Project 2 Report

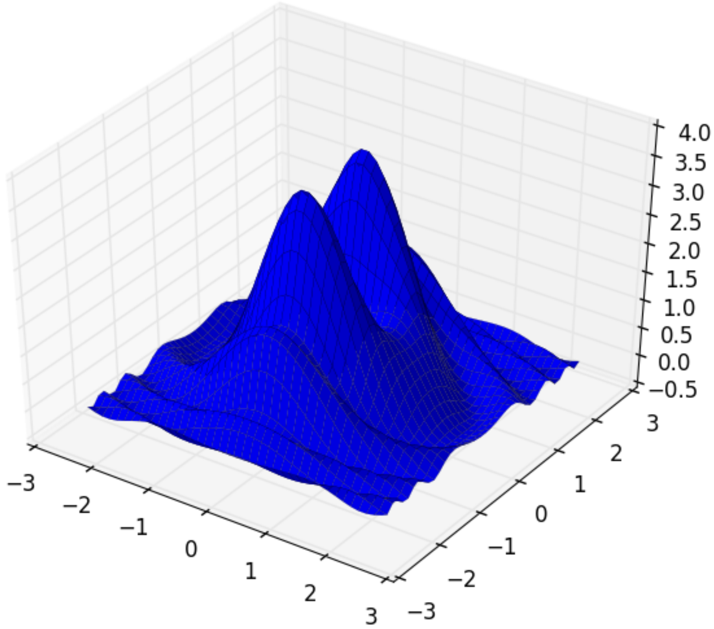
Benjamin Nace

March 16, 2016

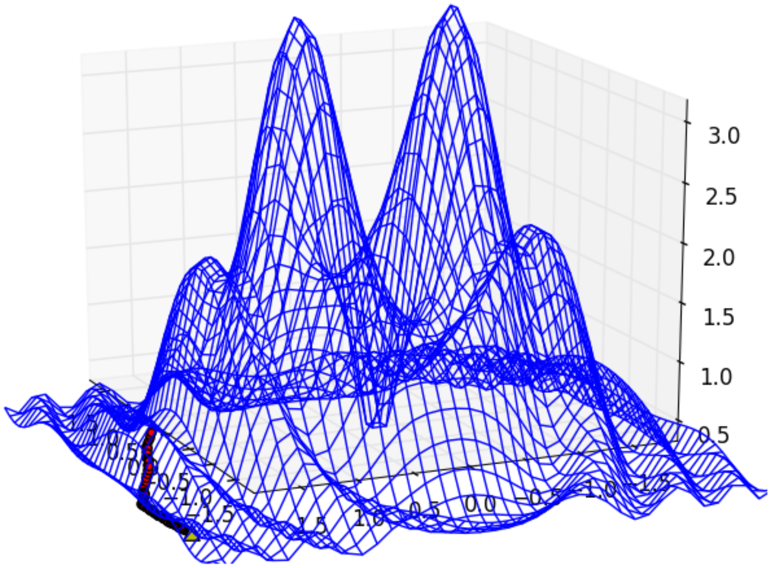
Introduction This report documents the results from using hillclimbing, hillclimbing with random restarts, and simulated annealing to minimize the following function:

$$z = \frac{\sin(x^2 + 3y^2)}{0.1 + r^2} + (x^2 + 5y^2) \times \frac{\exp(1 - r^2)}{2}, r = \sqrt{x^2 + y^2} \qquad (1)$$

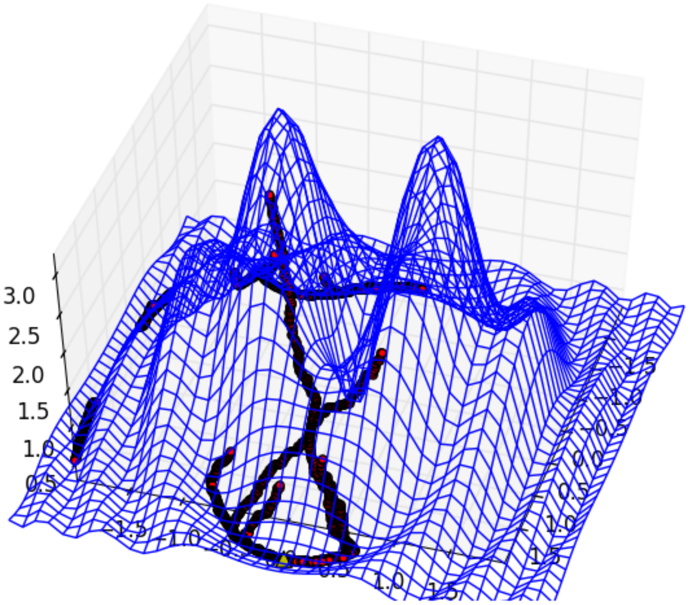
The equation was bounded from (-2.5, 2.5) to (2.5, 2.5), which produces the following graph:



