Charles Wang
Duke University
Class of '05
ECE/CPS Pratt Fellow

Matching and Locating of Cloud to Ground Lightning Discharges

Advisor: Prof. Steven Cummer

I: **Introduction**

When a lightning discharge occurs in the vicinity of Duke and Clemson University, the discharge emits electromagnetic radiation in the ELF (3-3000Hz) and VLF (3-30 kHz) bandwidth. The electromagnetic radiation will be recorded by ELF-VLF magnetic induction field receivers both Duke and Clemson. For my research project, I will describe a way to match identical lightning discharges in the Duke and Clemson sensor data and then locate the origin of those lightning discharges. I will then verify the matching and locating of the lightning discharges by estimating the error for my calculations with data provided by the National Lightning Detection Network (NLDN). The National Lightning Detection Network is composed of many sensors across the United States. The Network only records strokes with peak current amplitudes of greater than 5kA and the detection probability is between 80 and 90 percent. Therefore, many lightning discharges will not be picked up by the NLDN sensors. Furthermore, the timing accuracy provided by NLDN is only accurate to the millisecond. The sensor data is accurate to the order of 10 microseconds. If we can locate the origin of the lightning discharges with a small error based only on our sensor data, we will then be able to locate many of those lightning discharges not detected by NLDN data.

II: Description of Data

a) The main focus of this research requires the analysis of sensor data at Duke and Clemson. These sensors, controlled by LabVIEW virtual instrument (VI) software, continuously record magnetic field strength at the sampling rate of 100kHz. Each of the sensors contains two channels: one channel that acquires data in the magnetic North-South direction and one channel that acquires data in the magnetic East-West direction. Furthermore, each of the sensors samples a digital GPS signal to ensure the timing of the signal. During a lightning discharge, the sensors will record a "spike" that is easily distinguishable from the ambient noise. Figure 1a presents a sample of this type of data from the sensor at Duke.

b)

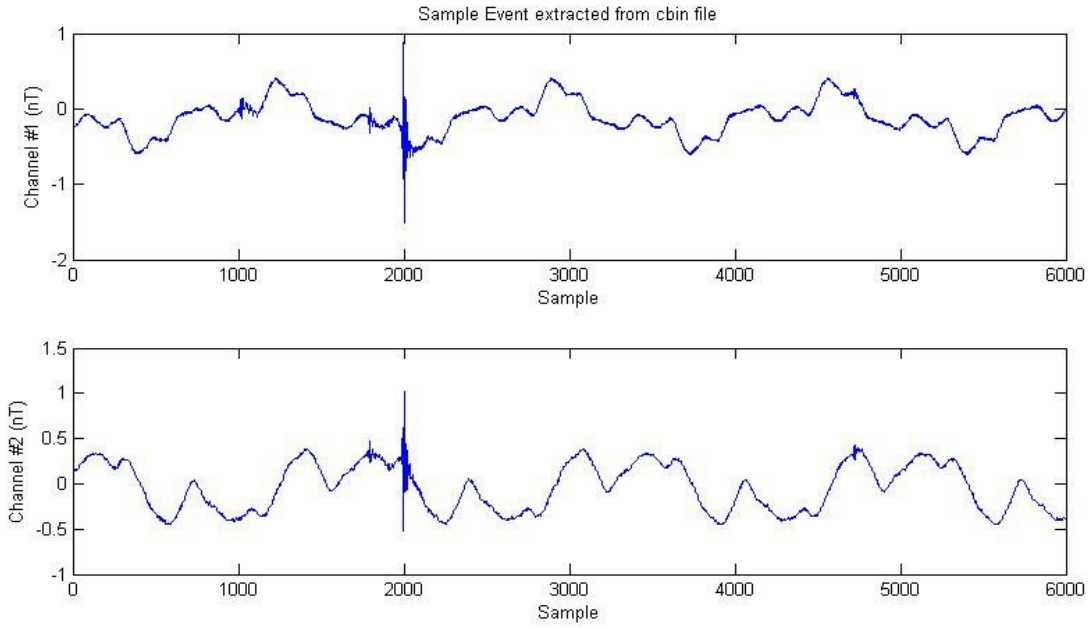

Figure 1a: Sample event extracted from our sensor data.

