2B.1 SIMULATION AND DATA ANALYTICS WITH APPLICATION TO VEHICULAR DATA AND OTHER IN SITU SENSING PARADIGMS

Samir Chettri [*], John Evans, Dmitry Tislin
Global Science & Technology, Inc., Greenbelt, Maryland

1 INTRODUCTION

This document discusses the theory and algorithmic considerations for the simulation of mobile environmental and its extensions to non–vehicular data. Furthermore, it broadens the discussion to data analytical methods enabled by the simulated data and applies these methods to real data collected from a fleet of trucks.

Some of the inspiration for ths project comes from the Mobile Platform for Environmental Data (MoPED) system [Heppner et al., 2016], a vehicle-based mobile platform environmental observation network operating in the US. Currently a commercial fleet of trucks, ∼600 at last count, with sensors and communications devices, provides this environmental data to the MoPED system. These vehicles travel interstate, state, local and metropolitan routes, thus are moderately dense in space and time. A denser pattern of data acquisition would be attained if several thousand more trucks were to augment this fleet. The MoPED system processes and performs validation on data received from the fleet and delivers it in real time to the National Weather Service's Meteorological Assimilation Data Ingest System (MADIS). Sample environmental observations include air temperature, road temperature, atmospheric pressure, dewpoint, rain intensity, ozone and light level, typically taken at 10s intervals.

In anticipation of a vastly greater fleet, 10,000 vehicles or more, this study focuses on simulation of ambient temperature and atmospheric pressure along truck routes. For example, such data are produced by within–vehicle original equipment manufacturer (OEM) sensors that conform to the Society of Automotive Engineers' J1939 standard for vehicle networking, the Controller Area Network (CAN) bus, also called J–data in this paper. One way to prepare for this increased data rate and the inherent uncertainties in the data sets is by simulation. The simulator will not be restricted to generation of only temperature and pressure, thus extending its functionality to other mobile weather sensors such as smartphones.

Thus this simulator creates a fleet of anywhere between 1000-10,000 vehicles that start and finish at random times and travel at randomly–varying speeds along a US highway network extracted from a geospatial database, with tables designed by one of the authors. Instrument packages on these virtual platforms sample surface temperatures and pressures from an environmental field produced by the National Centers for Environmental Prediction (NCEP) High-Resolution Rapid Refresh (HRRR) model, that is real–time, hourly with 3-km resolution covering the continental US. Prior analysis of MoPED data [Chang and Pietras, 2015] shows that sensors typically have noise characteristics that lead to drop-outs, drift and physically unrealizable values. Accordingly, this simulation produces temperature and pressure values, obtained from HRRR, that are corrupted with these kinds of noise, Figure 3. Lastly, the simulator collects weather data from the National Weather Services Automated Surface Observation System (ASOS) for comparison, validation, and analytical experiments.

The simulator permits production of a mobile environmental data stream (surface temperature and pressure to start with) enabling studies as below:

1. Data analytics, including comparing noisy truck data with ASOS data for calibration coincident temporally, and in spatial proximity, around 10-km in this paper.
2. Inter–calibration between truck sensors when they are near-coincident spatially.
3. Detection of spatial surface weather structures such as dry lines; for example, with the addition of a relative humidity sensor to the instrument package.
4. The data management infrastructure for mobile environmental data, e.g., geospatial databases, scal-

[*] Corresponding author address: chettri@gst.com

ing with number of vehicles and data structures to keep track of the calibration state of mobile sensors.

5. Expansion of the abovementioned to include other mobile sensors, on, say, smartphones.

2 SIMULATION DETAILS

The simulator has the following high–level steps:

1. Generate tour for a vehicle. This process is repeated for as many vehicles as required.
2. Initialize fixed weather station locations using locations ASOS stations. Temperatures and pressures from ASOS data are treated as "true values," to be statistically compared with values obtained via the instrumentation on moving vehicles.
3. Use HRRR forecast data to generate readings from vehicles and add noise to them to simulate instrument characteristics. HRR data, highway location information and ASOS data are on the same grid.
4. Environmental data gathered from each vehicle are written to a database for further data analytics.

items enumerated above are explained in more detail in what follows.

