**Simulated Annealing vs Gradient Descent**

I.        Introduction/Motivation/Problem Statement

Simulated Annealing (SA) is a well-known, stochastic, global search optimization algorithm that competes with Gradient Descent, to be utilized in backpropagation, to optimize convergence of Neural Net Classifiers (NNCs). This task is vital to the efficiency and efficacy of the decision-making process of Neural Networks (NN). With the millions of potential inputs and weighted decisions that need to be made per layer of the NN, even milliseconds of improved performance could result in hours or even days of saved training time via minimizing computational cost overhead. We need to explore the difference between the two and the pros and cons of each performatively and cost-effectively. We will compare along a 3-dimensional search space by optimizing an objective function.

    II.        Related Work

There are many studies published on cooling schedules and learning rates for simulated annealing and gradient descent, respectively. The purpose of this study being a comparative analysis between the two algorithms, we determined the best approach was to utilize existing methods as opposed to developing a novel approach to each.

  III.        Contributions

Standard approaches to each algorithm were utilized as a stable means to compare hyperparameters such as initial temperature (T0), final temperature (Tf), cooling factor (k), and various learning rate co-factors, to be listed below. A more stable approach was conducive to a more thorough comparative analysis of metaheuristic factors such as cooling schedules and learning rates for their respective functions.

I.        Approach

For the purpose of comparative analysis, we will be utilizing the Rastrigin function (1) to test objective minimization techniques. The reason we chose this function is that it has great utility not only for multidimensional analysis, as the function is n-dimensional, but for testing an algorithm's ability to avoid local minima as well. *T*his is because the function's surface is riddled with local minima, as we would expect from a *multimodal,* n-dimensionally concave (i.e., non-convex) function[1]*.* For our two-dimensional input space, we will utilize the variable names x and y as opposed to x1 and x2 reflected below.

****

**(1)**

The primary factor in determining the efficiency and efficacy of each algorithm is its metaheuristic. For simulated annealing (SA) this is the cooling schedule (CS) while for gradient descent (GD) this is the learning rate (LR). The basic form of each algorithm is to start from a random point in our search space, (x, y) on the domain [-5.12, 5.12], according to the bounds of the objective function. We then initialize our hyperparameters for SA as follows: initial temperature (T0), final temperature (Tf), and our cooling factor (k). For GD these involve the final gradient (Z), our cooling factor (k), and the following metaheuristic-based hyperparameters: period (T), cycle (M), the factor to reduce scaling LR by (reduction), and the number of iterations at which the scaling LR is static (rate). We then proceed according to the following algorithms until they terminate.

Simulated Annealing

𝑠 = 𝑠0

𝑇 = 𝑇𝑚𝑎𝑥

while T > Tf (until our terminal condition is met):

𝑇←𝑐𝑜𝑜𝑙𝑖𝑛𝑔(𝑇, 𝑘)

𝑠𝑛𝑒𝑤←𝑛𝑒𝑖𝑔ℎ𝑏𝑜𝑢𝑟(𝑠)

if 𝐸𝑠 ≥ 𝐸(𝑠𝑛𝑒𝑤):

𝑠←𝑠𝑛𝑒𝑤

otherwise:

random (1) < eE(s) - E(snew)T

𝑠←𝑠𝑛𝑒𝑤

Output is the final solution *𝑠*

Gradient Descent

𝑠 = 𝑠0

for i < 1,000:

Fx

Fy

LR←learning\_rate(k, i, t)

x = x - LR(Fx)

y = y - LR(Fy)