b) Data provided by the National Lightning Detection Network (NLDN) provides validation methods and error estimates for our locating. This system is composed of a network of sensors that provide time of arrival and direction finding information for cloud-to-ground lightning discharges across the United States in a text-based format. For each lightning stroke, data consists of date, time, latitude/longitude measurements of the source, and peak current. Data for different strokes are separated by a new line. Figure 1b displays a sample for this type of data from March 2nd, 2004.

```
03/02/04 00:00:12.103  43.0982  -86.3097   -14.3
03/02/04 00:00:12.148  43.0977  -86.3084   -18.0
03/02/04 00:00:12.293  43.0976  -86.3092   -21.2
03/02/04 00:00:19.873  43.1134  -86.3013   -11.7
```

Figure 1b: Sample NLDN data taken from March 2nd, 2004. Each line consists of one lightning event.

II: Description of Algorithm

The locating of lightning discharges takes place in two steps. We use the denoised version of the sensor data to apply the locating algorithm since we want as little interference as possible from noise. The noise from the sensor data occurs at 60Hz harmonics. The denoise routine first identifies and masks out lightning discharges and replaces it with averaged noise data. Then, the routine applies bandpass filters to the resulting data at multiples of 60Hz until 2.4kHz and adds the filtered data together. Finally, this filtered data is subtracted from the original data to achieve the denoised data. We use this version of the data because we want the effects of noise in the sensor data to be minimized. Note that this will add a preprocessing time to the time the algorithm takes to run. Future considerations would be to minimize this preprocessing time to optimize the algorithm. Note that all subsequent plots of data will be denoised data, not raw sensor data. Finally, this algorithm will only deal with 10 second time windows since we need to split the entire intersection of the two data files into manageable pieces.

1) We first need to acquire two data files, one recorded by the Duke sensor and the other recorded by the Clemson sensor, which overlap in time. The intersection of these two data files corresponds to the time we are interested in. We then need to create a rule to identify lightning discharges in the data files. Once we have identified the events, we then need to match together corresponding lightning events in the two data files.

2) The second step is to position the matched lightning events recorded by the sensors based on information acquired only through the sensor data. We would like to find the latitude/longitude coordinates of the origin of the event.

III: Identification of events in the sensor data

Events in a data file are identified when the magnitude of the signal recorded by one of the channels of the sensor exceeds a certain threshold. To make sure we only identify lightning discharges, we make sure this threshold is above the level of the noise. Since events have duration of about a hundred samples, after an event is identified, we

skip 1000 samples to make sure we don't identify the same event twice. Refer to Figure 2 for a sample time window and all the events identified by that time window.

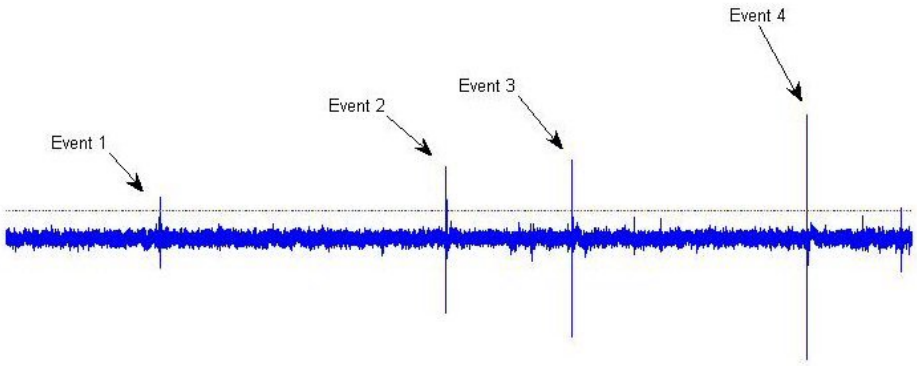

Figure 2: Sample time window of a data file. There are four events identified in this time window. The dotted line corresponds to the threshold level. Currently, the threshold level is at .0753 nT.