2.1 Generate vehicle tours

Details are itemized below and pseudo–code follows.

- Open a preexisting table in the PostGIS database (DB) having a list of highways, cataloged by ID and total length. LCC refers tothe standard Lambert Conformal Conic, [Grafarend and Krumm, 2006].

Hwy ID	Length km	LCC proj geom
HID	l_{HNum}	lccgeom

- For each tour assign a vehicle with:
  - A velocity $\in [65 \ \ 110]$ kmh and generate a random start time between 5am and 6am. The velocity shfits ± 5 kmh with every Δt, 10 seconds. Make sure that velocity is always $\in [30 \ \ 130]$ kmh.
  - Two standard deviations (σ), one for temperature and the other for pressure. The temperature and pressure are obtained from a known environmental field – see section 2.3 - and these are identified as the mean μ. Using appropriate (μ, σ) several different temperature and pressure distributions may be generated, i.e., uniform, Gaussian, truncated Gaussian, Gaussian mixture and Cauchy.
- Randomly select highway ID, HID, from the set of HIDs. If the vehicle finishes traversing the highway before the limit time (10pm) is reached, increment the time value $\in [30 \ \ 60]$ min and assign a new HID.
- Initialize vehicle start and end points - choose uniform random numbers $\in \left[0 \ \ l_{\text{HNum}}\right]$ for each location. See Figure 1 for details.
- Generate a query to the DB as in the table below. The table will have as many rows as there are timesteps in the vehicles tours. Since vehicles return environmental data every $\Delta t = 10s$, timesteps will increment by this amount for every step from start time through end time. The Position ("Pos") quantity is in km and "Velo" is the velocity in kmh.

Hwy ID	Truck ID	Time	Pos	Velo
HID	TID	t	LinPos	Vel

The DB will produce a large table

Hwy ID	Truck ID	Time	East	North
HID	TID	t	x	y

The East and North columns refer to easting and northing and are the LCC coordinates.

Pseudocode to generate a vehicle tour is as follows

```
numberOfTrucksToGenerate = input()
Amazon_server.getHighwayData()

loop (for each truck)
{
  assign truck characteristics
  initialize truck position and time
  start time at 5am

  loop (until time reaches 10pm)
  {
    if (truck != moving and truck =
       resting)
    {
      if (done resting): resting =
         false
```

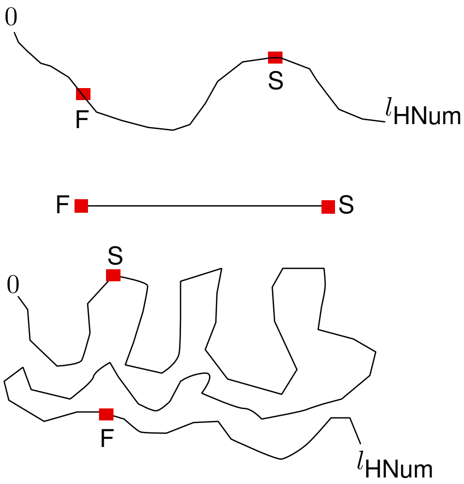

Figure 1: A road as a database object that has a length of l_{HNum}. Start (S) and end (F) points, chosen randomly, are indicated. The middle line is the straightened version of the topmost line from F to S. Knowing velocity, and time increments, distance along the middle (and therefore top) line can be calculated, again from finish to start or vice–versa. The bottom curve shows a more complicated road for which the straightening procedure would also be used.

```
      else: decrease
         rest_time_remaining
    }
    else if (truck != moving and
       truck != resting)
    {
      randomize location
      randomize start and end points
      place truck at start location
      set new speed between 65 and
         110 kmh
      truck = moving
    }
    else if (truck = moving)
    {
      server.store (tid, hwy_id,
         timestamp, linear position
         along highway, speed, other
          truck characteristics)
      truck.update(position)
      truck.update(speed)
    }
    increment time by 10 seconds
  }
  increment truck_count
}
```

2.2 Fixed weather station data - ASOS

In order to get actual temperature and pressure values (not estimations or predictions), ASOS station stored data are retrieved.

