Task 1: Gradient ascent and Hill-climbing with a simple function

1.1

Although the two algorithms share a common goal, the discrepancies in how they function lead to some interesting differences in performance. In many cases, the hillclimber has the ability to reach a greater height than the gradient ascent algorithm due to its stochasticity, and resultant ability to escape suboptimal local minima - these currently seem to be the 'Achilles heel' of our implementation of gradient ascent, so to speak.

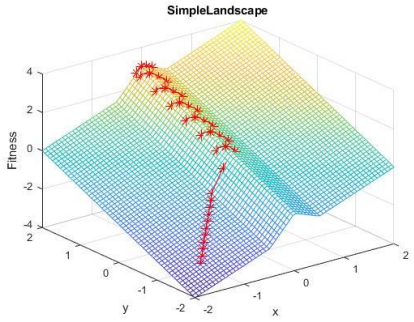


Figure 1.1 – Gradient ascent getting stuck in suboptimal local-minima

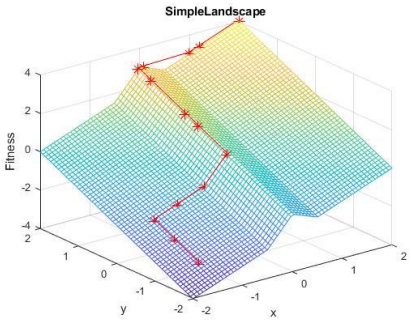
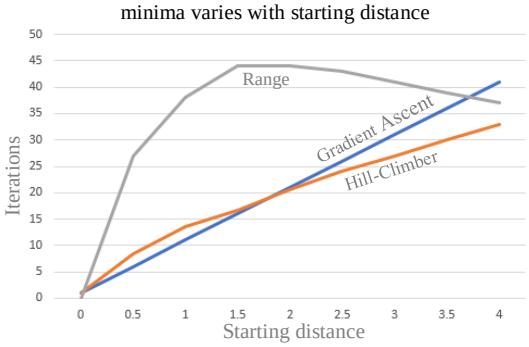


Figure 1.2 – Hill-climber escaping suboptimal local-minima

In figures 1.1 and 1.2, you can see a side-by-side comparison of the two algorithms when initiated from the same starting point. It's easy to see how the success of our gradient descent implementation is highly dependant on its starting position. It must also be noted however that the behaviour exhibited in figure 1.2 is not guaranteed, the same stochastic nature that allows for this path to be possible, of course means that there is a chance it won't happen. I anticipate factors such as the amount of iterations, the maximum mutation distance, and by all means the structure of the landscape, to be key factors in the probability of escaping a suboptimal local-minima - but this is a point for further investigation. In the cases where the gradient ascent approach is able to achieve global-minima, it will do so far more consistently thanks to its deterministic nature.

In terms of the relative times taken by each algorithm to achieve global-minima (in the cases where this is possible for the gradient ascent), this is somewhat dependent on the starting-distance from it. In the cases where a random starting point is in close proximity of the global minima, the gradient ascent fairly consistently outshines the hillclimber. Contrary to this, the hillclimber is at an advantage in the cases where the starting-point is further afield, this can be credited to its ability to cover a much greater distance with each iteration. In figure 2 you can see the results from my experimentation, in which each hill-climber data point is an averaged from 100 using our default parameters. I must however stress the level of variation existent amongst these, whilst the gradient ascent performed consistently as expected, the amount of iterations required by the hillclimber would vary hugely. The specifics of the amount of variation at each distance can be seen plotted by the line labelled "Range" in figure 2. Further experimentation would be required to explain why it tapers off as the distance begins to surpass 1.5, however I anticipate it the maximum iterations allowed, in conjunction with the nature of the landscape, to be a factor in this.

