Matthew Schroder

4350

Dr. Philips

The goal of this project is to find the maximum value of the sum of gaussians function using both a greedy algorithm and a simulated annealing algorithm. The SoG class accepts two parameters, the number of dimensions (D) and the number of gaussian hills (N) desired. Using these two parameters the class will create a space with dimension D that the algorithms will attempt to maximize. Also provided in the SoG class are methods for evaluating the function and finding the derivative at a point. The general strategy for both algorithms is to start at random position x, evaluate the SoG function at x, then move to the next state.

The greedy algorithm uses the SoG class's Deriv() function to determine the next state. More specifically, it adds the result of 0.01*SoG.Deriv(x) to the current x value and evaluates the SoG function at this new point. This results in the algorithm climbing the hill nearest to the starting position. An obvious drawback to this method is that the peak of the nearest hill is not always the optimal solution. Furthermore, as the number of hills increase the odds of greedy finding the optimal solution decreases.

The simulated annealing algorithm uses a temperature variable in conjunction with the metropolis criterion to explore the SoG function and eventually settle on what it thinks is the optimal solution. The temperature is controlled by what we call an annealing schedule. The annealing schedule chosen for this program is

$$100000 * e^{-0.0004*i} \quad 0 \leq i < 10000$$

where i is based on the current iteration of the for-loop that controls the algorithm's iterations. This function is based off the exponential decay function.

$$P = P_0 \mathrm{e}^{-\mathrm{rt}}$$

This means that the initial temperature of 100,000 will exponentially decrease as i increases. After much tweaking and testing, I found that starting with a high P_0 and a low *r* allowed the algorithm to spend most of the time exploring the state space. The specific values used were based purely on observation, but I attempted to have a bigger drop around 50,000 iterations.

Once set, the metropolis criterion uses the temperature to produce a probability between 0 and 1 using the formula below.

$$e^{\frac{(y_2 - y_1)}{temp + 1e-8}}$$

This probability decreases as the temperature decreases and dictates whether the algorithm will make a move that decreases the current value of y and the probability. This results in an algorithm that starts highly explorative and ends much like a greedy algorithm, that is, one that constantly climbs. The simulated annealing algorithm would occasionally produce values that were out of the range (0,10). This did not seem to affect the performance by much, but it did waste valuable iterations to evaluate nonexistent y values. In response to this I wrote a loop that checked each value of the new x position and that subtracted a random value if it was outside of (0,10) as oppose to adding it otherwise.

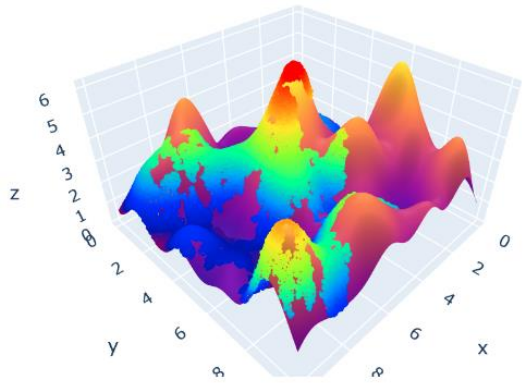

Figure 1 Simulated Annealing

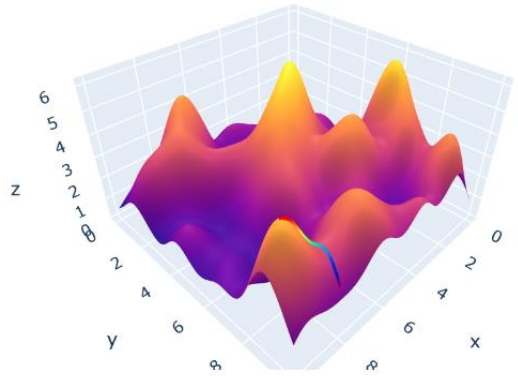

Figure 2 Greedy

Once finalized I ran both programs with seeds 0-100 and D =1, 2, 3, 5 and N = 5, 10, 50, 100. The max values from 1,600 runs of each algorithm were compared to see which would perform better and the results were unsurprising.

Greedy: 335 SA: 1262 Tie: 4

The simulated annealing algorithm is less constrained by its random starting state than the greedy algorithm. But the greedy algorithm is more focused than the simulated annealing algorithm. If both programs start in a position next to a hill, the greedy algorithm will immediately begin climbing that hill while the simulated annealing algorithm will explore other prospective peaks. Simulated annealing may have a position that is close to a suboptimal hill when the temperature starts dropping exponentially.

This would result in simulated annealing climbing that hill and greedy potentially finding a higher y value. Overall, the simulated annealing algorithm performed with higher consistency than the greedy algorithm when the number of gaussian hills increased.