1. Station ID, station type, observed time, temperature, and pressure are extracted from the ASOS file and stored in the DB. Since each Station ID is unique to a location, x and y coordinates are retrieved.
2. Assign gridded data form for station data where values are assigned surrouding a station while grid locations that do not contain a station within a specified range are left null.
3. When data values at vehicle's location are found to contain values, distance between corresponding station and truck is calculated. If distance is within the user-specified station influence range, the temperature, pressure, and station ID is stored along with the vehicle's data in the DB.

CAN bus and ASOS temperature data are returned in 1°F increments. This quantization should be part of any data analytical approach to truck and ASOS temperature (or pressure) comparisons.

2.3 Get environmental data as vehicle moves through the physical field

The simulation has produced tours that provide a vehicle with a time stamp, velocity and coordinates in LCC. This folowing uses the truck location (x, y, t) to produce time–varying temperature and pressure fields. The vehicle samples actual temperature and pressure fields from NCEPs High–Resolution Rapid Refresh (HRRR) , a real–time, hourly, 3–km resolution model covering the continental US. A sample HRRR pressure field is shown in Figure 2. A brief description of the process follows:

1. Read in HRRR forecast values for pressure and temperature as GRIdded Binary (GRIB) data. HRRR uses the LCC projection, and highways within PostGIS use the same.
2. The HRRR pixel data has a lower spatial and temporal resolution than the point set of times and truck locations in (x,y); each pixel has

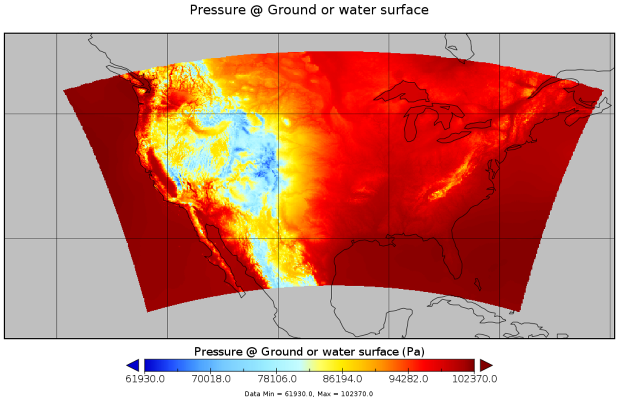
Pressure @ Ground or water surface
Pressure @ Ground or water surface (Pa)
61930.0
70018.0
78106.0
86194.0
94282.0
102370.0
Data Min = 61930.0, Max = 102370.0

Figure 2: HRRR field that is sampled by a vehicle traversing its tour. This data shows the pressures, in Pa, at ground level at a particular time. The simulator requires successive images and performs spatial and temporal interpolation to produce vehicle temperatures and pessures.

a geographic coordinates and a value). The weather fields are available at the discrete set $\{t, t+\Delta t, t+2\Delta t, \ldots, t+j\Delta t, \ldots, t+N\Delta t\}$, e.g., with $\Delta t = 1$ hours and $N = 24$, there are one hour increments and a twenty–four hour period. Truck times belong to the set, $\{t_0, t_0+\Delta t_0, \ldots, t_0+j\Delta t_0, \ldots, t_0+K\Delta t_0\}$, where Δt_0 is typically 10s, i.e., time increments in the weather fields are much larger than in the truck positions, or, $K >> N$. Different time resolutions of the truck and HRRR therefore require temporal interpolation and the different spatial resolutions necessitate spatial interpolation. Temporal interpolation is linear, spatial interpolation requires tan inverse weighted distance (IWD) spatial interpolation [Shepard, 1968], [Gordon and Wixom, 1978] is used. This interpolant uses a weighted average of the values at surrounding data points, with the weighting a function of the distances, typically euclidean, to those points. These steps are performed for each row in the DB until the entire dataset is traversed.