Figure 2 – Graph showing how time taken to reach global-minima varies with starting distance

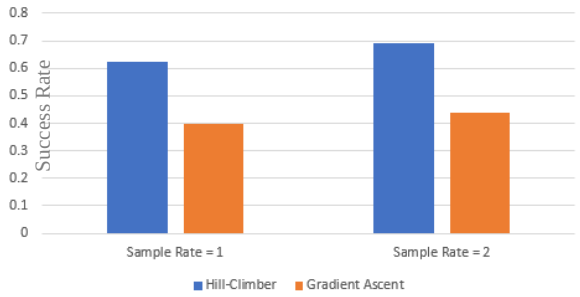


1.2

I proceeded to test each algorithm systematically over a grid of starting points covering the fitness landscape, this grid originally involved a starting point at every integer coordinate. I worried however that this may not sufficiently sample the fitness landscape as no starting points fell on the two faces of the central ridge.

Because of this, I made the decision to introduce a variable called *gridDensity* which allowed me to scale up the amount of points in the grid. When set to 2, a starting point is placed at every 0.5 coordinate intersection (instead of every integer when *gridDensity = 1)*, and as a result, every face in the Simple Landscape is sampled.

Figure 3 – Clustered bar chart showing success rate of each algorithm at both grid densitys.



In figure 3 you can see the success rates of each algorithm at each tested *gridDensity*. It must be noted that the results for the hill-climber were averaged over 10 iterations due to its decreased consistency. Whilst the success rate increased for both algorithms with a higher grid density, it did slightly more-so for the hill-climber, as any new start points situated below the ridge that leads to suboptimal-minima, had a chance of avoiding or escaping it.

Since 62.5% of the total landscape surface ($\forall\ x < 0.5$) leads to the suboptimal local-minima by deterministic gradient ascent, the hill-climber outperformed it by a significant margin in terms of success rate. When we consider the amount of iterations required to reach it however, gradient ascent saw a 12.8% (to 3 s.f) improvement on the hill-climber from 25.56 down to 22.3.

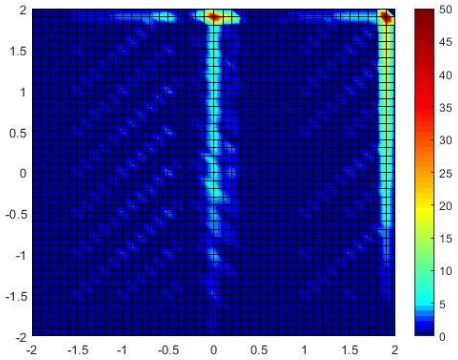


Figure 4.1 – Pcolor visualisation of Gradient Ascents

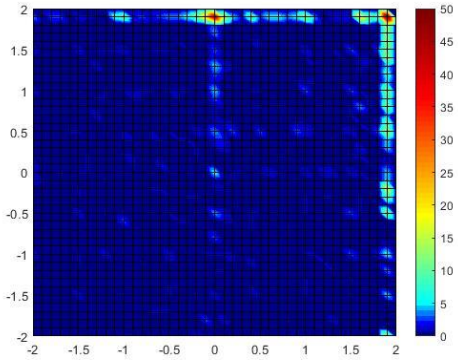
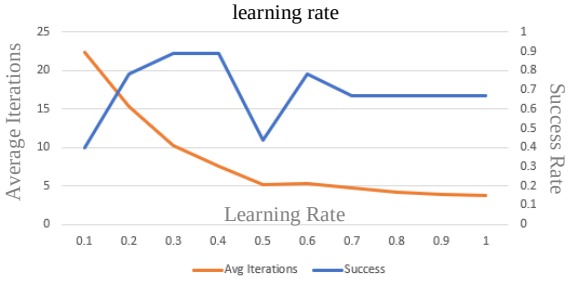


Figure 4.2 – Pcolor visualisation of Hill-Climbers

In figures 4.1 and 4.2, you can clearly visualise the nature of each algorithm. The distribution of points in figure 4.2 is completely stochastic, as you would expect, on all faces of the fitness-landscape, with concentrations only visible on edges and extruding vertexes (minima). Figure 4.1 shows the very systematic approach of the gradient ascent, including the clear boundary where $x = 0.5$. Below this boundary it has a 0% success rate, and above it has a 100% success rate. I decided to test the hill-climber on these two regions individually, and found where the gradient ascent has a 0% chance of success, the hill-climber still averaged 52%. Yet above this boundary where the gradient ascent has a perfect success rate, the hill-climber averaged out at 94% over the 10 iterations of the full grid of starting points. This suggests that the hill-climber is the more well-rounded of the two, yet it does outline where there may be slight room for improvement.

