Gradient Descent on Scalar Field Datasets

Zinn Morton*
Oregon State University

Mishary Alotaibi*
Oregon State University

1 Abstract

Finding the smallest scalar value for a given dataset is necessary for various scientific visualization applications, such as creating contour-based visualization. In these applications, the smallest scalar value can directly influence the outcome's quality. This paper examines the use of gradient descent on scalar field datasets to find the lowest scalar field within the given dataset.

As the basis of our system, we provide a method for calculating the gradient in 2 dimensions in a scalar field dataset and present an efficient algorithm to identify minima and maxima. At the core of our analysis is the observation that the finding of a global maxima/minima can be dependent on the selected source point.

Keywords: scientific visualization, scalar field, gradient descent, maxima, minima.

2 Introduction

To find minima in the scalar field, one can use a gradient descent algorithm. This will require a couple of things.

Firstly, it is important to note that, in our dataset, the scalar field values are only given at certain points within the field. Thus, linear interpolation is required to find the scalar value at any point in the field. Any coordinate will be within a quadrant, surrounded by four points with given scalar values.

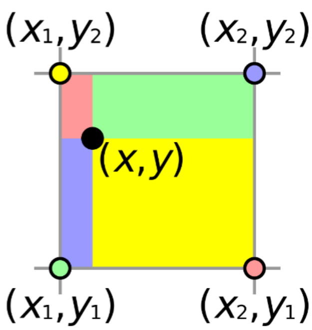

Figure 1. Linear Interpolation (Zhang).

We can use linear interpolation to find a scalar value of the given point by using the four scalar values of the quadrant it is in. This is given by the following formula, where f(x,y) is the scalar value at that point on the quadrant:

$$f(x,y) = \frac{x_2 - x}{x_2 - x_1}\frac{y_2 - y}{y_2 - y_1}f(x_1,y_1) + \frac{x - x_1}{x_2 - x_1}\frac{y_2 - y}{y_2 - y_1}f(x_2,y_1) + \frac{x_2 - x}{x_2 - x_1}\frac{y - y_1}{y_2 - y_1}f(x_1,y_2) + \frac{x - x_1}{x_2 - x_1}\frac{y - y_1}{y_2 - y_1}f(x_2,y_2)$$

For the gradient descent algorithm, the gradient is also needed. The two partial derivatives of the interpolated scalar value will be:

$$\frac{\partial f}{\partial y} = \frac{x_2 - x}{x_2 - x_1}\frac{-1}{y_2 - y_1}f(x_1,y_1) + \frac{x - x_1}{x_2 - x_1}\frac{-1}{y_2 - y_1}f(x_2,y_1) + \frac{x_2 - x}{x_2 - x_1}\frac{1}{y_2 - y_1}f(x_1,y_2) + \frac{x - x_1}{x_2 - x_1}\frac{1}{y_2 - y_1}f(x_2,y_2)$$

$$\frac{\partial f}{\partial x} = \frac{-1}{x_2 - x_1}\frac{y_2 - y}{y_2 - y_1}f(x_1,y_1) + \frac{1}{x_2 - x_1}\frac{y_2 - y}{y_2 - y_1}f(x_2,y_1) + \frac{-1}{x_2 - x_1}\frac{y - y_1}{y_2 - y_1}f(x_1,y_2) + \frac{1}{x_2 - x_1}\frac{y - y_1}{y_2 - y_1}f(x_2,y_2)$$

Using linear interpolation, one can find these two partial derivatives, and can therefore also find the gradient at any given point. This gradient can be used in the gradient descent algorithm.

3 Previous Work and Background

Gradient Descent is an algorithm that takes repeated steps in the opposite direction of the gradient of a function at the current point because the direction of the gradient goes towards the local/global maxima. In contrast, the opposite direction goes towards the local/global minima.

Zheng, Pawar, and Goodman [J. Zheng 2018] have provided a pseudocode in which gradient can be used for graphs (Figure 2). However, their problem had an information visualization dataset, and our experiment uses scalar field datasets. Hence, we decided to simplify their implementation by removing the annealing schedule feature, and our algorithm focuses on finding the minima/maxima of a scalar value.