This produces several tours that are indexed by time, (x, y), temperature, and pressure.

2.4 Output

Output is sent to a cloud–based PostgreSQL database organized with the following structure: tid, hwy_id, time, easting, northing, temperature, pressure, near_asos_station_id, near_asos_station_temperature, and near_asos_station_pressure.

Sample graphical output (temperature versus time) from the simulator is shown in Figure 3. The chart clearly shows that ASOS data remains constant for periods of time, and typically changes by one degree increments whereas the simulated truck temperatures have a variation about the a mean (obtained from HRRR) that is given by the standard deviation, σ, chosen in the simulation.

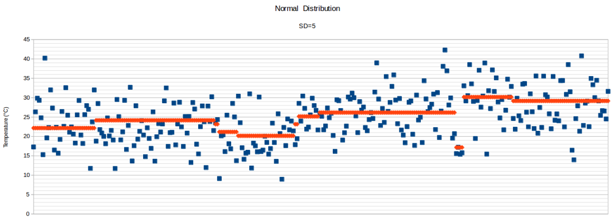

Figure 3: Simulator produced noisy temperature data compared with nearby ASOS (airport) readings. The noise in this graph is defined by $\mathcal{N}(\mu =$ Interpolated HRRR value at truck location, $\sigma = 5)$

3 DATA ANALYTICS

Analysis of actual mobile sensor data by Chao–Hsi Chang and John Pietras of GST [Chang and Pietras, 2015] indicates that even well behaved sensors produce outliers. The scatter plot in Figure 4, adapted from [Chang and Pietras, 2015], shows a clear linear relationship between the temperatures collected from an instrument package installed on the vehicle and from the truck Controller Area Network (CAN) bus, coloquially referred to as J–Data. It clearly shows the effects of noise, both at a low and high levels. For example, the point (20,36) indicated by the symbol ⊕, is clearly a large magnitude outlier. Another aspect of the graph is a quantization effect, clearly seen in the horizontal stepping of data points at one degree increments along the y–axis. The simulator thus reproduces key features of the data described above with Noise models such as the Gaussian, truncated Gaussian, uniform and the Cauchy distributions.

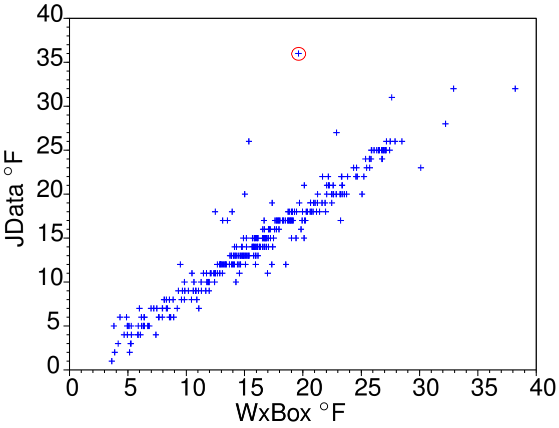

Figure 4: CAN bus data (J–data) versus WxBox temperature data. Large outliers are possible as indicated by the point at (20,36) indicated by the symbol ⊕

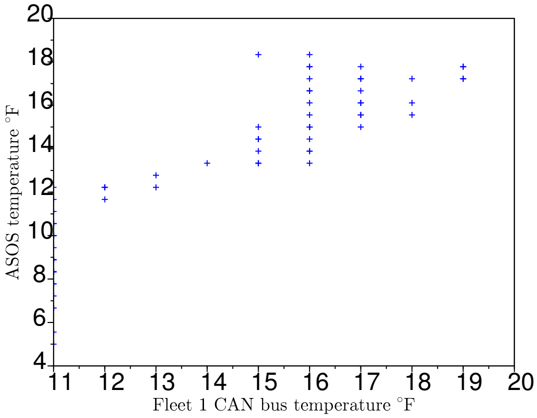

Figure 5: Fleet 1 CAN bus temperature data versus ASOS temperature data (°F). The graph shows temperature data as the vehicle drove within 10 km of ASOS station KEPH; the least distance was 1.33 km and the greatest was 4.95 km.