1.3

Figure 5 – How success rate and iterations vary with learning rate



Initially, increasing the learning rate seems to only bring great benefit to the gradient ascent – the success rate skyrockets, whilst the average iterations required plummets. A greater learning rate allows it to take steps large enough to escape the sub optimal local-minima. I don't anticipate this level of success to carry over to other, more complex, landscapes however. Due to the nature of how *x,y* boundaries are enforced, combined with the fact that the global-minima resides on both an *x* and *y* upper-boundary, this negates the usually-calamitous

shortcoming of a learning rate that is too large – consistent overshooting. In the case of an inevitable overshoot on either axis, here the coordinate is simply rounded to the boundary value, shared by the global-minima.

The sharp drop in success that came with a learning rate of 0.5 also caught my attention, however upon investigation it became evident that this was purely due to the specific positions of the start points coincided with the fact that 0.5 is the exact width of a face of the ridge. Combined with a gradient of one, any instance that began below, will land directly on the top of the ridge, getting channelled towards suboptimal minima. To test this I offset every start point by 0.1 on the x-axis, and this then achieved a 100% success rate. This illustrates how dependant the effectiveness of the learning rate is on the specific fitness-landscape in question.

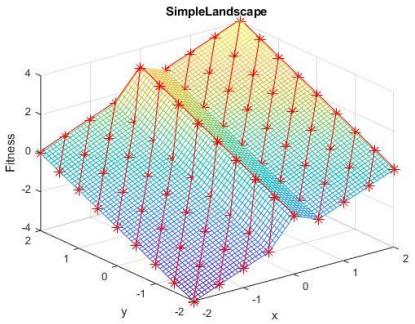
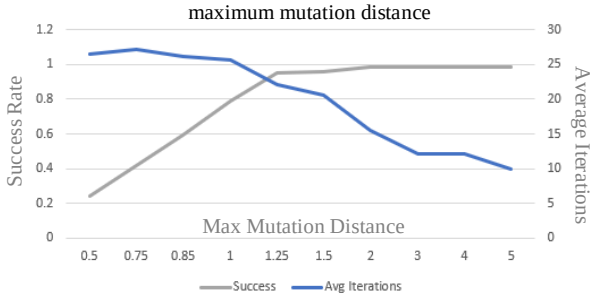


Figure 6 – The drop in success when learning rate = 0.5

Figure 7 – How success rate and iterations vary with maximum mutation distance



When experimenting with the maximum mutation distance of the hillclimbers, I once again got results that could be said to be 'too good to be true' - that I anticipate will not be replicable on any landscape that doesn't share the very specific characteristics of this one. Namely that the global-minima resides on the *x* and *y* limit, towards which, any out-of-bounds values are rounded.

Task 2: Gradient ascent and Hill-climbing with a complex function

2.1

As we move on to the complex landscape, the first thing that becomes apparent after launching the gradient ascent function from various random starting points - is that a significant proportion of the time it will be initiated at a point with a gradient of 0, such a point didn't exist with the simple function. This fatal flaw means it simply has no way of achieving any progress, regardless of the learning rate or maximum iteration count, as its movement is a direct function of the current gradient. Although an increased learning rate would be a great asset when dealing with terrain of very low gradients, this would become increasingly detrimental as the gradient steepens with regards to localising on a minima.