IV: Matching of events in the sensor data

Once events are identified in both data files, we wish to match corresponding events. To this extent, we note that the average time between events is many times larger than the time of propagation from the origin to one of the sensors. Therefore, we can match corresponding events based on their proximity in terms of time without worrying if we are matching non-corresponding events. The current routine matches two events if they differ by less than 120 samples, or 1.2 msec. Note that the difference in time between the matches is the difference between the speed of light propagation time from the origin of the event to Duke and speed of light propagation time from the origin to Clemson. Figure 3 illustrates the matching of two events in the two different sensor locations.

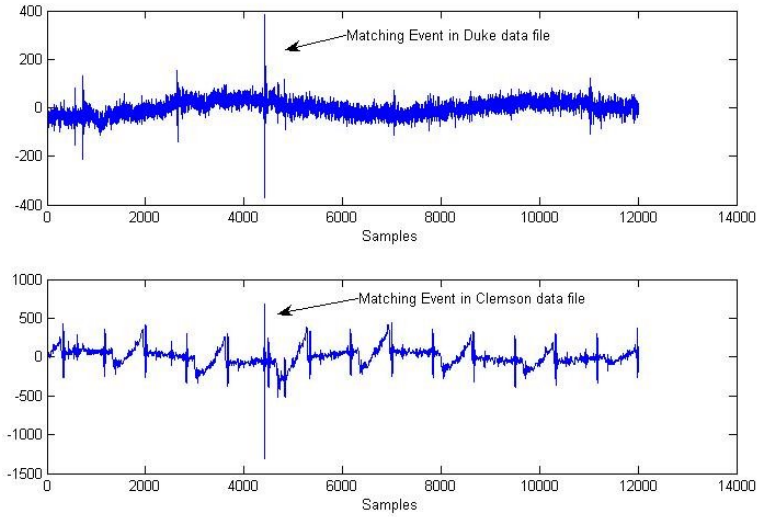

Figure 3: Sample event that was matched in the data file from the Duke sensor and the data file from the Clemson sensor. The time difference of the event between the Duke data file and the Clemson data file is less than 120 samples.

V: Finding the location of a lightning event

There is a constant time difference associated with each matched event. This time difference is equal to the speed of light propagation time to Clemson subtracted from the speed of light propagation time to Duke. We can find the difference of distances by multiplying the time difference by the speed of light. To find where on that locus is the actual point of origin of the event, we have to measure the angle of arrival of the event to the Duke sensor. The angle of arrival is the same as the bearing angle from the sensor at Duke to the origin of the event. To measure the angle of arrival, we first need to convert the sensor data around the event to from magnetic North-South and magnetic East-West to geographic North-South and geographic East-West. We then simply take the inverse tangent of the quotient of the two channels to find the bearing angle. Figure 4 shows the general picture of the locating algorithm.

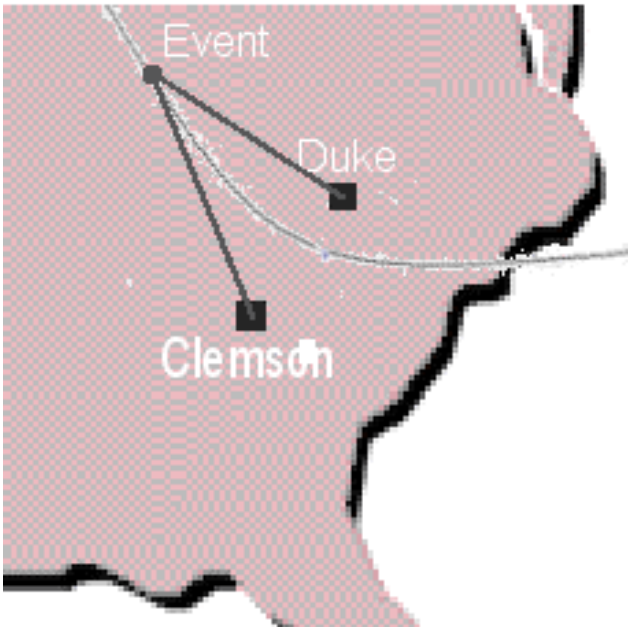

Figure 4: The curve is the locus of all points such that the time difference between the Duke and Clemson sensors is constant. The angle from the Duke sensor will help us calculate the which point on that locus is the origin.

