Project #3: GNSS Data Processing and Analysis

Ryan Nguyen and Timothy Kehoe

ESSE 3670

March 22, 2021

Executive Summary

In this lab, the effects of processing parameters on satellite observation data was analyzed. This required data processing with precise point positioning for six individual stations using different scenarios. We selected three Canadian stations: ALGO00CAN, DUBO00CAN, WHIT00CAN, and three international stations: GOLD00USA, WUHN00CHN, IISC00IND, and first mapped where the stations were located. The scenarios that were compared using data for each station were: for 24 hours using kinematic processing mode, 24 hours using static processing mode, 30 minutes using kinematic processing mode, and 30 minutes using static processing mode. Using data from all stations, these four scenarios were processed through on two days, resulting in 48 datasets in total. Analyzing the effects of the processing parameters, we created tables, charts, and histograms to discuss the results. The effects discussed were how static and kinematic processing mode affected the time it takes for the horizontal and vertical error to be less than 5 centimeters, the effects of how the data arc length affects these errors, comparing the results of each scenario and drawing conclusions. We found out that static produces more precise results than kinematic but is not as accurate and vice-versa. Static was faster in reaching less than 5 centimeters of error and produced smoother results than kinematic. The reason static produced better results is expected to be based on how the GNSS treats the processing modes in the CSRS-PPP engine. We also found out that the shorter the data arc length was, the smoother their data would be. We concluded that this was because that the data arc length covers a timeframe where it is unlikely for the stations to have shifts in movements, so it is uncommon that we would see spikes or noise in their data. Using data from two different days for the different scenarios gave us more proof to support our conclusions.

Table of Contents

1 Introduction

The purpose of this lab is to gain experience in geodetic GNSS data processing and data analysis. To do this, RINEX files of GNSS observation data for six stations around the world on two adjacent will be obtained. Then, this data will be edited and processed using NRCan's Precise Point Positioning service. All six stations have a dataset with 24 hours and 30 minutes of data, processed in both kinematic and static modes, over two days (January 2nd, 2021 and January 3rd, 2021). With all processed datasets obtained, the effects of various processing parameters including processing type and data arc will be analyzed by comparing the estimated station coordinates with published reference coordinates. All references to Appendices are referring to files submitted externally with the lab.

2 Methodology

2.1 Obtaining Data

Files containing GNSS observation data for all six stations on two adjacent days were first obtained from the SOPAC & CSRC Garner GPS Archive in .crx format. Three stations are located in Canada, and the other stations are located in China, India, and the United States. To convert to a usable RINEX format, the provided *crx2nrx.exe* executable file was used. Then, these twelve files were duplicated and edited to included data arcs of both 24 hours and 30 minutes. Next, the NRCan CSRS-PPP service was used to process each dataset in both kinematic and static Precise Point Positioning processing modes.

2.2 Description of Acquired Data

Prior to editing the RINEX files or processing the data with the NRCan CSRS-PPP service, the obtained RINEX files contain raw observation data for satellites in both the GPS and GLONASS constellations. Each file contains satellites' broadcast ephemeris' at epochs over 24 hours, as well as a variety of parameters used for point positioning.

2.3 Data Processing

Once files were duplicated and edited to contain both 24-hour and 30-minute data arcs, the NRCan CSRS-PPP service applies the method of relative positioning to provide precise positioning data, including the estimated latitude, longitude, and height of a specific receiver (station). In the kinematic processing mode, the receiver (station) is treated as if it is in motion, and a greater number of satellites over multiple epochs are required to compute the station position estimate. In the static processing mode, stations are treated as if they remain in the same position, and fewer satellites are required over a single epoch to compute an estimate of the station position.

3 Data Analysis and Discussion

3.1 Analysis Overview

Once datasets were obtained, all analyses were completed using MATLAB, and provided code was used to read each RINEX file. Desired plots included the number of satellites and DOP as functions of time, a time series showing the evolution of horizontal and vertical error, histograms including the vertical and horizontal RMSE for specific results, and histograms including the time it took specific solutions to reach 5 cm of vertical and horizontal error.

Though the provided *readPos.m* was capable of reading a RINEX file and returning various types of data including time epochs, GDOP at epochs, and ECEF coordinate at all epochs, additional data was required. The *readPos.m* file was edited to return error in East, North, Up instead of ECEF XYZ coordinates, both horizontal and vertical error at each epoch, total horizontal RMSE, total vertical RMSE, and the time it took any solution to reach 5cm of horizontal/ vertical error. Additionally, this function was used to develop plots of GDOP and number of satellites as a function of time. The edited *readPos.m* file will be described further in *Data Processing for Individual Files* and can be found in Appendix 1.0.

Since *readPos.m* obtains data from individual RINEX files, another file called *main.m* was developed for analysis purposes. Here, all RINEX files were first organized by station, data arc length, processing mode, day of acquisition, and various combinations of the mentioned categories. By organizing files in this way, vectors could be developed that included, for example, the horizontal RMSE for all datasets with a 30-minute data arc, or the vertical RMSE for all datasets processed in static mode. These vectors were then used to develop histograms to analyze the effects of data arc length and processing mode on position estimates. In addition, *main.m* was developed to plot overlapped time series of horizontal or vertical error in specific combinations of datasets for further analysis. The *main.m* function will be described further in the *Effects of Data Arc Length and Processing Mode on Position Estimates* section and can be found in Appendix 2.0.

3.2 Data Processing for Individual Files

In order to draw meaningful conclusions from the datasets, the first step was to modify the provided *readPos.m* file to output variables relating to the quality of the data in each individual RINEX file. These variables include time epochs in a usable format, position error, and summations of position error. Detailed descriptions are provided in the following four sections, along with brief sections of *readPos.m*.