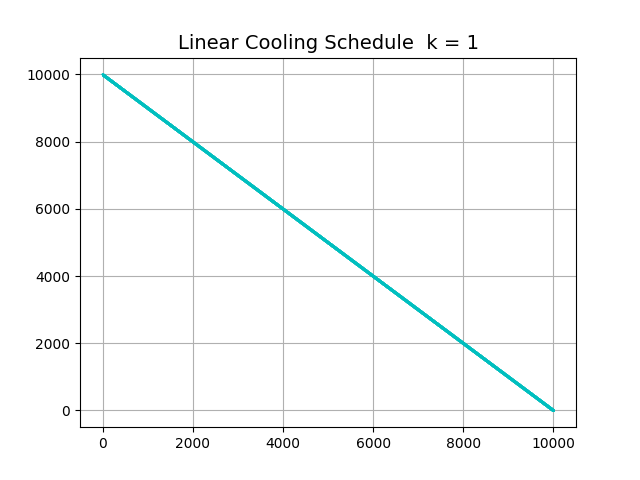
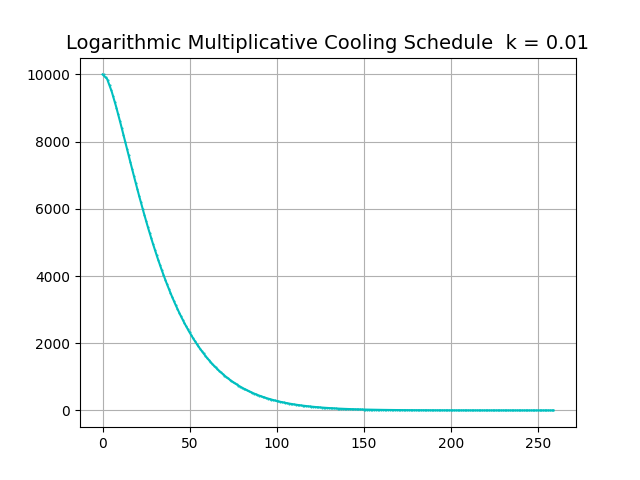
*z* ← rastrigin(x, y)

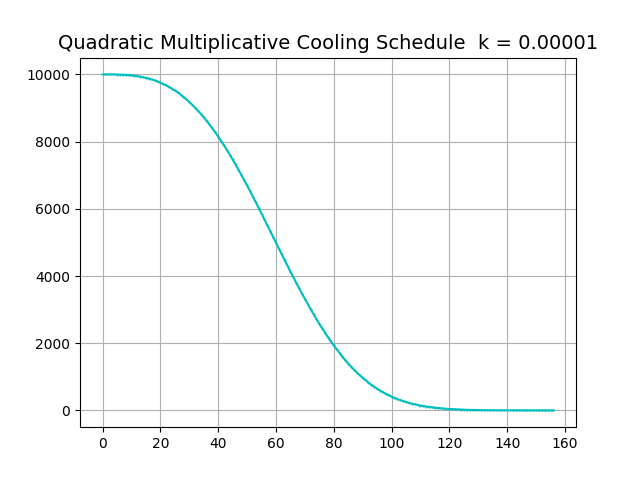
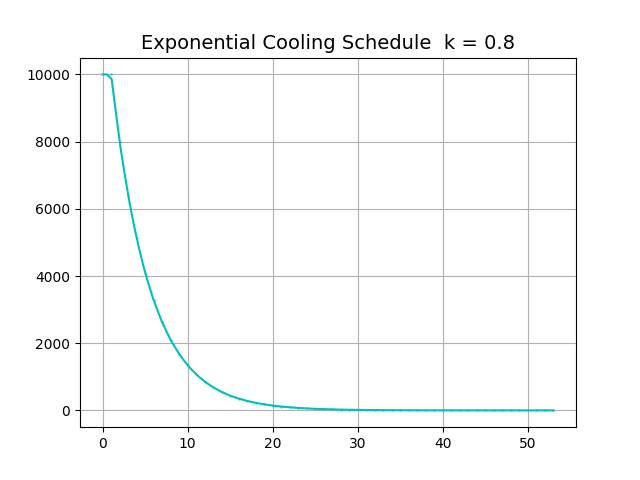
Output is final solution *z*

Every algorithm must have a stopping point, be it when the global optima are found, when the trajectory of the algorithm has stagnated, or when an explicitly stated terminal condition is met. As such, the terminal condition for SA is our final temperature approaching near-zero values to avoid a divide by zero error in computation. This is because of the inability of the arithmetic logic unit (ALU) to store and compute floating-point numbers beyond an order of magnitude relative to its word size. It also generalizes well to the terminal conditions found in physical annealing. The terminal condition of the GD algorithm is set for a stagnation of five consecutive equivalent values for Z, implying stagnation. It will later be determined that the oscillation, albeit small, around minima does not consistently allow for this and a fixed number of iterations becomes our new terminal condition to be set at 1,000 iterations.

