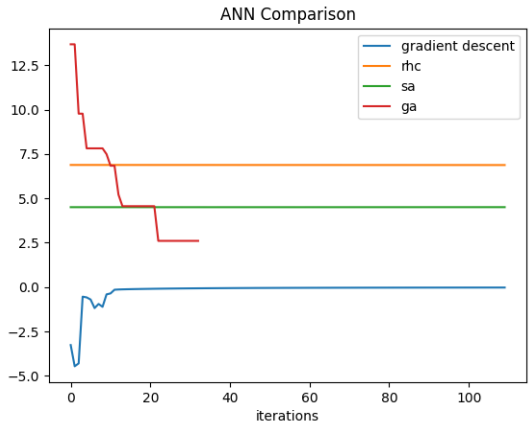

Figure 22—fitness curve

Kernel	Fit time (seconds)
Gradient Descent	*1.06*
RHC	*2.85*
SA	*0.73*
GA	43.1

Table 1—ANN run times

Unsurprisingly, gradient descent performed the best in accuracy, there's a reason it is used after all. It was not the quickest in terms of speed, getting beaten by SA, but still vastly outperformed it in terms of correct classification. As noted above, the iterations were limited for the sake of plotting, as all algorithms had functionally stopped improving by 50 iterations, with only GA truly converging and utilizing the early stopping mechanism.

The randomized approaches varied in their efficacy with random hill climbing and simulated annealing performing fairly similarly and gaining no real ground as iterations increased. This is indicative that the landscape of the problem did not lend itself well to even SA's ability to avoid local optima and it ended up getting trapped just like RHC. This seems to indicate very specific points in the space where the global optima reside that SA was unable to find, even with its random walks about the space. The Genetic Algorithm, on the other hand, comparably well to gradient descent. This shows that there is a way to combine good answers within the space to reach even better ones, moving towards these global optima, as was the case in the Six Peaks

problem. The long stretches of trying to converge and minimal change to the loss values seems to indicate a predominantly flat landscape such that the single step iterative approaches have trouble gaining much ground. GA is able to gain ground quite well due to the crossover ability of having a population spread across the landscape, all of which can help the next generation arrive in a better place, rather than simply trying to improve just where you are, as hill climbing would do as would gradient descent due to its use of derivatives to determine movement. It is worth noting that despite the number of iterations needed for gradient descent to converge, it still performs exceptionally well, it seems to be in another area where the landscape is fairly uniform despite being close to the global optimum. Given more time to experiment, interesting next paths would include some kind of dimensionality reduction or feature selection to help remove extraneous details which may be clouding the landscape and causing this very uniform "texture."

5 References

1. De Bonet, J., C. Isbell, and P. Viola (1997). MIMIC: Finding Optima by Estimating Probability Densities. In Advances in Neural Information Processing Systems (NIPS) 9, pp. 424–430.
2. UCI machine Learning Repository: Wine data set. (n.d.). Retrieved February 18, 2021, from http://archive.ics.uci.edu/ml/datasets/Wine