3.2.1 Map with locations of Stations

The stations position was given in ECEF XYZ coordinates. Converting the six station coordinates to latitude, longitude, and height, we can plot the stations position on a world map using the MATLAB add-on tool, geoplot (Figure 1).

```
ALGOlat = 45.9558;

ALGOlong = 281.92863;
```

```
DUBOlat = 50.25881;

DUBOlong = 264.13382;



WHITlat = 60.75051;

WHITlong = 224.77788;



GOLDlat = 35.42516;

GOLDlong = 243.11075;



WUHNlat = 30.53165 ;

WUHNlong = 114.35726;



IISClat = 13.02117 ;

IISClong = 77.57038;



lat = [ALGOlat DUBOlat WHITlat GOLDlat WUHNlat IISClat]';

long = [ALGOlong DUBOlong WHITlong GOLDlong WUHNlong IISClong]';



geoplot([ALGOlat, DUBOlat, WHITlat, GOLDlat, WUHNlat, IISClat],[ALGOlong, DUBOlong,
WHITlong, GOLDlong, WUHNlong, IISClong],'^')
geobasemap streets

title('Map of Locations of Stations');
text(ALGOlat,ALGOlong,'ALGO00CAN');
text(DUBOlat,DUBOlong,'DUBO00CAN');
text(WHITlat,WHITlong,'WHIT00CAN');
text(GOLDlat,GOLDlong,'GOLD00USA');
text(WUHNlat,WUHNlong,'WUHN00CHN');
text(IISClat,IISClong,'IISC00IND');
```

Figure 1 – Map of six selected stations

3.2.2 DOP and Number of Satellites per Epoch

To determine the GDOP at station receiver locations at each epoch for a specified file, the *readPos.m* function iterates line by line and stores GDOP per epoch in a data structure called *solutions*:

```
solutions.GDOP(epoch_index, 1) = str2double(split_line(index_GDOP));
```

Next, the number of available satellites at each epoch is read from the RINEX file and stored in *solutions*:

```
solutions.num_sat(epoch_index, 1) = str2double(split_line(index_numSat));
```

Finally, since time is read from the RINEX file in Hours, Minutes, and Seconds (HMS), these values were converted to decimal hours and stored in a single vector called *decimalHour* in *solutions*:

```
hms = split_line{index_HMS};

solutions.time(epoch_index, :) = [str2double(hms(1:2)) str2double(hms(4:5))
str2double(hms(7:11))];

% decimal hours (used for time series instead of just HMS)
```

```
solutions.decimalHour(epoch_index, :) = [str2double(hms(1:2)) +
str2double(hms(4:5))/60 + str2double(hms(7:11))/3600];
```

3.2.3 Horizontal and Vertical Error per Epoch

To determine horizontal and vertical error at each epoch, the estimated latitude, longitude, and height of the corresponding station are first read from the RINEX file. Once obtained, the provided *llh2XYZ.m* function (Appendix 3.0) is used to convert from latitude, longitude, and height to ECEF XYZ coordinates. These llh and XYZ coordinates are then stored in *solutions*:

```
[X, Y, Z] = llh2XYZ(lat_degree, lon_degree, height);

solutions.llh(epoch_index, :) =  [lat_degree lon_degree height];

solutions.ECEF(epoch_index, :) = [X, Y, Z];
```

However, since the error is computed using East/North/Up, a function called *XYZ2enu* (Appendix 4.0) was written to transform between the two systems. *XYZ2enu* makes use of the following equation:

$$\begin{bmatrix} e \\ n \\ u \end{bmatrix} = \begin{bmatrix} -\sin(\lambda) & \cos(\lambda) & 0 \\ -\cos(\lambda)\sin(\phi) & -\sin(\lambda)\sin(\phi) & \cos(\phi) \\ \cos(\lambda)\cos(\phi) & \sin(\lambda)\cos(\phi) & \sin(\phi) \end{bmatrix} \begin{bmatrix} X_{estimate} - X_{reference} \\ Y_{estimate} - Y_{reference} \\ Z_{estimate} - Z_{reference} \end{bmatrix}$$

This function was then called to store East/North/Up differences in a matrix called *enu* in *solutions*:

```
% enu relative to reference station
[e, n, u] = XYZ2enu(stations(station,1), stations(station,2), stations(station,3), X,
Y, Z, lat_degree, lon_degree);
    solutions.enu(epoch_index, :) = [e, n, u];
```

Finally, horizontal and vertical error are computed as:

$$h_{error} = \sqrt{e^2 + n^2}$$

$$v_{error} = |u|$$

These values are computed at each epoch and stored in *solutions* as follows:

```
% horizontal error
hor_error = sqrt(e^2 + n^2);
solutions.hor_error(epoch_index, :) = hor_error;

% vertical error
```

```
vert_error = abs(u);
solutions.vert_error(epoch_index, :) = vert_error;
```

3.2.4 Total Horizontal and Vertical RMSE

To compute the total horizontal and vertical RMSE for a singular dataset, the sum of each error over all available epochs was first required. To compute this, error values were initialized outside of the epoch-by-epoch loop:

```
hdiff = 0;

vdiff = 0;
```

Within the epoch-by-epoch loop, each value was iteratively updated:

```
% sum of differences used for horizontal RMSE
    hdiff = hdiff + hor_error^2;

    % sum of differences used for vertical RMSE
    vdiff = vdiff + vert_error^2;
```

Finally, the total horizontal and vertical RMSE for a single dataset can be computed and stored in *solutions*:

```
% horizontal root mean square error
h_RMSE = sqrt(hdiff/epoch_index);
solutions.h_RMSE = h_RMSE;

% vertical root mean square error
v_RMSE = sqrt(vdiff/epoch_index);
solutions.v_RMSE = v_RMSE;
```

3.2.5 Computation of Time to Reach 5cm Horizontal and Vertical Error

When determining the time that it takes a solution to reach 5cm of error, our first approach was to determine the last epoch at which the solution rose over 5cm of error, and then use the following epoch. However, some noisy datasets spike to 5cm at random times near the end of the dataset and this approach gave inaccurate results.