Algorithm 1: Stochastic Gradient Descent

```
SGD (G):
    inputs: graph G = (V, E)
    output: k-dimensional layout X with n vertices
    d_{i,j} ← ShortestPaths(G)
    X ← RandomMatrix(n, k)
    for η in annealing schedule :
        foreach {i, j : i < j} in random order :
            μ ← w_ij η
            if μ > 1 :
                μ ← 1
            r ← (||X_i − X_j|| − d_ij)/2 · (X_i − X_j)/||X_i − X_j||
            X_i ← X_i − μ r
            X_j ← X_j + μ r
```

Figure 2. Stochastic Gradient Descent (Zheng, Pawar, and Goodman).

4 Results

Alongside the implementation of gradient descent, we have also visualized the path the algorithm takes. This can allow one to see the behavior of the algorithm after it has ran. For our visualizations, the yellow point is the initialization location. The purple point is the result of gradient descent, and the light blue point is the result of gradient ascent. Additionally, the traced lines become more blue as it descends, and more red as it ascends.

*e-mail: {mortonzi|alotaimi}@oregonstate.edu

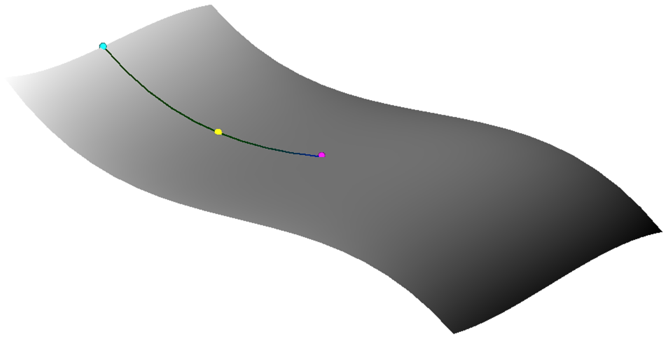

Figure 3. Scalar field gradient descent and ascent ran on r2.ply (5, 0).

As can be seen, this simple implementation of gradient descent is volatile to changes in the initialization location, and can get stuck at local minima. As seen in Figure 3, the algorithm gets stuck at the flat area in the middle, and doesn't find the global minima.

To address this issue, one could choose to run the algorithm multiple times with different initialization points, and choose the lowest minima. To showcase this, we have ran gradient descent and ascent at 39 different locations:

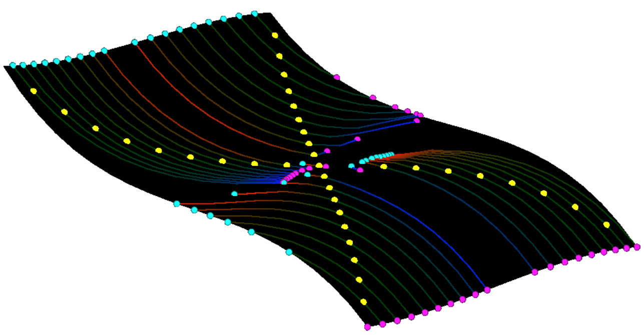

Figure 4. Gradient descent ran at 39 locations in r2.ply.

We have also created plots showcasing scalar value vs. iterations. As expected, scalar value increases as ascent runs and decreases as descent runs. We can see a plateau behavior with descent, as it likely has reached some minima.

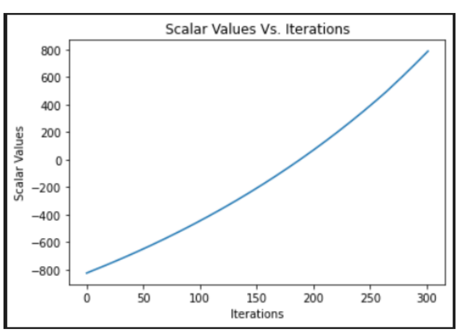

Figure 5. Gradient ascent run on r2.ply.

