**Software System for Interpolating Particulate Matter Measurements Extremely Rapidly with Applied Use of Concurrency**

**Authors**

Tim Elam

Roland Heintze

James Ashachik

Kevin McBrayer

***Abstract***

**This project seeks to provide a valuable tool for various analysts in the visualization of pollution data. It works by receiving and parsing an import of particulate matter, or PM. This application then allows for the execution of a spatiotemporal extension interpolation algorithm. This method provides an estimate of the value of PM at a site utilizing spatially sampled data from a Geographic Information System. Additionally, users would be allowed to query a location by latitude & longitude and receive an appropriate interpolated PM value. The application also provides error estimation through 4 algorithms, MAE (Mean Absolute Error), MSE (Mean Squared Error), RMSE (Root Mean Squared Error), and MARE (Mean Absolute Relative Error). This data may then be used for visualization purposes to evaluate the potential values of PM at relative locations.**

**Introduction**

Particulate matter in the context of the EPA’s interest involves particles that are 10 micrometers in diameter or less. These particles pose health risks as they are generally the ones able to enter the throat and nose with the potential of reaching the lungs. They may then cause serious health effects concerning the heart and lungs. This application attempts to provide researchers and others interested, a means of extrapolating data from an input file of PM data and easily visualizing it. From this point, informed decisions may be made from the gathered data.

**Method/Provided Functionality**

1. Import file of measurements.

2. View interpolated value at a specified x, y, and time.

3. Specify how many neighbors(limited 1 to 5) and what p-value to use for the interpolation function.

2. individual queries- The functionality for querying individual data points at designated locations has been added to the GUI.

3. Error Detection- The data sets that are generated from the 9 different interpolation methods along with the original values are run through 4 different methods for estimating errors. The methods are MAE (Mean Absolute Error), MSE (Mean Squared Error), RMSE (Root Mean Squared Error), and MARE (Mean Absolute Relative Error). All interpolated data sets are run through the 4 methods resulting in 36 error estimations that are grouped and printed into error\_statistics\_idw.txt. Conclusions for the best error detection method can be made from the results.

**Conclusions**

Through our efforts, we have provided what we believe to be a reasonable tool for the visualization of PM data. According to our error statistics, we believe that IDW with three neighbors and exponent one to be the most accurate interpolation method. This method had the best scores in all four error statistic categories.

* Optimization

Working with enormous data sets is hard. Optimization is also hard. Because we did not wish to wait for days for data processing to complete, we fine-tuned our algorithms and used some handy built-in tools.

* + Distance vs. Distance Squared

We avoid the extra overhead of using the square root function when searching for nearest neighbors, but use distance when computing measures based on distance.

* + Lazy Initialization

Finding the closest neighbors to a data point runs in O(n) time, so we do not find the neighbors of all of our data points initially- that is deferred until a point’s interpolated value is requested. The neighbors of an object are stored to the object itself to avoid requesting this information again from the data. To avoid having a branch mis-prediction when interpolating all values from a data set, all data points have their neighbors initialized when they are about to be processed for mass-interpolation.

* + Concurrency:

For concurrency we use Java’s ExecutorService to create cached thread pools. These The main thread can then block at each task and wait for its completion. In the case of generating LOOCV files, this allows us to process data and output to a file in tandem, freeing resources in memory for tasks to be completed and preventing users from losing all of their data in the case of operating system-level failure.

* + Results

We are proud to report that processing 140,000 measurements to generate files for LOOCV and error correction has been performed in under three minutes on an Intel(R) Core(TM) i5-2400 CPU @ 3.10GHz with 8GB of DDR3 RAM and a solid state drive for primary storage.

**References**

Li, L., Revesz, P. (2004). “Interpolation methods for spatio-temporal geographic data.” Computers, Environment and Urban Systems 28: 201-227

Refaeilzadeh, P., Tang, L., Liu, H. (2008). “Cross-Validation.” Encyclopedia of Database Systems 532-538

Li, L., Zhang, X., Piltner, R. (2006). “A Spatiotemporal Database for Ozone in the Conterminous U.S.” Thirteenth International Symposium on Temporal Representation and Reasoning168-176

Li, L., Tian, J., Piltner, R., Zhang, X., Holt, J.B. (2012). “Estimating Population Exposure to Fine Particulate Matter in the Conterminous U.S. using Shape Function-based Spatiotemporal Interpolation Method.” GSTF Journal on Computing **1**(4): 24