Our second approach was to determine the first epoch at which the error was under 5cm. For horizontal error, this was completed by first initializing the 5cm error epoch to zero and then looping through all horizontal errors to find the desired epoch. However, if the solution never reached 5cm accuracy, the 5cm epoch was stored as NaN. This process can be seen in the following code:

```
time to reach 5cm horizontal error
solutions.h_error_5cm = 0;
terminate = 1;
k = 1;
while terminate == 1
    if solutions.hor_error(k) <= 0.05
         solutions.h_error_5cm = solutions.decimalHour(k);
         terminate = -1;
    elseif k == length(solutions.hor_error)
         terminate = -1;
         solutions.h_error_5cm = NaN;
    else
        k = k+1;
    end
end
```

For vertical error, the process was identical:

```
% time to reach 5cm vertical error
solutions.v_error_5cm = 0;
terminate = 1;
k = 1;
while terminate == 1
    if solutions.vert_error(k) <= 0.05
        solutions.v_error_5cm = solutions.decimalHour(k);
        terminate = -1;
    elseif k == length(solutions.vert_error)
        terminate = -1;
        solutions.v_error_5cm = NaN;
    end
    k = k+1;
end
```

3.3 Number of Satellites and GDOP as a Function of Time

For each RINEX file, the GDOP and number of satellites were plotted against time using the *GDOP, num_sat,* and *decimalHour* matrices stored in the *solutions* data structure:

```
% ********** GDOP and Number of Satellites ******
figure('Position', [50 50 1000 600])
```

```
title(sprintf('GDOP and Number of Satellites vs Time for %s', path_file),
'Interpreter', 'none');
yyaxis right
plot(solutions.decimalHour, solutions.GDOP);
ylabel('GDOP'); ylim([0 5])
yyaxis left
plot(solutions.decimalHour, solutions.num_sat);
ylabel('Number of Satellites'); ylim([0 25])
grid on; xlabel('Time (Hours)')
```

Plots for a 24-hour and 30-minute data arc of ALGO00CAN can be seen in figures 2 and 3.

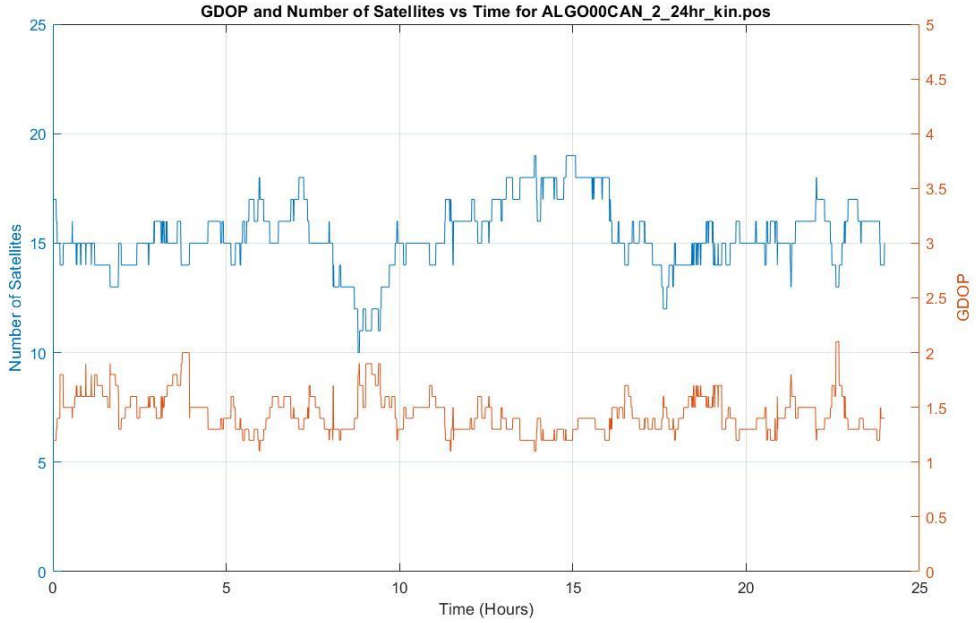

Figure 2 - GDOP and Number of Satellites vs Time for a 24-Hour Data Arc of ALGO00CAN

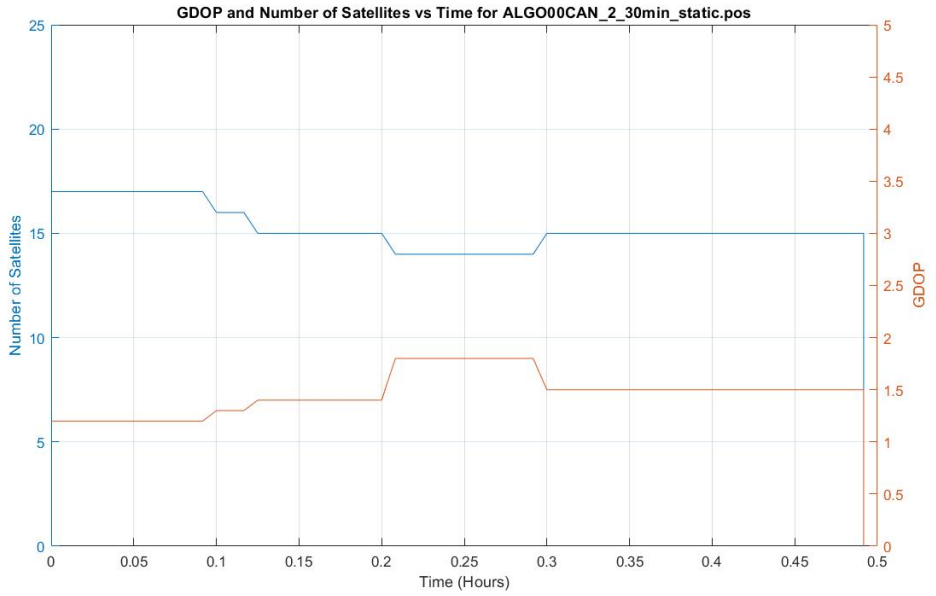

Figure 3 - GDOP and Number of Satellites vs Time for a 30-Minute Data Arc of ALGO00CAN