Figure 5 shows temperature measured using the CAN bus temperature obtained from a vehicle versus ASOS temperature. Several issues may be seen in the plot. While there is a generally evident linear relationship between CAN bus and ASOS temperature, the best fit line is not obvious. The least squares (LSQ) method may be used to fit the line. However, the LSQ method has limitations as enumerated below:

1. An important assumption is that x values are noise–free and only y values have noise.
2. The noise in the y data is drawn identically and independently from a Gaussian distribution.
3. Asymptotically, LSQ has a breakdown point of 0%, i.e., even a single outlier can strongly change the slope and intercept. Figure 6 shows this dramatically.

Inspection of this data indicates the following:

1. The CAN bus and the ASOS temperature data are quantized, meaning temperature readings are returned as $\left[T_{\text{CAN}} - q_1/2 \quad T_{\text{CAN}} + q_1/2\right]$ and $\left[T_{\text{ASOS}} - q_2/2 \quad T_{\text{ASOS}} + q_2/2\right]$ where q_1, q_2 are the quantizing ranges.
2. Outliers are possible along the x–axis and the y–axis and may be non–Gaussian.

Since two key aspects of LSQ are violated, i.e., noise exists only along the y axis and it is Gaussian, something other than LSQ should be used.

The Least Median of Squares (LMS) method [Rousseeuw, 1984] is used for preliminary analysis because it has a 50% breakdown point, i.e., 50% of the data can have outliers before it will produce arbitrary and aberrant values for the estimator, the slope and intercept.The LMS is not the last word in robust statistical analysis; the literature [Rousseeuw and Leroy, 1987] is filled with newer methods [Huber and Ronchetti, 2009] each with its advantages and disadvantages.

Just as the LSQ may be said to be a generalization of the average (mean), the LMS may be considered a generalization of the median. Assuming a straight line model between data x_i and y_i, the equation is:

$$y_i = a x_i + b \tag{1}$$

with $i = 1, 2, \ldots, n$, i.e., there are n corresponding pairs of data (x_i, y_i); a is the slope and b the intercept.Thus the aim of linear regression is to obtain estimates $\hat{\boldsymbol{\theta}} = \{\hat{a}, \hat{b}\}$ of a, b. The residuals, r_i, of a straight line fit are:

$$r_i = y_i - (\hat{a} x_i + \hat{b}). \tag{2}$$

LSQ minimizes the sum of squared residuals, i.e.,

$$\min_{\hat{\boldsymbol{\theta}}} \sum_{i=1}^{n} r_i^2 = \min_{\hat{\boldsymbol{\theta}}} \frac{1}{n} \sum_{i=1}^{n} r_i^2 \tag{3}$$

The least median of squares (LMS) estimator is written as

$$\min_{\hat{\boldsymbol{\theta}}} \ \underset{i}{\text{med}} \ r_i^2 \tag{4}$$

An interpretation of LSQ, equation (3), is that it is a minimization of the average of the squared residuals whereas LMS is the minimization of the median of the squared residuals. Intuitively the LMS should be robust since the median is known to be robust while the mean is not.

Algorithms for LMS are discussed in [Rousseeuw, 1984], [Souvaine and Steele, 1987], [Steele and Steiger, 1986] and [Olson, 1997]. The LMS problem is complex due to the fact that, unlike LMS, the multi–dimensional optimization surface is not smooth and has multiple minima given asymptotically as $\Omega(N^2)$ [Steele and Steiger, 1986]. Straight line regression involves optimizing over only two quantities, i.e., $\hat{\boldsymbol{\theta}} = \{\hat{a}, \hat{b}\}$, so a brute force approach is feasible and takes a fraction of a second to complete. In order to test this algorithm data was used from Figure 4 of [Rousseeuw, 1991], listed below and displayed in Figure 6.

i	x_i	y_i
1	0.4	1.00
2	0.9	1.85
3	1.2	2.60
4	1.6	3.00
5	1.8	3.85
$1'$	5	1

Table 1: Straight line data from Figure 4 of [Rousseeuw, 1991]. The last point, $1'$, represents the conversion of point 1 into an outlier, i.e., $(x_{1'}, y_{1'}) = (5, 1)$.