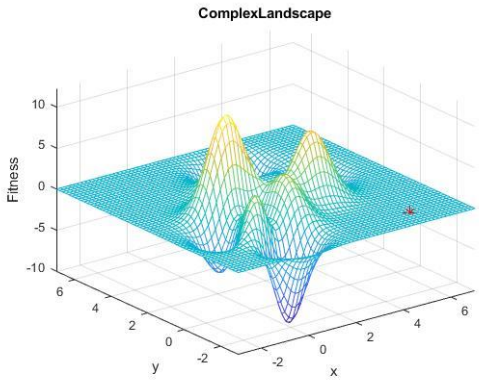


Figure 8.1 – Gradient ascent if initiated where the gradient = 0

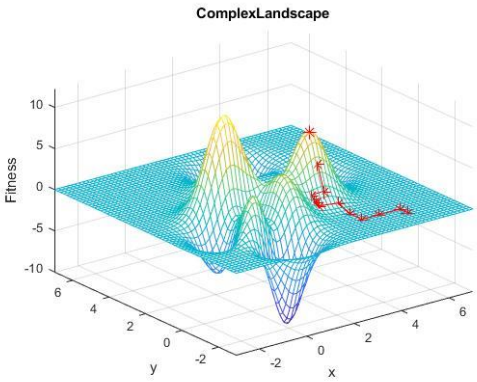


Figure 8.2 – Hill-climber initiated from same coordinates

The hill-climber on the other hand allows neutral moves, and although its movement over a flat landscape will be entirely random, it has a high probability to soon find itself within its maximum mutation distance of an upwards gradient, and thus increased fitness. Thereafter there will be some logic or directionality to its movements.

As with the simple function, the gradient ascent is highly dependant on its starting position, but to a far greater extent than before. And even in the unlikely event that it is initialised on the direct path to global-minima, we see overshooting (as mentioned in section 1.3) coming in to play, in which it fails to localise on the minima, instead oscillating around it due to the alternate gradients on each side. The terminating conditions that sufficed on the simple landscape no longer do so.

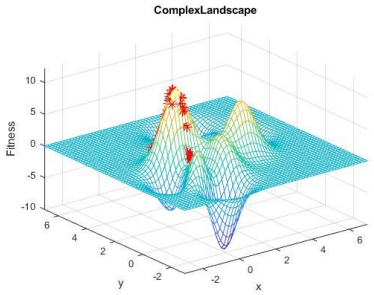


Figure 9 – Gradient ascent overshooting global minima

The hill-climber isn't as limited by its starting position due to its ability to traverse the flat regions, but with the default parameters it does seem to be just as susceptible to getting caught in a suboptimal minima. When it reaches a maxima, as opposed to oscillating around it as per the gradient ascent, it settles on a point near to it. With continuous data, limited iterations and anything more than a miniscule maximum mutation distance, the chances are minute that it will land perfectly on it. On this landscape I anticipate there to be a very real trade-off between the greatest possible starting-distance from a minima (whilst still localising on it in a given amount of iterations) and how close it is able to achieve to the actual fitness limit, as we vary the maximum mutation distance. I will investigate this in section 2.2.

With regards to the learning rate of the gradient ascent, I expect decreasing this to lead to a lesser trade-off than varying the gradient ascent's maximum mutation distance. The key disadvantage that comes with a reduced learning rate is the increased time it takes to close in on a minima. On this landscape however, in the majority of cases where this disadvantage would have come in to play and have an impact, the algorithm will often not progress at all regardless, as these are the cases where it is initialised far from a minima, where the gradient is likely 0, therefore making at large part of the downfall redundant.