Note that the locus of a constant difference would define a hyperbola in a two-dimensional plane. However, since we are dealing with the surface of a spherical object, we would not obtain a true hyperbola.

The equation used to calculate the distance between two latitude/longitude points, (δ_1, λ_1) and (δ_2, λ_2), is:

$$d = r \cdot \cos^{-1}(\cos\delta_1 \cdot \cos\delta_2 \cdot \cos(\lambda_2 - \lambda_1) + \sin\delta_1 \cdot \sin\delta_2)$$

where $r = 6366.197$ km is the radius of the Earth.
This equation is derived from converting spherical coordinates to Cartesian coordinates and then taking the dot product to find the angle between the two points. The radius multiplied by that angle in radians is the distance between the two points on a sphere. The locus of points we are interested in is every latitude/longitude point (δ, λ) such that:

$$\Delta d = r \cdot \cos^{-1}(\cos\delta_1 \cdot \cos\delta \cdot \cos(\lambda - \lambda_1) + \sin\delta_1 \cdot \sin\delta) - r \cdot \cos^{-1}(\cos\delta_2 \cdot \cos\delta \cdot \cos(\lambda - \lambda_2) + \sin\delta_2 \cdot \sin\delta)$$

where $(\delta_1, \lambda_1) = (35.974, -72.101)$ is the latitude/longitude coordinate of the Duke sensor and $(\delta_2, \lambda_2) = (34.7, -82.8229)$ is the latitude/longitude coordinate of the Clemson sensor and Δd is the measured speed of light propagation difference. Note that this equation usually will produce a locus with two branches. One of these branches corresponds to points closer to Clemson than Duke. The other branch corresponds to points closer to Duke than Clemson. The sign of the measured time difference will determine which branch the point of origin is on. If the time difference is negative, this means that the point of origin of the event is closer to Duke. If the time difference is positive, the point of origin of the event is closer to Clemson.

We use the routine that calculates the bearing angle from the Duke sensor to the origin of the event to find our measured angle of arrival. The equation of the bearing angle between two points on a sphere would give us a second locus of points. This equation is:

$$\alpha = \tan^{-1}\left(\frac{\sin(\lambda_1 - \lambda)}{\sin\delta \cdot \cos(\lambda_1 - \lambda) - \tan\delta_1 \cdot \cos\delta}\right)$$

where $(\delta_1, \lambda_1) = (35.974, -79.101)$ is the latitude/longitude coordinate of the Duke sensor and α is the bearing angle obtained from the routine.
We are interested in all points (δ, λ) that satisfy this equation.

The measured location of the origin is the intersection between these two loci of points. Figure 5 shows a more exact, graphical picture of this locating algorithm.

