Opti II: Piecewise Linear Function Approximation

Lucy Malmud and Diran Jimenez

April 2025

1 Formulation

Given a set of points $(x_1, y_1), \ldots, (x_n, y_n)$ that define a piecewise linear function, we hope to construct an approximation by only selecting a subset of these points. We will construct a graph $G = (V, E)$ to represent this underlying collection of points and associated costs for approximating a particular segment.

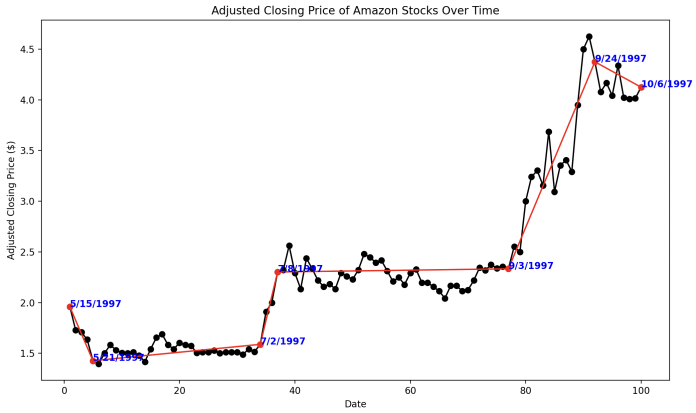
Let α be the storage cost associated with adding another segment and β be the error cost (will be the scale value of the deviation between the approximation and true function value). Let $f_1(x)$ be the true value of the function defined by the original set of points.

- Insert a vertex v_i for each point (x_i, y_i)
- For each pair of points $(x_i, y_i), (x_j, y_j)$ where $j > 1$ insert an edge (v_i, v_j) with cost $c_{ij} = \alpha + \beta \sum_{k=1}^{j} (f_1(x_k) - f_2(x_k))^2$ where $f_2(x)$ is the linear function through (x_i, y_i) and (x_j, y_j)

We then run Dijkstra's Algorithm on this network in order to find the shortest path from (x_1, y_1) to (x_n, y_n). An assumption of this model is that the approximated function must have the same endpoints. The vertices traversed along the shortest path represent the points that will thus define the piecewise approximation.

2 Application: Financial Data

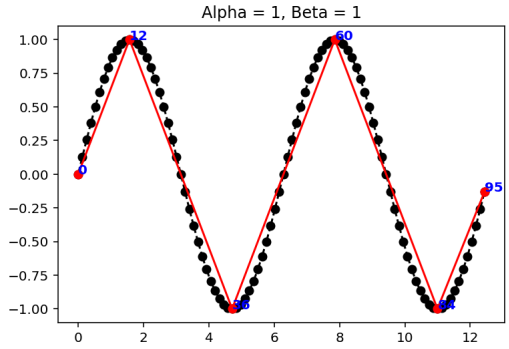
One such instance for which it may be useful to utilize a piecewise linear approximation is in financial data. This model would not be useful in order to predict stock prices due to their inherent volatility, but rather assess periods of time for which there may have been more "steady" trends, as demonstrated by segments between two points.



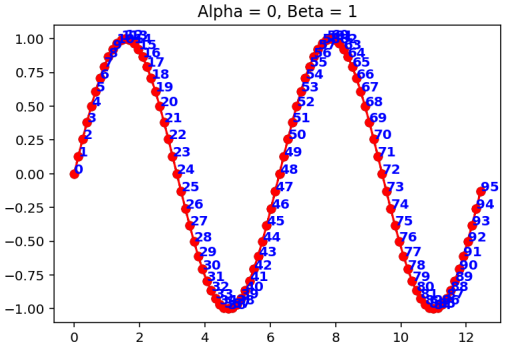
In the figure above, we plot the adjusted closing price of Amazon stocks over time. It is of note that this dataset is comprised of daily stock prices, and the above plot only shows a subset of 100 points. The algorithm as currently implemented is quite slow, so a further application would be to admit an optimized version.

3 Exploration: Sin

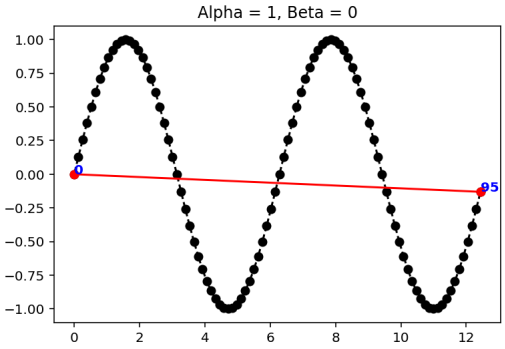
The relative values between α and β determine how many or few points are included in the approximation. This can be clearly demonstrated using the sin function. As a simple case, consider when $\alpha = 1$ and $\beta = 1$:



The approximation includes 4 points besides the intial and final points. However, what happens when $\alpha = 0$?



The approximation includes every point! Recall α represents the cost of storing an additional point. Since it's zero, it makes intuitive sense that the approximation should include every point. There's no downside! What about the opposite case?



Now the approximation includes no points! Again this makes intuitive sense. β represents the scale of the cost of having a poor approximation. If there's no cost to having a bad approximation, then it's cheapest to include as few points as possible.

Finally, how does the number of points change as α and β vary. Since only their relative values matter, α can be held constant while β is increased from 0. The results of simulation can be seen in Table 1.

The behavior is not smooth: at very specific values of β the number of points jumps, which increases at a rapid rate. Plots for different values of β can be seen below.

β Range	Number of Points in Approximation
0-0.09	2
0.1-2.31	6
2.32-14.22	10
14.23-15.27	13
15.28-58.97	14
58.98-100	18

Table 1: Beta vs. Number of points in approximation

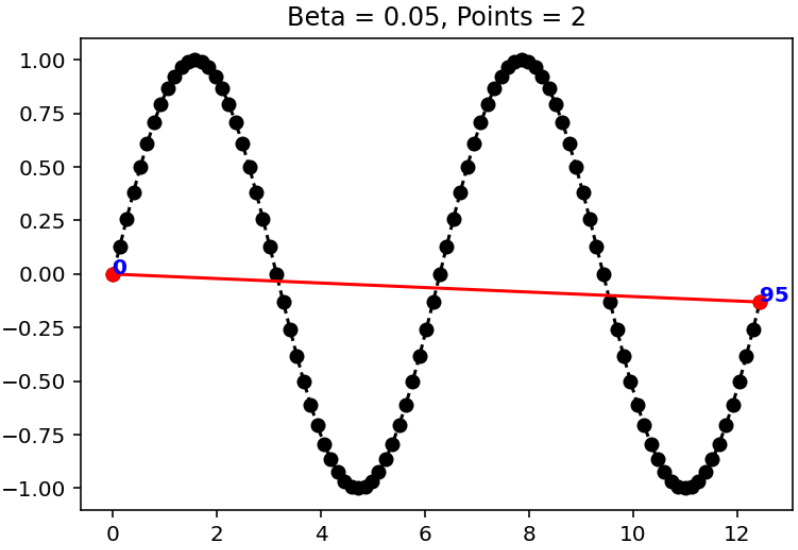


Figure 1: Comparing Beta