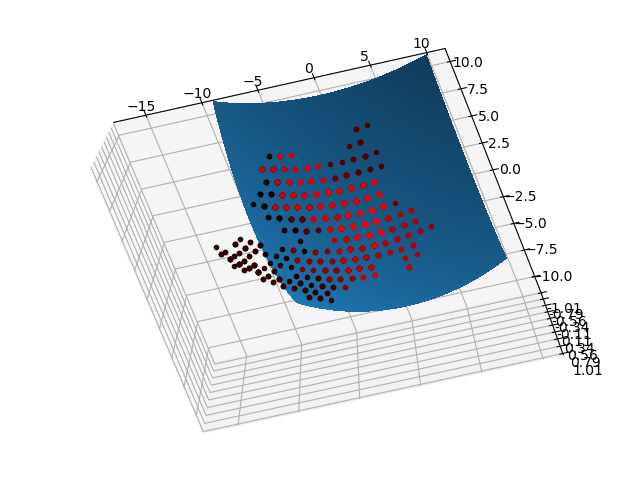
Robert Genega

AI project 2

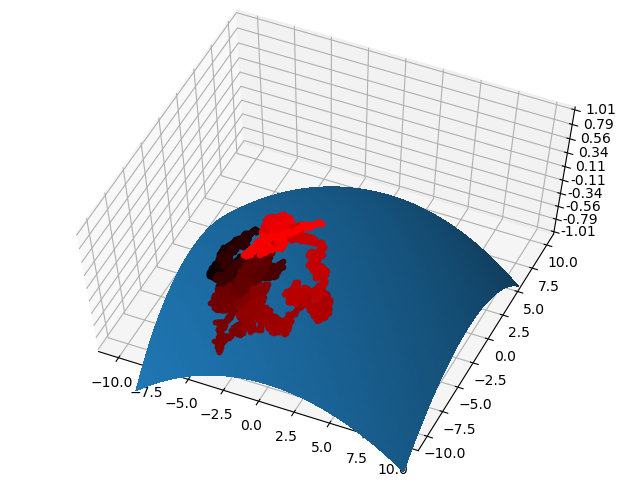
For the first part of the project I coded a gradient ascent search with step size =1. As per the project description, the input to my program is python local\_search.py <input\_file> <xmin> <ymin> <xmax> <ymax>. I used a step count of 1000 and passed all the test cases.

After dealing with the nightmare of installing matplotlib to Microsoft visual studio, I decided to code a Simulated Annealing Search. I used the same function from the input of the first part of the project (<input\_file>) to do the search on. When graphing that function, I bounded the graph by the same inputs from the first part (<xmin>, <ymin>, <xmax>, <ymax>). I bounded the start point of the simulated annealing algorithm by these parameters, but I allowed the search to travel out of this range (see first graph) as to stay true to the nature of the algorithm. I added two extra parameters, a variable for the number of steps taken (numsteps) and a variable for the step size (step). I adjusted these parameters until my graph clearly demonstrated an annealing pattern (see second graph).

Aside from installing matplotlib, I had trouble with angling the output graphs, as the program was glitchy and unresponsive.



Step size = 1, number of steps = 1000. That looks right, the redder dots are concentrated near the maximum, lets try it with step size = 0.1 and number of steps = 10,000



Ah that looks beautiful! It looks like we found the maximum!