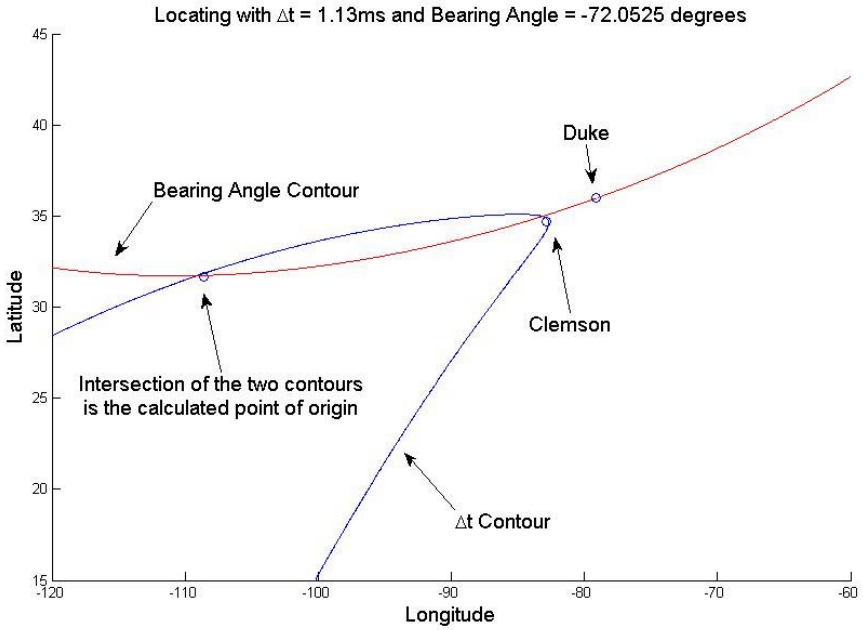

Figure 5: This is a plot of the two loci that was generated with $\Delta t = 1.13$ *msec* and $\alpha = -72.0525°$. The intersection of these two loci is (31.7290,-109.0691), the calculated latitude/longitude location of the point of origin.

VI: Technical Issues

Issue 1: Events in data files are identified when their magnitude exceeds a certain threshold, not when the event starts.

When we identify events in the data files, the timing of the start of the event is not identified. Instead, the timing of when the event exceeds a certain threshold is identified. This could produce some minor problems since NLDN data is based on the start time. However, the errors are lessened because we are dealing with time differences, so some of the timing errors caused by the difference between the start time of the event and the threshold time of the vent are subtracted out. But to try to minimize, we have developed a MATLAB script that attempts to figures out the first section of the event before the threshold where the signal waveform starts to "smooth over." Figure 6 shows the input and output of this MATLAB script.

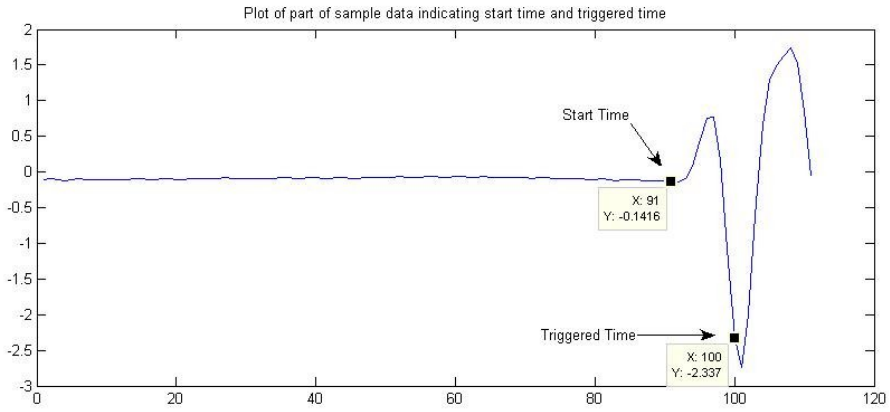

Figure 6: The MATLAB routine tries to find the Start time of the event from the Triggered time and the signal waveform. From the plot, the Start time is approximately the correct part of the waveform where the smooth part of the signal ends.

The MATLAB code for this routine is reproduced below:

```
function testpoint = getStart(data)

%data has to be 6000 samples long and the triggered time is at sample 2000.

threshold = .1;        %threshold slope

start = 1960;          %start with a time 40 samples before the triggered time
triggeredTime = 2000;

a=start;
b=triggeredTime;

testpoint = floor((a+b)/2);     %first test the midway point

for i=1:8 %test a maximum of 8 points.
   if(abs((data(testpoint-1)-data(testpoint+1))/2) > threshold)
      %if the midway point has a slope greater than the threshold slope,
      %the start time is before the midway point
      b=testpoint;
      testpoint = floor((a+b)/2);
   else
      %else the start time is after the midway point.
      a=testpoint;
      testpoint = floor((a+b)/2);
   end
```

Issue 2: There exists a periodic unwanted signal in the Clemson data files.

There exists a periodic signal in the Clemson data files by visual examination. Since this periodic signal cannot be an event, the threshold level for Clemson data files will not be consistent with the threshold level of the Duke data files. Figure 7 displays an example of this periodic signal.