This inverse relationship between the number of satellites and GDOP can be seen for all datasets. The relationship is very symmetrical for the 30-minute data arc, however the noise in the 24-hour data arc may be attributed to other factors including the spread of the available satellites. Similar noise can be seen in Figure 3 below.

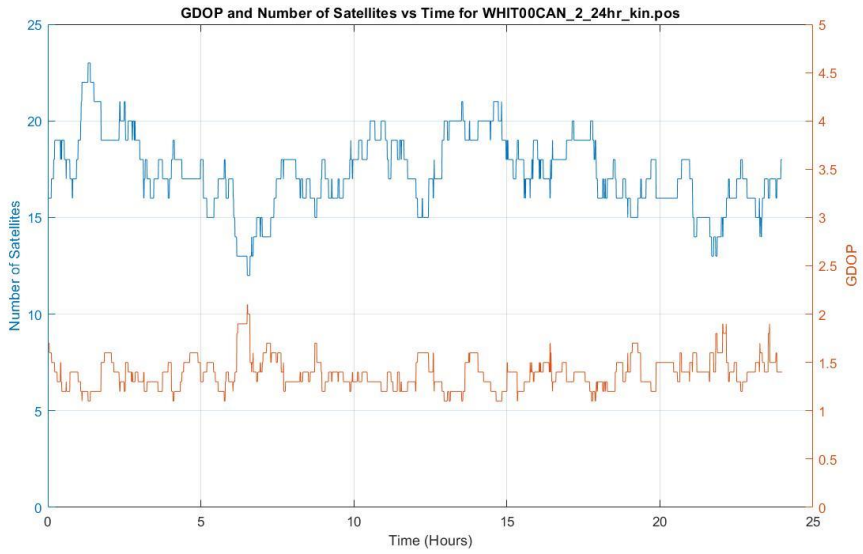

Figure 4 - GDOP and Number of Satellites vs Time for a 24-Hour Data Arc of WHIT00CAN

3.4 Data Processing for Multiple Files

3.4.1 Combing Datasets for Plotting / Analysis

In order to analyze relationships between various datasets, a function called *main.m* was developed. All RINEX files are first organized by station, data arc length, processing mode, and day of acquisition. These combinations allowed us to compute vectors of all horizontal and vertical RMSE or time to reach 5cm of error for all 24-hour data, 30-minute data, kinematic data, and static data. Complete code can be found in Appendix 2.0, but for example, the following example shows how horizontal and vertical RMSE vectors are computed for all 24 sets of 24-hour data:

```
% ****** horizontal and vertical RMSE for 24 hour data arc ***********
% j counts through the stations numbers (1 to 6) to be used in readPos.m
% i iterates through each element of the 24H vector
h_RMSE_24hr = zeros(24,1);
v_RMSE_24hr = zeros(24,1);
j = 1;
for i = 1:24
    solutions = readPos(sprintf('%s', station24H{i}), j);
    h_RMSE_24hr(i) = solutions.h_RMSE;
    v_RMSE_24hr(i) = solutions.v_RMSE;
    j = j+1;
    if j > 6
        j = 1;
    end
end
```

Next, all RINEX files were organized into the following eight categories:

Day 2				Day 3			
24 Hour		30 Minute		24 Hour		30 Minute	
Kinematic	Static	Kinematic	Static	Kinematic	Static	Kinematic	Static

Table 1: RINEX File Categories

This categorization allowed us to develop overlapping histograms comparing RMSE and time to reach 5cm of error for specific cases, such as horizontal RMSE of kinematic data vs static data, or time to reach 5cm or vertical error for a 24-hour data arc vs a 30-minute data arc. Tables with results of vertical RMSE, horizontal RMSE, time to reach 5cm vertical error, and time to reach 5cm horizontal error for each category can be seen in Appendix 5.0.

Additionally, this organization allowed us to plot time series of the evolution of horizontal and vertical error for very specific scenarios. For example, the following portion of *main.m* computes plots of vertical and horizontal error as a function of time for all 24-hour static data on Day 2:

```
24H STATIC DAY 2
all_hor_error24S2 = [];
 all_vert_error24S2 = [];
 hour = [];
```

```
 length = [];
 sizes = [];

for i=1:6
    solutions = readPos(sprintf('%s', station24S2{i}), i);
    hor_error = solutions.hor_error;
    vert_error = solutions.vert_error;
    hour = solutions.decimalHour;

    if i==5
        hor_error(numel(hor_error)+1) = 0;
        hour(numel(hour)+1) = hour(numel(hour));
    end
    if i==5
        vert_error(numel(vert_error)+1) = 0;
    end

    length = numel(hor_error);
    sizes = [sizes, length];

    all_hor_error24S2 = [all_hor_error24S2, hor_error];
    all_vert_error24S2 = [all_vert_error24S2, vert_error];

    subplot(2,1,1)
    plot(hour, all_hor_error24S2(:,i));
    hold on

    subplot(2,1,2)
    plot(hour, all_vert_error24S2(:,i));
    hold on
end
subplot(2,1,1)
title('24 Hour STATIC Day 2 Horizontal Error');
xlabel('Hours');
ylabel('Horizontal Error (m)');
legend('ALGO00CAN','DUBO00CAN','WHIT00CAN','GOLD00USA','WUHN00CHN','IISC00IND')

subplot(2,1,2)
title('24 Hour STATIC Day 2 Vertical Error');
xlabel('Hours');
ylabel('Vertical Error (m)');
legend('ALGO00CAN','DUBO00CAN','WHIT00CAN','GOLD00USA','WUHN00CHN','IISC00IND')
```

3.5 Effect of Processing Mode on RMSE and Time to Reach 5cm Error

To analyze the effect of the processing mode on RMSE and time to reach 5cm of error, histograms were created using categories described in section 3.4.1, however data for day 2 and day 3 was combined.

