Interpolation Methods

Caleb Koch

August 2016

This is a brief document summarizing the work that I've done over the summer. Crystal requested that I do something like this before writing a formal article (which I will be working on after this).

Motivation for Work

Linearly interpolating trajectories can yield highly inaccurate results.

Identifying potential methods

Notes and Remarks on Potential Interpolation Methods

The paper *A stable and fast implementation of natural neighbor interpolation* by Liang & Hale (2010) does a good job articulating an important distinction that should be made when considering various interpolation methods. Specifically, the authors note that some interpolation methods use scattered data. These types of interpolants accept pairs of the form $(\mathbf{x}_i, f_i)$ for $i = \{1, 2, ...\}$ such that $\mathbf{x}_i \in \mathbb{R}^n$ is a n-tuple and $f_i \in \mathbb{R}$ is the corresponding real value. Typically, this data would represent some characteristic of a multidimensional surface (for example, rainfall values over a mapped area). Hence these particular interpolating methods (i.e. ones that rely on scattered data input) wouldn't work well with trajectory data.

Barnes Interpolation

Not appropriate for trajectory or time series data since it typically interpolates an atmospheric variable in the form $f(x, y)$. Typically f gives some parameter which can be used with coordinates x, y to create an contour plot.

Spline Interpolation

This method includes linear and cubic spline interpolation. It can be applied to both the time series and trajectory data.

Polynomial Interpolation

This method can be used on trajectory and time series data.

Bilinear

Conventionally, bilinear interpolation would not be used with trajectory or time series data; however, the principle behind it (performing linear interpolation in both directions) could be adapted to compute interpolants on the desired datasets.

Kriging

Kriging is used when the data reflects output fitted to a particular map. It would be used for reasons similar to that of Barnes interpolation.

Principal Curve Detection

This method would work best with a "data cloud" through which one constructs an optimal curve. Hence the format of the trajectory data makes it difficult to use with principal curve detection.

Frequent Trajectory Mining

This method is outlined in the paper *Frequent Trajectory Mining on GPS Data* (2010) by Savage et al. It could be appropriately applied to trajectory data; however, the algorithm would have to be reconstructed since it doesn't seem the authors have published any code related to their study.

Neural Network/Machine Learning and Probabilistic Modeling

These methods still stand as potential means of interpolating trajectory data.

Nearest Neighbor

This method can be applied to both time series and trajectory data.

Natural Neighbor

This method relies on an implementation of a Voronoi diagram. If one of these diagrams can be constructed from the dataset then natural neighbor interpolation can be performed. This works best with scattered data.[1]

Inverse Distance Weighting

This method also applies to scattered data. It is similar to natural neighbor interpolation in that it implements a weighting procedure.

Latent Statistical Model

GPS Trajectory Data Enrichment Based on a Latent Statistical Model (2016) by Kinoshita et al. outlines the latent statistical model. The model estimates the mode of transportation at various parts of the trajectory and uses such information to yield more accurate interpolations. Their experimental results report a 78% accuracy. So, this may be a potential method to implement; however, it does not seem that the authors have published any code accessible to the public. Nonetheless, their paper includes ample detail regarding model construction to build a separate implementation.

[1]

Experimentation

Pattern mining involves the extraction of nontrivial similarities within and between moving object trajectories. The current pattern mining project at the Geospatial Research Laboratory has sought to achieve this extraction via mSEQUITUR - a recursive algorithm that discovers patterns by generating a formal grammar. In order to infer a formal grammar, the datasets must undergo a number of preprocessing steps (specifically, interpolation and linearization), and must ultimately represent whole, complete sequences of data values. Hence, data values must be interpolated when gaps exist in the input trajectories. Previously, the research team has been performing this interpolation as an initial preprocessing step. Namely, all gaps in the trajectories were linearly interpolated before the dimensionality was reduced via a linearization technique (e.g. Hilbert space-filling curve). However, this interpolation technique presented difficulties due to its inaccuracy. Our research this summer began with quantifying this inaccuracy and determining whether other interpolation methods could offer more accurate results. We also considered whether linearization should occur before or after interpolation. Making these assessments involved developing a standard experimentation framework with which various interpolation methods could be cross-analyzed - not only when they were applied to trajectory data but also when they were applied to linearized data. Over the period of this internship, five interpolation techniques were tested using an implementation of this framework. The results showed not only that interpolation methods worked better when applied to linearized data but also that nearest neighbor interpolation yielded the most accurate results. After ensuring that these results were statistically significant, it could be concluded that the current preprocessing techniques for pattern mining are suboptimal. In the future, nearest neighbor interpolation will replace linear interpolation so that the accuracy of the trajectory data can be preserved as much as possible before discovering patterns. Moreover, this research and its results are adaptable and mutable since the implementation of the experimental procedure has been documented and made available on GitHub.

Results

A number of observations can be made from the results.