    II.        Experiments/Results

Our experiments consisted of exploring 4 metaheuristics per algorithm. These include four cooling schedules and four learning rates, for SA and GD respectively. This was performed in conjunction with various cooling factors and learning rates to find the optimal metaheuristic for our chosen search space. The SA metaheuristic was implemented across four different cooling schedules with the following consistent hyperparameters: T0 set to 10,000 and Tf set to 0.01. Various values of k were used in association with the following SA cooling schedules: logarithmic multiplicative, linear, exponential, and quadratic multiplicative cooling schedules as shown in Figure 1.





**Fig. 1**

The GD metaheuristic was implemented across four learning rates with a universally static k-value of 0.001. These learning rates are as follows:  static, scaling, cosine, and periodic learning rates. An iteration of either algorithm was determined to be accurate if it fell within one unit of the global optima. For the objective function this optimum exists 3-dimensionally at (x, y, z) = (0, 0, 0). To garner a more realistic determination of their behavior, we ran 10,000 trials of each algorithm under all four of their respective metaheuristic modes, cooling schedules and learning rates. An average accuracy across all trials was then calculated, per mode, for each algorithm based on the number of trials that ended within the accuracy constraint. This value was then averaged, by division of the total number of trials run. Logarithmic cooling in Figure-2 was determined to be the best cooling schedule for SA with an accuracy of 17.5% while linear cooling in Figure-3was determined to be the worst at a mere 0.3% accuracy. While various learning rates were used for GD, it did not seem to affect its accuracy. We found that GD in Figure-4 performed nearly identical across all learning rates that did not involve momentum. The mean accuracy was calculated to be approximately 1.32%.