The following four histograms represents the relationship between static and kinematic processing modes on vertical and horizontal RMSE, and on time to reach 5cm of vertical and horizontal error.

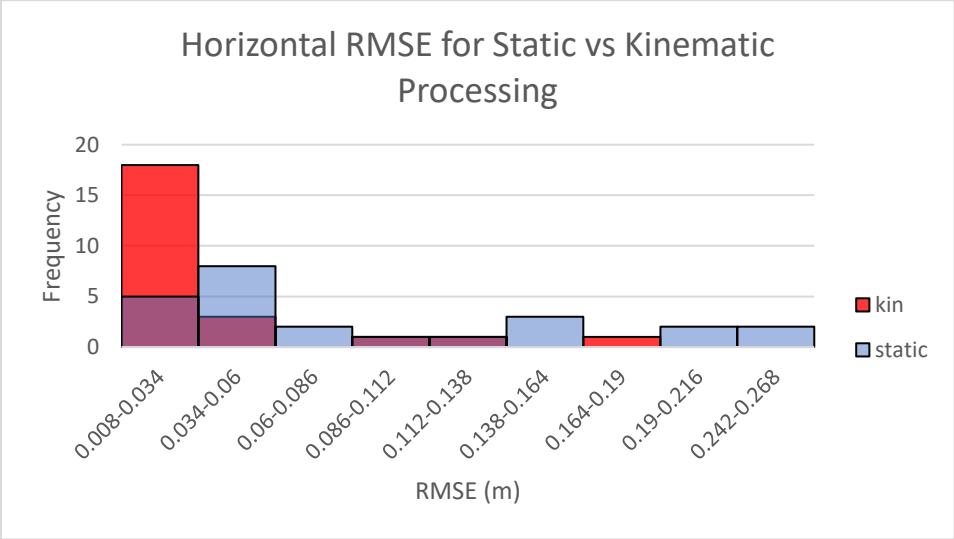

Figure 5 - Horizontal RMSE Histogram for Static vs Kinematic Processing

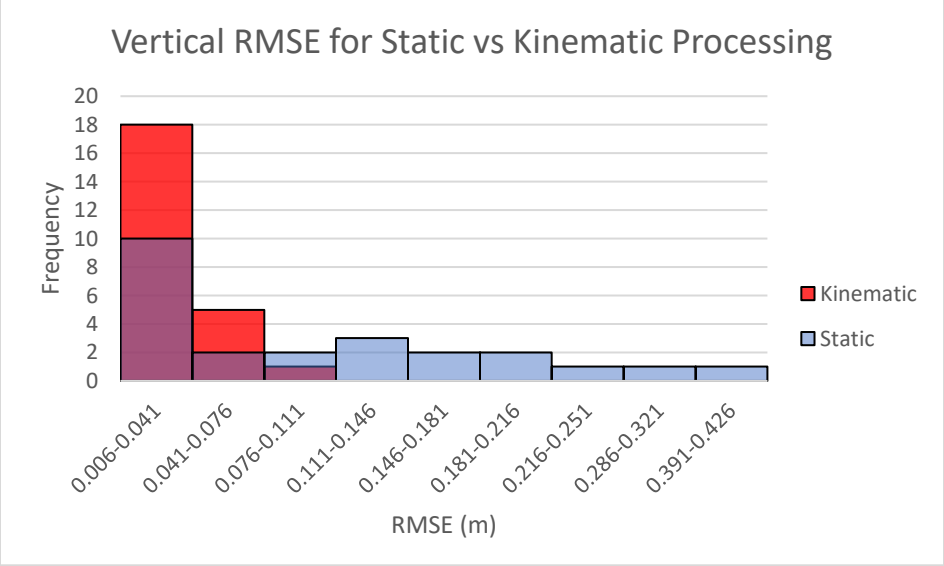

Figure 6 - Vertical RMSE Histogram for Kinematic vs Static Minute Data

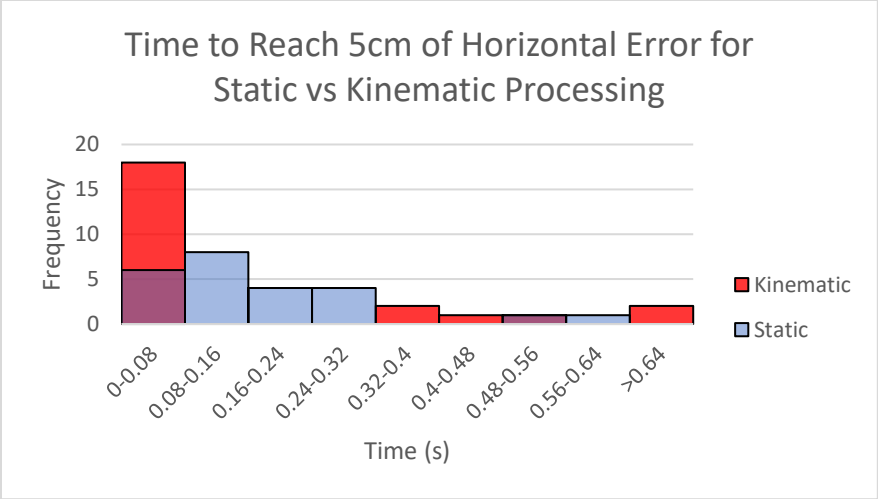

Figure 7 – Histogram of Time to Reach 5cm of Horizontal Error for Static vs Kinematic Processing