Hillclimbing The first method for minimization was hillclimbing. Running the algorithm on the function produced a minimum point of (-1.429, 2.259), rounded to three decimal places. Running it again produced a minimum position of (-2.176, -0.003). The hillclimbing algorithm runs very fast, but does not produce very consistent results. Where it ends up depends on where it randomly starts, since it only goes downhill from the starting position. Here is a graph of the first sample run's path:

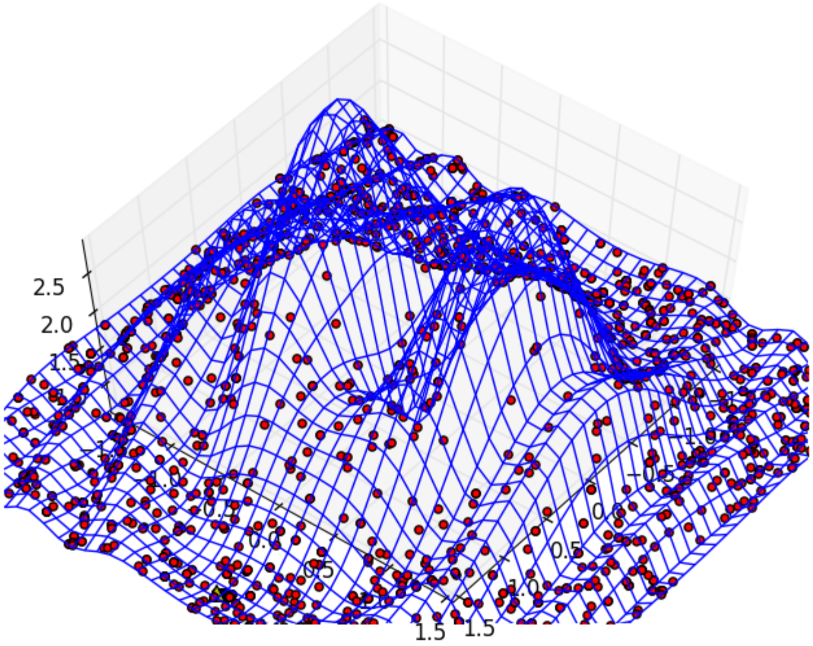


Hillclimbing with Random Restarts The second method was hillclimbing with random restarts. It used the same method as the hillclimbing algorithm, except it repeated it a set number of times, in this case, 100. This algorithm ran a bit slower than hillclimbing. There was a noticeable pause before the results were generated, whereas hillclimbing produced near-instant results. A first run produced a minimum location of (2.173, -0.001). A second run produced (2.173, 0.000). So, this method appears to be more consistent than just hillclimbing. Here's a sample graph of the path for this algorithm:



Please note: This graph is from a run with only 10 restarts, for sanity's sake.

Simulated Annealing The final method was simulated annealing. Running it once produced a minimum location of (-2.173, 0.000). Running it again also produced a minimum point of (-2.173, 0.000). This algorithm was a little bit slower than hillclimbing, but faster than hillclimbing with random restarts. A sample graph for this algorithm's path looks like:



Please note: This graph is from a run with a lower starting temperature and step size, for sanity's sake.

Final Analysis In terms of finding the minimum, hillclimbing with random restarts and simulated annealing did the best. They were both about equal. The graph is symmetrical, so (2.173, 0) and (-2.173, 0) are essentially the same. In terms of speed, hillclimbing was the fastest, simulated annealing was second fastest, and hillclimbing with random restarts was the slowest. This is not surprising given the implementations of the three algorithms. Hillclimbing goes always downhill from a random starting point, which makes it the fastest, but the least accurate. Hillclimbing with random restarts does hillclimbing, starting from a bunch of different points. The repeated nature of it makes it the slowest, but with enough restarts, it can be relatively reliable in finding the minimum. Simulated annealing can move uphill at the start, and continues to move with a decreasing uphill probability until it eventually turns into hillclimbing. It is the most accurate, since it has the most freedom to explore before settling on a solution. And it only has to do one pass, which makes it faster than hillclimbing with random restarts, but the added complication makes it take longer than hillclimbing. Given these facts, I think simulated annealing is the best algorithm for this minimizing job, since it is the fastest algorithm that still produces accurate results.