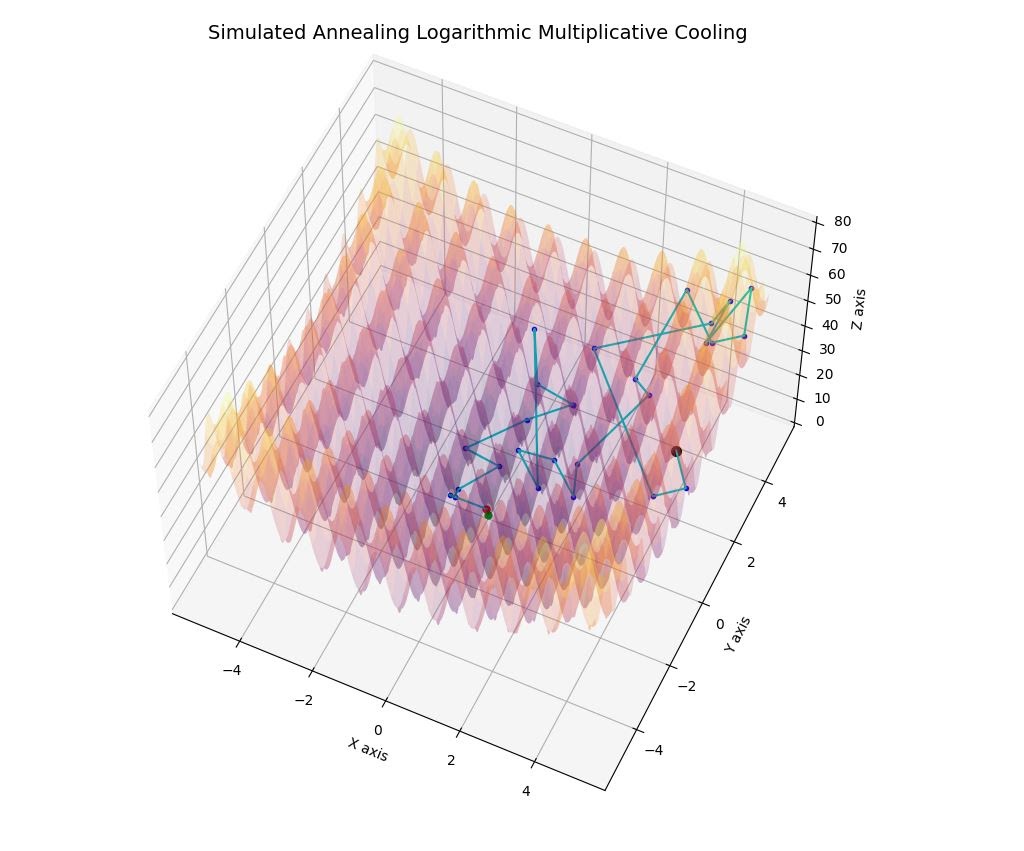


Figure 2 - S.A.: one trial of logarithmic multiplicative cooling, where k = 0.01

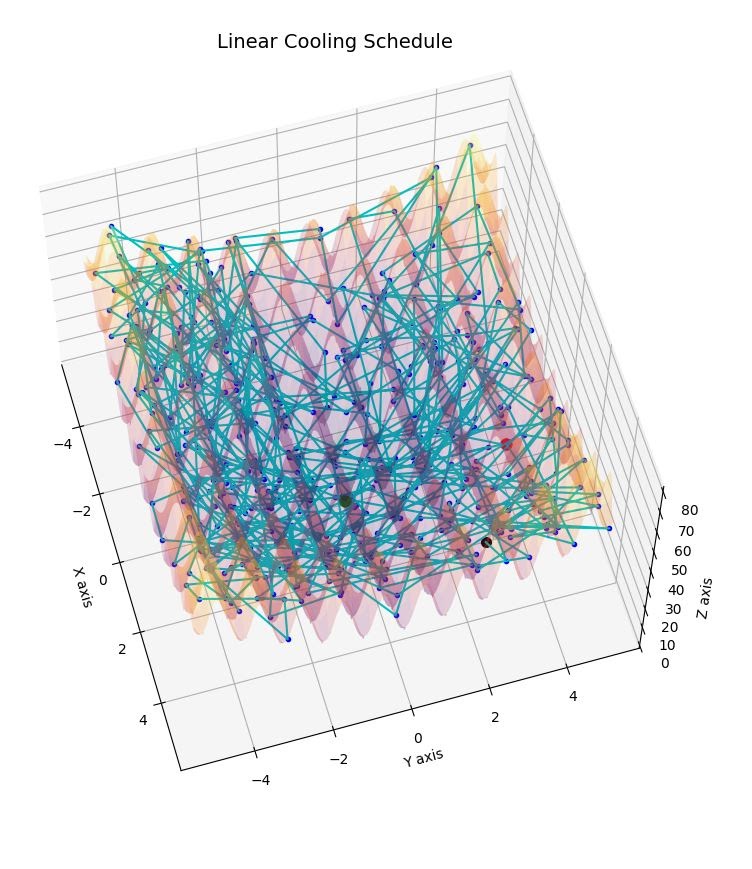


Figure 3 - S.A.: one trial of linear cooling, where  k = 1

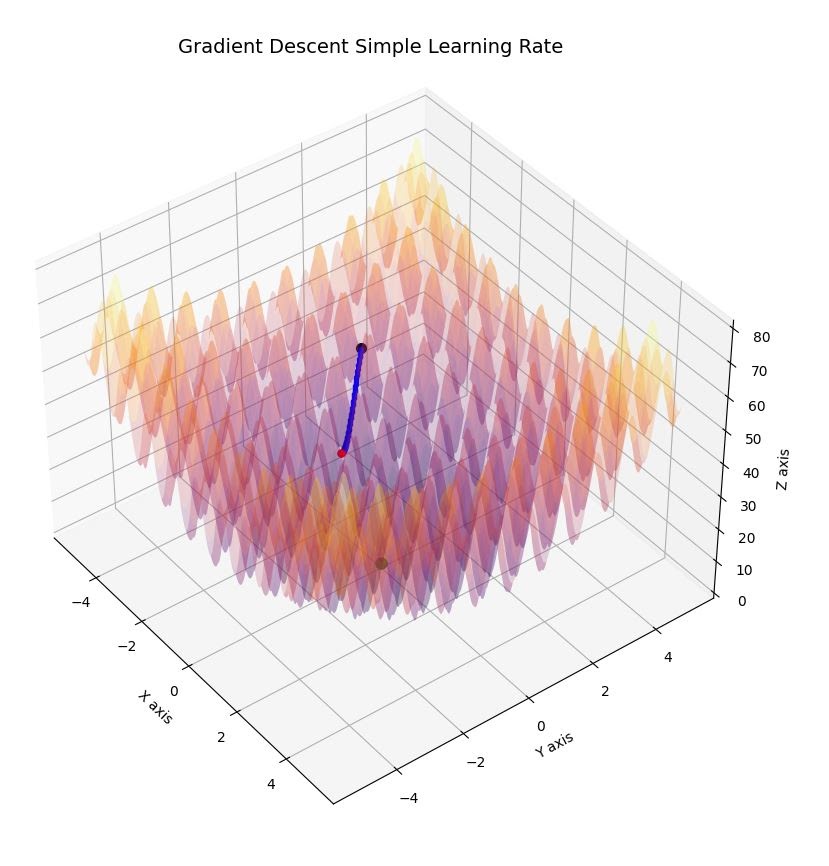


Figure 4 - G.D.: one trial of static learning rate, where  k = 0.001

 III.        Conclusion

The results bear conclusive evidence that simulated annealing outperforms gradient descent about assessing global optima. The accuracy across all trials within each metaheuristic mode seemed low for simulated annealing (SA). A reason for this could have been due to the accuracy constraint being too restrictive; within one unit. Additionally, the computation cost overhead was prohibitive in that, on a slow machine, the computation cost of running so many trials across so many experiments did not allow for an exhaustive exploration of cooling factors (k-value). After our initial work, prior presentation, more time was given to explore cooling factors across the best and worst-performing cooling schedules. It was found that with a cooling factor 100 times smaller than the original k=0.01, the logarithmic multiplicative cooling schedule performed with 99.96% accuracy across 10,000 trials while a linear cooling schedule performed with 19.7% accuracy across an equal number of trials. The difference in accuracy is attributable to the amount of time spent at high temperatures vs lower temperatures. This is because our probability of accepting worse solutions is directly proportional to the current temperature. In minimization problems, higher temps push for exploration while lower temps push towards exploitation. The difference here is how quickly one metaheuristic reaches lower temperatures than another. It may be clear to the reader that logarithmic multiplicative cooling dominates