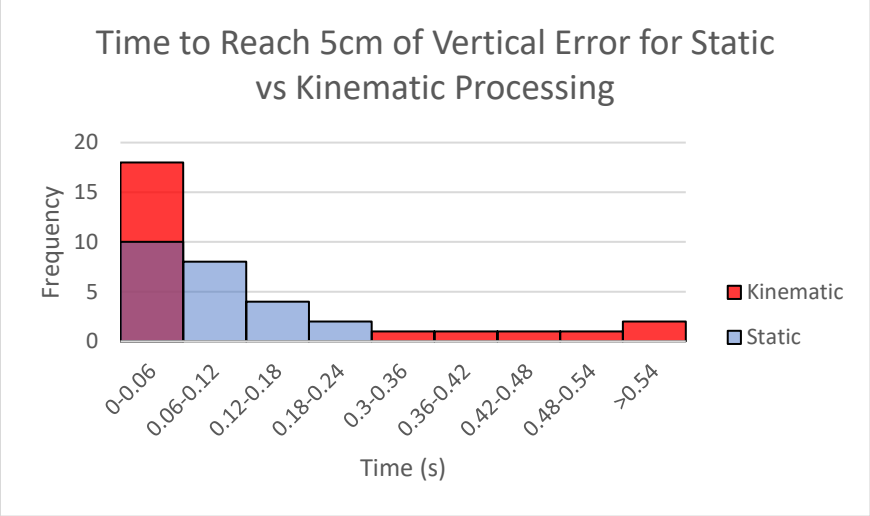

Figure 8 – Histogram of Time to Reach 5cm of Vertical Error for Static vs Kinematic Processing

Based figures 5 to 8, it can be concluded that kinematic processing generally results in a lower RMSE and reaches 5cm of accuracy faster. This result was unexpected as static processing treats a receiver as if it is stationary, however the results could be attributed to the fact that a greater number of satellites are used to compute position in kinematic mode, or that kinematic processing is a more fluid approach to estimating a static position. Therefore, based on these results, kinematic processing provides more accurate results.

3.6 Effect of Data Arc on RMSE and Time to Reach 5cm Error

To analyze the effect of data arc length on RMSE and time to reach 5cm of error, histograms were created using again using categories described in section 3.4.1.

The following four histograms represents the relationship between 24-hour and 30-minute data arcs on vertical and horizontal RMSE, and on time to reach 5cm of vertical and horizontal error.