2.2

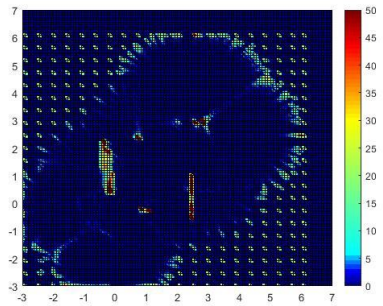


Figure 10.1 – pcolor visualisation of gradient ascent

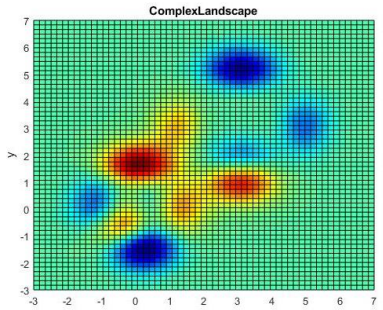


Figure 10.2 – Complex landscape topography

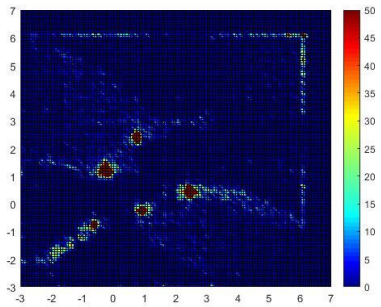


Figure 10.3 – pcolor visualisation of hill-climbers

In figures 10.1 and 10.3 you can see pcolor visualisations of each algorithm when systematically initiated from a grid of starting points spanning the entire landscape with all parameters as default, except a grid density of 2. Figure 10.2 is a useful point of reference to visualise what topography lies at any given coordinate, helping us to interpret the results seen in the two neighbouring pcolor plots. In figure 10.1 you can clearly see all the points where the algorithm has failed to make any headway, and all 50 iterations have landed on the same spot, leading to the high concentrations we see at each one. With the increased grid density, the basins, or range of influence, of each maxima and

minima are also clearly visible, however without figure 10.2 we would struggle to deduce the directionality of each one.

In terms of the minima, we can see a high concentration of hill-climbers on all 5 of them. A single instance of a perfect algorithm would of course only localise on the global-minima as opposed to those that are suboptimal-locals, but this effect can be emulated by instead initialising a large population of them spanning the problem-space (as we have done), and simply settling on that which achieved the greatest fitness. When we turn our attention to figure 10.1, we can see that the gradient ascent has high concentrations around 4 of the 5 minima, with an additional point of interest at approximately (3,3) which is not present in figure 10.3. Upon comparison with figure 10.2, we can see that this is due to the surrounding troughs in the landscape. It would be tempting to then call this a suboptimal local-minima, however this would be erroneous because in addition to, and much more critically than, carrying a fitness of 0, there is a neutral gradient in the negative *x* direction leading to a real local-minima. This is why this particular concentration doesn't exist in figure 10.3, as hill-climbers don't require a gradient for movement. The other point I would like to draw attention to in figure 10.1 is the nature of the identified minima - rather than focused points as we see in fig 10.3, they are much more linear. As we have previously mentioned, this is a result of overshooting, in which the gradient ascent oscillates around the minima, as opposed to settling on it. I hope to be able to reduce this, and I believe the solution lies in a variation with the terminating conditions, and/or the learning rate.

Figure 11 – How the heights achieved by gradient ascent vary with learning rate

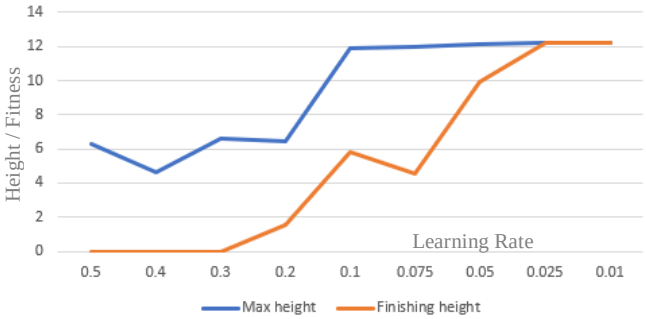
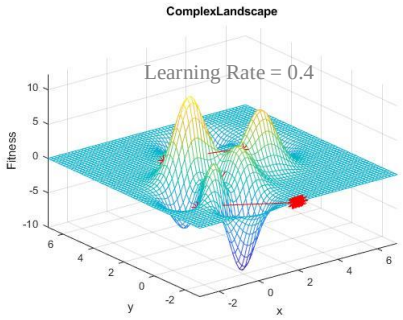


Figure 12 – Overshooting resulting in the discrepancy between max and finishing height