Figure 6 shows LSQ and LMS fits to the data in Table 1. Clearly the LMS fit is resistant to outliers whereas the LSQ fit breaks down with the slope changing by more than $90°$. Similar results are obrained if the point $1'$ were to move, parallel to the y axis rather than the x axis – this is illustrated in Figures 3 and 6 of Rousseeuw [1991] and is not reproduced in this document.

The method is now applied to temperature data collected over several days in 2016, 17 June and 20–22 June. This is shown in Figure 7 and consists of a total of 493 pairs of points (T_{CAN}, T_{ASOS}) where T_{CAN} is the temperature in deg. C obtained from the vehicle CAN bus and T_{ASOS} is the ASOS temperature. The closest the vehicle was to the ASOS station is 10.62km and the furthest 14.65km. The data shows clustering with the majority of the points running diagonally from lower left to upper right and a second group lying between 24–28 deg. C on the x–axis (UPS temperature) and 28–34 deg. C on the y–axis. These data act to deflect the LSQ fit away from the main body of data. The LMS fit is not affected by these leverage points. Another way to consider the fit to the data is to consider a band around the LMS and LSQ line of ± 2 deg. C along the y–axis. There are 71 points between this band for the LMS line but only 55 points for the LSQ line.

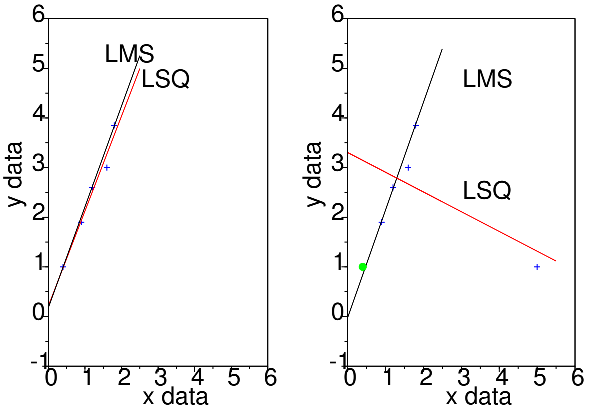

Figure 6: LMS versus LSQ. The left graph shows unperturbed data taken from the first five data points in Table 1 along with LSQ and LMS fits to these. The right graph shows a single point, ●, perturbed along the x–axis – denoted as $1'$ in Table 1. The LMS fit remains close to the original slope and intercept but the LSQ fit is drastically changed due to the strong leverage of this new point.

4 CONCLUSION & EXTENSIONS

This study has two parts. The first part has a discussion the theory and algorithmic considerations for the simulation of environmental data via a fleet of vehicles. To this end code was written that simulated the motion of vehicles through an environmental field making measurements, in this case temperature. Recognizing that data gathering processes may not be perfect, noise is added to the temperature in order to match actual data received from mobile sensors. This simulator was used to create a fleet of 10,000 mobile sensors, emulate their motion across US highways, gather temperature data at ten second intervals, add noise to these, and store these in a cloud–based database. The time taken to simulate these 10,000 vehicles with mobile sensors on a small laptop was less than 10 minutes.