linear cooling, on such a noisy surface, however, there is no one-size-fits-all metaheuristic for optimization and some problems may require problem-specific solutions. The reader may imagine a function with many flat surfaces benefiting from increased exploration of the total search space.

The initial experiments with gradient descent (GD) did not include a momentum-based metaheuristic. This was rectified and results produced that ran counter to what we assumed they would be. Throughout 1,000 trials, momentum-based gradient descent was performed with 0.0% accuracy since it would shoot past local and even global minima should the momentum (P) rate be set too high and caught in local minima with a P-value too low (0 < P 0.8). Another approach was attempted by providing a weighted average of the prior gradients. However, utilizing the second derivative caused stagnation points uncharacteristic to gradient descent, at the inflection point. The implementation must be incorrect and belongs to the future work of the authors, alongside some relatively lesser-known approaches for cooling schedules, such as the *Lundy & Mees*, *variable cooling factor (VCF)*,[3] *and Stochastic Tunneling (STUN), cooling factors* [2].

**References:**

[1] “Difference between Convex and Non-convex | Convex vs Non-convex.” Difference Between. http://www.differencebetween.info/difference-between-convex-and-non-convex (accessed: May 02, 2021).

[2] Haeno H, Maruvka YE, Iwasa Y, Michor F (2013) Stochastic Tunneling of Two Mutations in a Population of Cancer Cells. PLOS ONE 8(6): e65724. <https://doi.org/10.1371/journal.pone.0065724>

[3] Peprah, A. , Appiah, S. and Amponsah, S. (2017) An Optimal Cooling Schedule Using a Simulated Annealing Based Approach. *Applied Mathematics*, **8**, 1195-1210. doi: [10.4236/am.2017.88090](https://doi.org/10.4236/am.2017.88090).