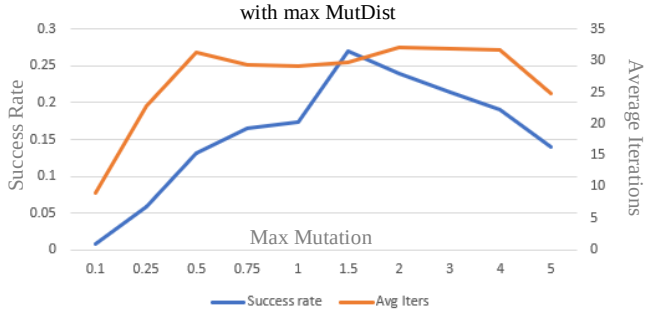


I first began my experimentation with the gradient ascents learning rate and how it would affect one individual instance that had a starting point within the global-minima's basin of attraction. The results of which can be seen in figure 11. As expected, instances run with a higher learning rate lacked the precision needed to effectively localise on the global minima. The discrepancy between the maximum height achieved, and the final height after *i* iterations also became apparent. In figure 12 you can clearly see the cause of this – overshooting can be far more detrimental than simply resulting in an oscillation about the minima, in the cases where the learning rate is great enough for the overshoot to land on a gradient beyond the directly opposing one, a situation like this is able to occur.

In order to effectively investigate the effects of varying a hill-climber's maximum mutation distance, I had to once again operate on an entire grid of them spanning the landscape, as due to their stochastic nature, the results I would achieve from operating on a single one would be far too varied to be at all credible. To be able to quantify these effects, I modified my code to classify any achieved fitness equal to, or above 11.59 as a 'good enough' solution, so for the purposes of this investigation we will consider values beyond that boundary as a global-minima, as this is in fact 95% of the true global-minima.

See figure 13 for the results of this experimentation. The immediately striking thing you may notice is the correlation between success rate and the maximum mutation difference, this is quite the opposite of what we found to be the case on the simple landscape (see Figure 7) in which the two appeared inversely proportional. For reasons I explained during section 1.3, I did not anticipate the behaviour we observed there to transfer on to other landscapes, and these findings have confirmed that.

Figure 13 – How success rate and associated average iterations vary with max MutDist



In terms of attempting to explain the reasons for this correlation, I would like to partition the graph in to 3 sections. The lower end of MutDist - where both success rate and average iterations are low, the middle section in which both variables are high, and the upper end of MutDist in which both are once again low. With a low mutation distance, the 'basin of attraction' of the global minima is significantly smaller, with only 50 iterations, only those that are initiated very close to the global-minima stand a reasonable chance of ever making it there, and so the success rate is low as a result. For those that are in range however, a smaller mutation distance gives more directionality, the behaviour may seem less random than one with a large mutation distance, as it has a higher chance of mutating by the very small and specific amounts needed when approaching the global-minima. This results in far fewer iterations being needed by those that were initiated within the possible range. As we increase the mutation distance to the mid-range, each instance of the algorithm is able to travel further, widening this this basin of attraction. Since it is now possible for those initiated further away to reach the global-minima, the success rate rises, and therefore so does the average iterations – as some have travelled from further afield. As we move towards the upper-end, a mutation distance that is too large effectively increases the search space, the probability of mutating by the small distances required to home in on the minima become ever smaller the closer it gets. This results in a reduced success rate, however those that are successful by chance, do so in few iterations due to the large step sizes being taken with each iteration.

The arcing shape of the graph also confirms the hypothesis made in section 2.1 regarding the existence of this trade-off between the size of the global minima's possible basin of attraction, and the precision with which points are able to successfully localise to the minima.

My final point of investigation concerned the effects of the iteration count. One would of course expect the performance of either algorithm to only increase given more time, but to what extent? Figure 14 depicts my findings. The hill-climber sees great improvement initially, however this very quickly tapers off. When we considering the differences in the order of magnitudes labelled on the *x* axis, this line seems to resemble that of a logarithmic, which would not come as a surprise considering the stochastic nature of the hill-climbing algorithm. As for the gradient descent, its success rate seems to climb in a much more linear fashion, in accordance with its deterministic nature, this is supported by the sudden increase towards the *x* limit when the jump in magnitude is taken in to consideration.

Figure 14 – How success rate varies for each algorithm with increased iteration count