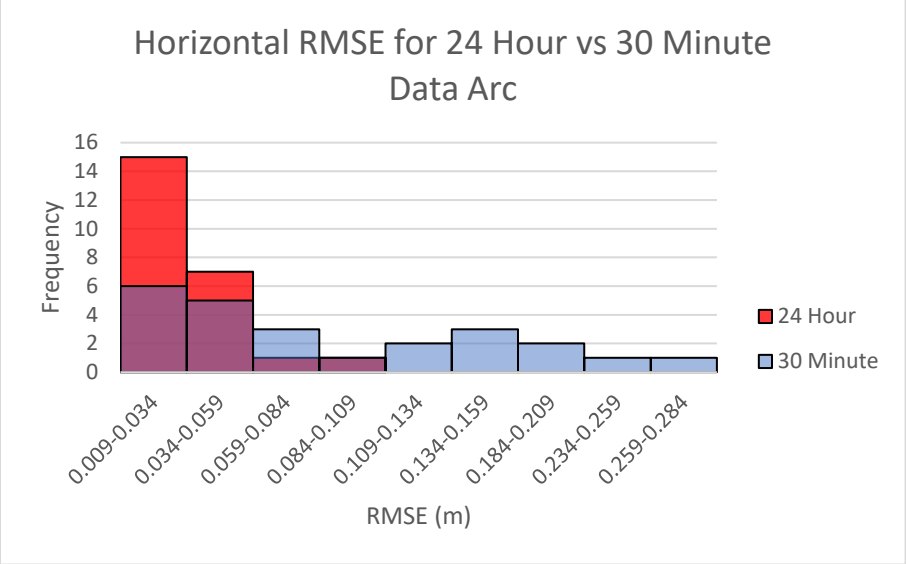

Figure 9 - Horizontal RMSE Histogram for 24 Hour vs 30 Minute Data Arcs

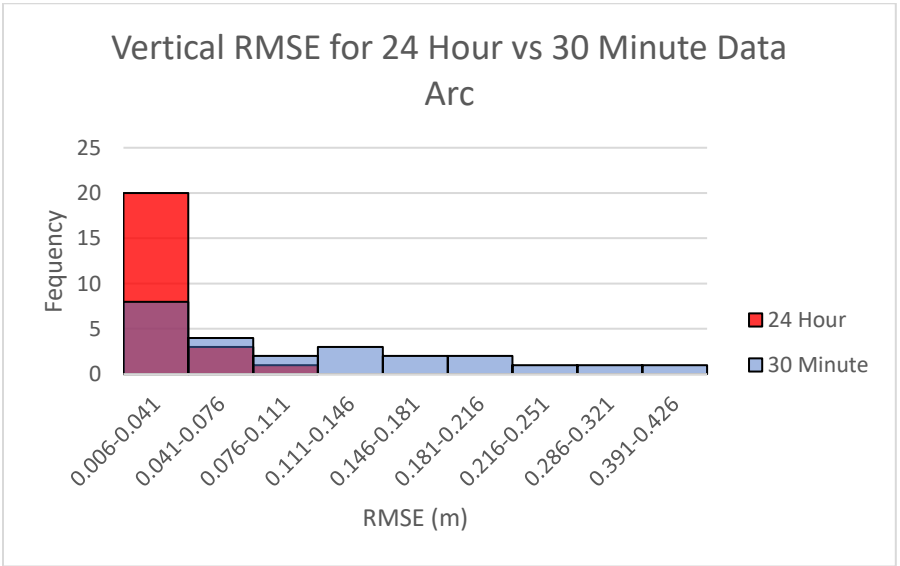

Figure 10 - Vertical RMSE for 24 Hour vs 30 Minute Data Arcs

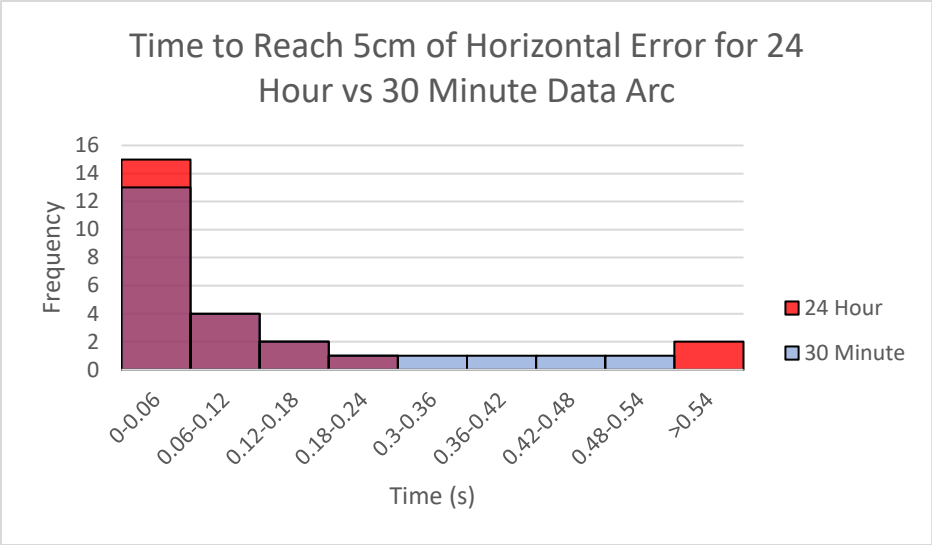

Figure 11 - Histogram of Time to Reach 5cm of Horizontal Error for 24 Hour vs 30 Minute Data Arcs

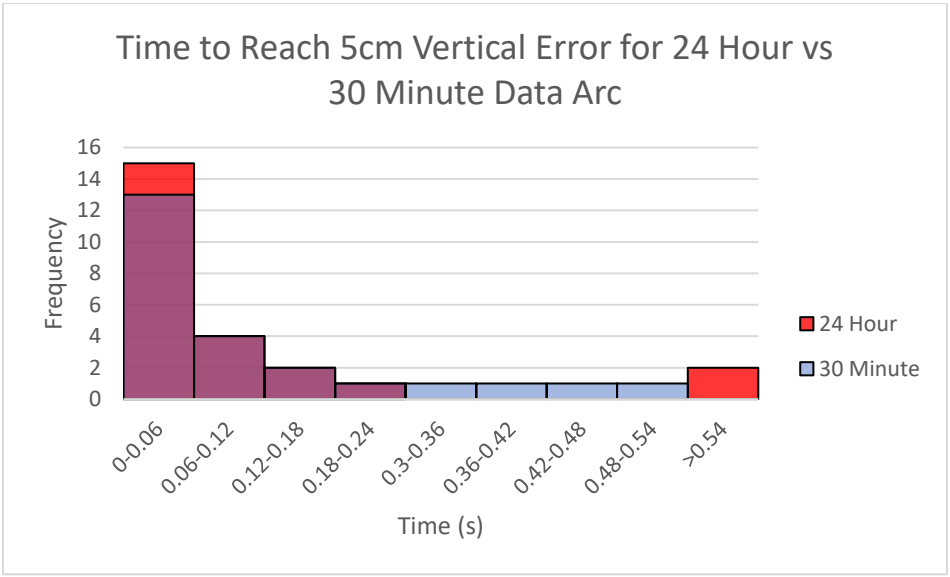

Figure 12 - Histogram of Time to Reach 5cm of Vertical Error for 24 Hour vs 30 Minute Data Arcs

Based on figures 9 to 12, it can be noted that data collected over a 24-hour period clearly reduces the total RMSE in both horizontal and vertical directions. However, when observing the effect on time to reach 5cm of error, there doesn't appear to be a difference. Since most solutions reach this accuracy within the first 30 minutes, this result is expected. Therefore, based on the above results, a longer data arc results in greater accuracy of the position estimate, while it has little impact on the time to obtain 5cm of accuracy.

3.7 Effect of Processing Mode and Data Arc Length on the Evolution of Horizontal and Vertical Errors Over Time

We created 8 charts corresponding to our 8 categories in [Table 1]. Each chart contains 2 plots, for both horizontal and vertical errors plotted against their data arc. The data arc length for 24 hour static charts are shorter because after the 2nd hour the error consolidates and becomes mellow. So, focusing on the noisy first couple hours of static will look more appealing as we can picture what happens after the 2nd hour of the static charts.