Secondly, a start was made toward the analysis of

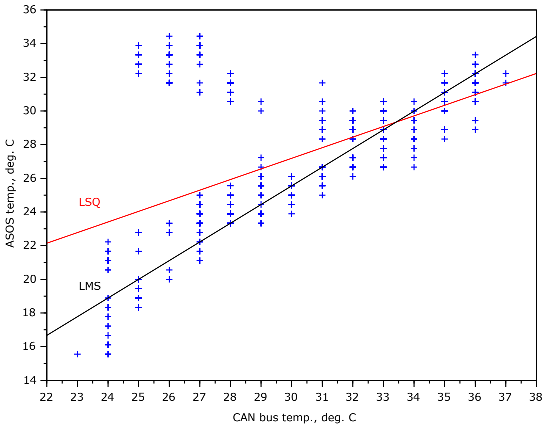

Figure 7: LMS and LSQ method applied to temperature data gathered from ASOS station KPDK (Dekalb–Peadhtree airport in Atlanta) and the CAN bus of a single vehicle. The data shows clustering with the majority of the points running diagonally from left to right and a second group lying between 24–28 deg. C on the x–axis (UPS temperature) and 28–34 deg. C on the y–axis. The second group of points serves to move the LSQ fit toward them; the LMS fit is not affected by these leverage points.

noisy data. Recognizing that the data was visibly contaminated by large outliers, had quantization artifacts and therefore was non–Gaussian, it was decided to utilize robust methods to perform data calibration. Comparative studies were done between LSQ and LMS. The latter has a breakdown point of 50%, i.e., half of the data has to be contaminated with noise before the parameter estimates (i.e., slope and intercept) become arbitrary and aberrant whereas for the former it is 0%. Examples were used to show how LMS was preferable to LSQ.

Where do we go from here? The literature makes it clear that the LMS is merely the tip of the iceberg in terms of data analytical possibilities. An example of a robust, high breakdown point method is the Least Trimmed Squares (LTS) estimator [Rousseeuw, 1984]. Furthermore, we have not considered robust correlation coefficients – this would be the first order of business in any sequel. Another approach would be to consider each sensor (ASOS, CAN bus) to have its own intrinsic noise, *after* which quantization occurs. Ideally the analysis should build an hierarchical Bayesian model with sensor noise, quantization and a Bayesian straight–line fitting method [Preuss and Dose, 2005] all combined together. Finally, each data point collected by a truck and corresponding ASOS data may be considered part of a time–series, e.g., see Figure 3. Certainly the techniques of time–series analysis, whether robust or otherwise, would be appropriate for such data.

References

C. Chang and J. Pietras. National Mesonet Program Evaluation of OEM J-Data from Mobile Platforms. Technical Report GST-NMPA TR-Version 0.09, Global Science and Technology Inc., Greenbelt, MD, July 2015.

W. J. Gordon and J. A. Wixom. Shepard's Method of "Metric Interpolation" to Bivariate and Multivariate Interpolation. *Mathematics of Computation*, 32(141), January 1978.

E. W. Grafarend and F. W. Krumm. *Map Projections: Cartographic Information Systems*. Springer, Berlin, 2006.

P. Heppner et al. Real-time Hazardous Weather Detection and Convective Model Data Assimilation from Mobile Platforms. In *96th American Meteorological Society Annual Meeting*, 2016.

P. J. Huber and E. M. Ronchetti. *Robust Statistics, 2nd Ed.* John Wiley, New Jersey, 2009.

C. F. Olson. An Approximation Algorithm for Least Median of Squares Regression. *Information Processing Letters*, 63:237–241, 1997.

R. Preuss and V. Dose. Errors in all variables. In *Bayesian Infererence and Maximum Entropy Methods in Science and Engineering*, 2005.

P. J. Rousseeuw. Least Median of Squares Regression. *Journal of the American Statistical Association*, 79(388), December 1984.

P. J. Rousseeuw. Tutorial to Robust Statistics. *Journal of Chemometrics*, 5:1–20, 1991.

P. J. Rousseeuw and A. M. Leroy. *Robust Regression and Outlier Detection*. John Wiley, New York, 1987.

D. Shepard. A two-dimensional interpolation for irregularly-spaced data. In *Proceedings of the 1968 23rd ACM national conference*, 1968.

D. L. Souvaine and J. M. Steele. Time– and Space–Efficient Algorithms for Least Median of Squares Regression. *Journal of the American Statistical Association*, 82(399), September 1987.

J. M. Steele and W. L. Steiger. Algorithms and Complexity for Least Median of Squares Regression. *Discrete Applied Mathematics*, 14:93–100, 1986.