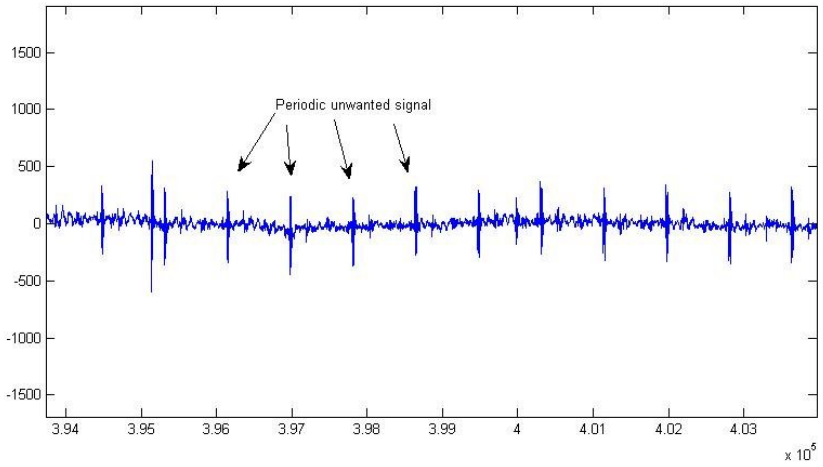

Figure 7: Segment of a Clemson data file displaying the periodic unwanted signal.

In order to have consistent threshold times in both Duke and Clemson data files, we need to somehow remove this unwanted periodic signal from the Clemson data. In order to achieve this, we need to obtain the time when the period starts in our data file segment and set that part of the signal equal to zero. By examination, the period of the unwanted signal is 833 samples. To find the unwanted signal of the first period, it is sufficient to take the cross-correlation between unwanted signal and the 999 samples of the data. The maximum of this cross correlation corresponds to the time when the unwanted signal in the first period occurs. We can then mask every period since we know the length of each period. For better performance, the MATLAB routine identifies the unwanted signal every half of a second through the cross-correlation method. This masking method fairly effective since it takes out most of the unwanted signals in the Clemson data file. We can then utilize the same threshold for event identification in both the Duke and Clemson data files. Figure 8 displays a 10 second window of the Clemson data file before masking and after masking.

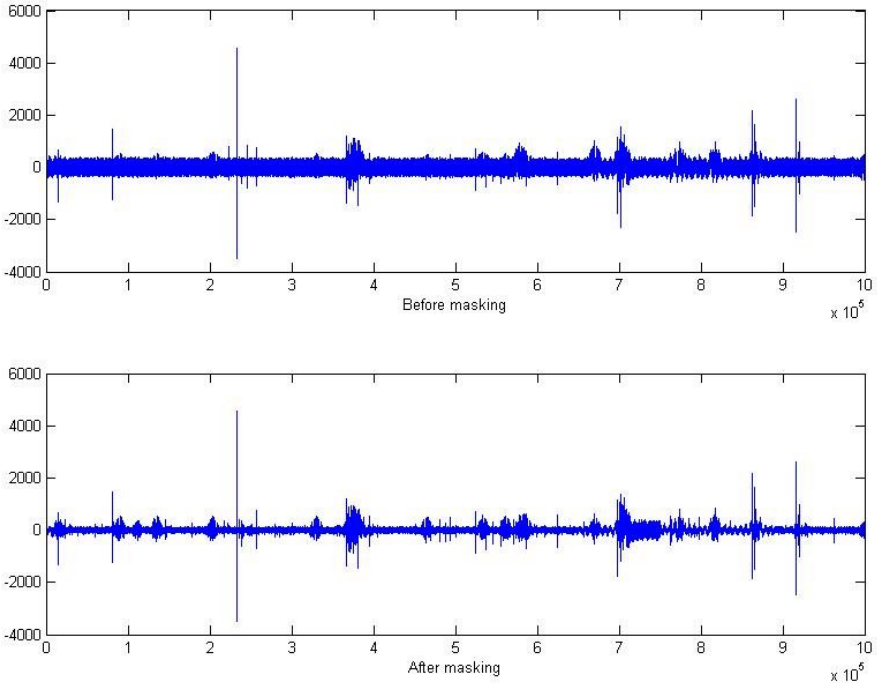

Figure 8: Before masking and after masking plots. As is evident, most of the unwanted signal is removed after masking.