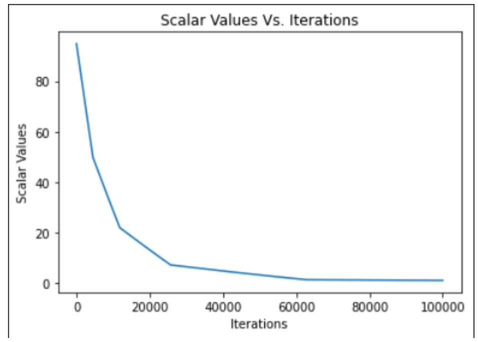

Figure 6. Gradient descent ran on r2.ply

5 Evaluation

One could also find minima and maxima on these datasets by comparing a vertex's scalar value to its neighbor's. If a vertex's scalar value is lower than all neighbors, it can be declared a local minimum. Similarly, if a vertex's scalar value is greater than all neighbors, it can be declared a local maximum. This only works because we have a finite number of coordinates, and scalar values in between are linearly interpolated. We can therefore compare the results from the gradient descent algorithm to those that we found before using this method.

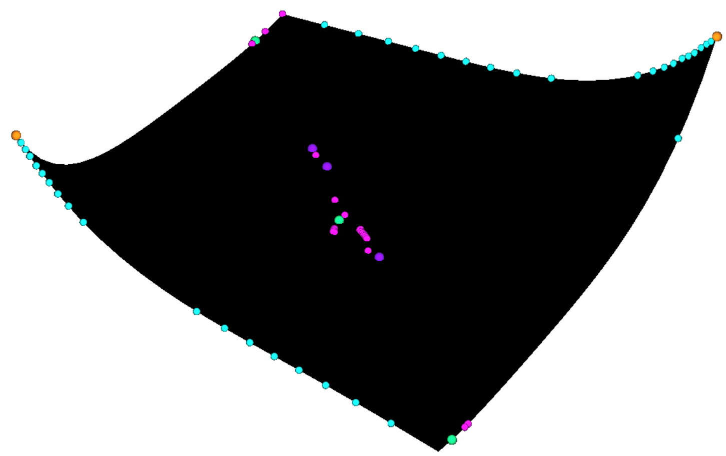

Figure 7. Purple points and light blue points are minimums/maximum candidates found through gradient descent/ascent respectively. Light green points and orange points are minimums/maximums found with the method describe above, respectfully. Dataset is r9.ply.

As can be seen above, the two methods find mimimums and maximums at similar scalar values. It is important to note, however, that gradient descent is sensitive to initialization. In this case above, we ran it many times starting at varying locations. If we wanted the global minimum, we could chose to take the lowest point that we found.

6 Division of Tasks

Mishary Alotaibi:

- 2d visualization of scalar field vs. iterations.
- Abstract
- Previous Work and Background.
- Results
- Acknowledgements

Zinn Morton:

- Gradient descent implementation.
- Introduction.
- Results
- Evaluation
- Conclusion

7 Conclusion

As seen above, gradient descent can be used to find minimums and maximums in a scalar field dataset. This paper's implementation can also stand as a proof of concept of sorts, since gradient descent can be used for far more complicated datasets with higher dimensionality, as seen in Zheng et. al's paper. Additionally, gradient descent can be used in scalar fields with continuous values. This is particularly important, since the other method of finding mimumums and maximums cannot be used in that case.

8 Acknowledgements

We would like to thank the following people for the gradient descent algorithm: Jonathan Zheng, Samraat Pawar, and Dan Goodman. We also want to thank Professor Eugene Zhang for helping us by proposing a formula for linear interpolation to find a scalar value of a given point and dataset to use.

References

J. ZHENG, S. PAWAR, D. G. 2018. Graph drawing by stochastic gradient descent. *IEEE Transactions on Visualization and Computer Graphics 25*, 9, 2738–2748.

ZHANG, E. 2022. Cs453 lecture notes 8.