Below is the code that masks a ten second window in the Clemson data file (a 2 by 1000000 matrix named data2)

```
for k=1:2  %mask both channels

   mask2 = ones(1,1000000);          %initialize the mask vector
   for j=1:20                        %mask 0.5 second at a
                                     %time (10 seconds total)
      start = 50000*(j-1)+1;
      stop = start+999; %check first 999 samples

      corr = xcorr(data2(k,start:stop), unwanted);
      [m ind] = max(abs(corr));      %max of xcorr is the index of the first
                                     %signal

      i=start+ind-1000+28;

      while(i+30<50000*j)
         mask2(i-30:i+30) = 0;  %fill up mask vector with zeros when
                                %there is an unwanted signal.
                                %unwanted signal is 60 samples long.
         i=i+833;       %unwanted signal is spaced 833 samples apart (8.33ms)
      end
   end
   datamask2(k,:) = mask2.*data2(k,:);  %masked data is masked vector
                                        %multiplied element-wise by the data

end
```

VII: Analysis of Data

We need analyze our data to make sure that we are performing the matching and locating correctly. To do this, we use the NLDN data to check our results. This occurs in two steps.

1) To check whether we are matching events correctly, we try to match events in our data files with events in our NLDN data using the same matching algorithm as between the two sensors. We can then calculate the actual time differences for those events and compare them to the measured time differences. If a high percentage of matches exist in the NLDN data and the actual and measured time differences correspond, we know we are matching events correctly.

2) To check whether we are locating events correctly, we try to locate events that are exhibited in both the sensor data and the NLDN data. Therefore, we can calculate the average distance between the calculated location and the actual location of those events.

VIII: Results

Below is a sample of the matches obtained when we attempt to match the sensor data with the NLDN data:

Duke time	Clemson time	Δt Measured (ms)	Δt Expected (ms)	Δ(Δt) (ms)
268:19:59:52.8707	268:19:59:52.8696	1.04	1.0569	0.016879
268:19:59:52.9002	268:19:59:52.8991	1.07	1.0568	-0.013158
268:19:59:52.9901	268:19:59:52.9889	1.2	1.1982	-0.0018403
268:19:59:53.0166	268:19:59:53.0154	1.2	1.199	-0.0009723
268:19:59:53.7155	268:19:59:53.7144	1.15	1.1535	0.0035465
268:19:59:53.8016	268:19:59:53.8004	1.14	1.1536	0.013635
268:19:59:53.8521	268:19:59:53.851	1.15	1.1536	0.0035688

The measured time difference is obtained by subtracting the Duke time from the Clemson time. The expected time difference is obtained by subtracting the time of propagation from the source of the event to Duke from the time of propagation from the source to Clemson. This is done using NLDN data. The last column lists the measured time difference subtracted from the expected time difference. Based on the results, the matching algorithm is accurate when we try to compare the matches to NLDN data. In fact, the error is never more than 1 or 2 samples (10-20 *usec*).

The locating algorithm needs to be optimized before being tested. Currently, locating one event takes about three minutes of time. Since there can be over 100 matches for a ten second window, this is prohibitively long. Even if we restrict ourselves to those matches

that also show up in NLDN data, the locating would still take over thirty minutes. This substantial amount of time is due to the complexity of single-linkage clustering, which we need to perform in order to find the intersection of the two contours described in Figure 5. The complexity is on the order of the product of the elements contained in the two contours, which is well over one million. Also, note that the locating is very sensitive to changes in bearing angle due to the sharpness of the Δt contour. If the calculated bearing angle is off by even a few degrees, the locating would perform poorly. However, for the sample event in Figure 5, the actual latitude/longitude coordinates are (31.6791,-108.5922), which is only 49.8 km. off from the calculated latitude/longitude coordinates of (31.7290,-109.0691). This is a low error since the event is 2,753km away from Duke.

References:

Heavner. M. October 10th. 2000. NLDN Description http://nis-www.lanl.gov/nis-projects/edot/ildc2000/node5.html. Accessed April 26th, 2005.

Acknowledgements:

-Prof. Steven Cummer

-Zhenggang Cheng, PhD Student

-Dean Martha Absher