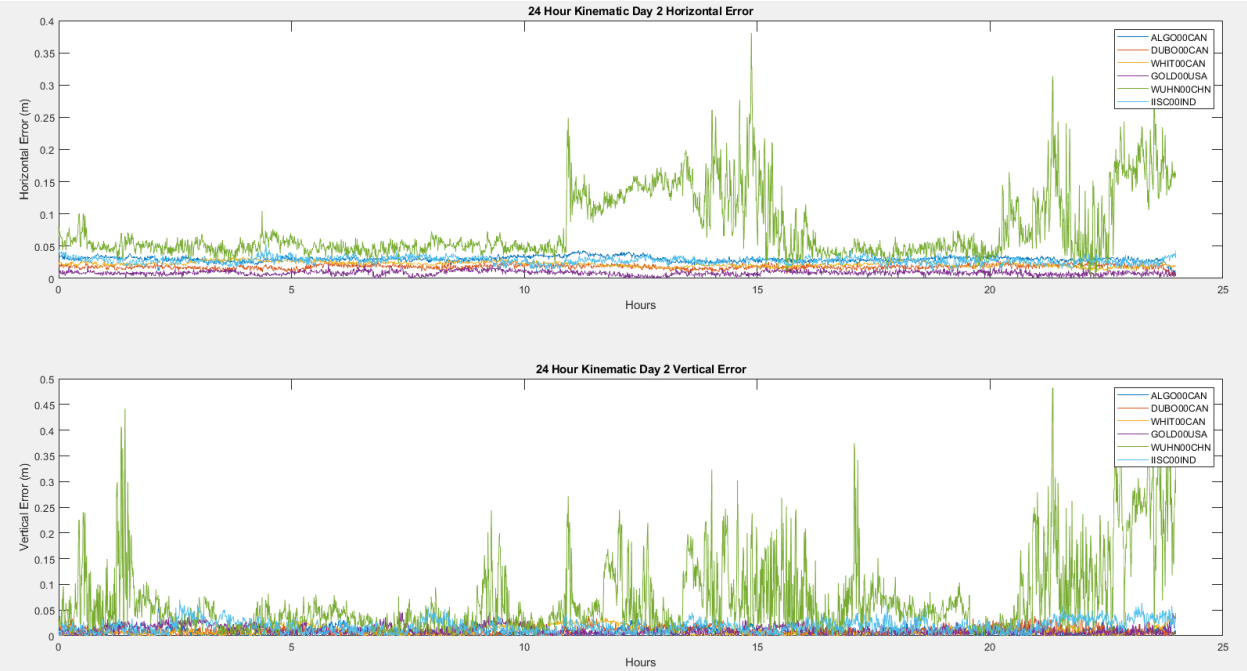

Figure 13 – Chart of 24 Hour Kinematic's Horizontal and Vertical Error for Day 2

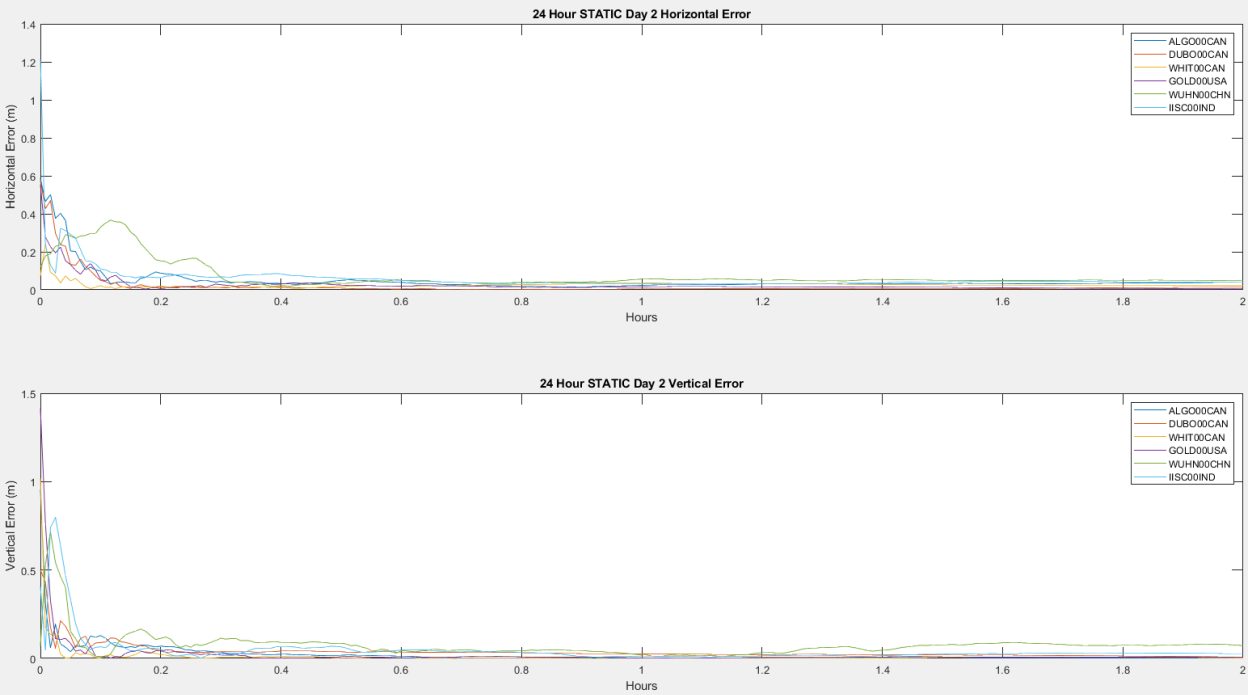

Figure 14 – Chart of 24 Hour Static's Horizontal and Vertical Error for Day 2

The kinematic chart is much noisier than the static chart. WUHN00CHN spikes a lot in the kinematic chart and doesn't seem to consolidate by the end of the 24 hours. In the static chart, WUHN00CHN spikes a bit but still consolidates by the 5th hour.

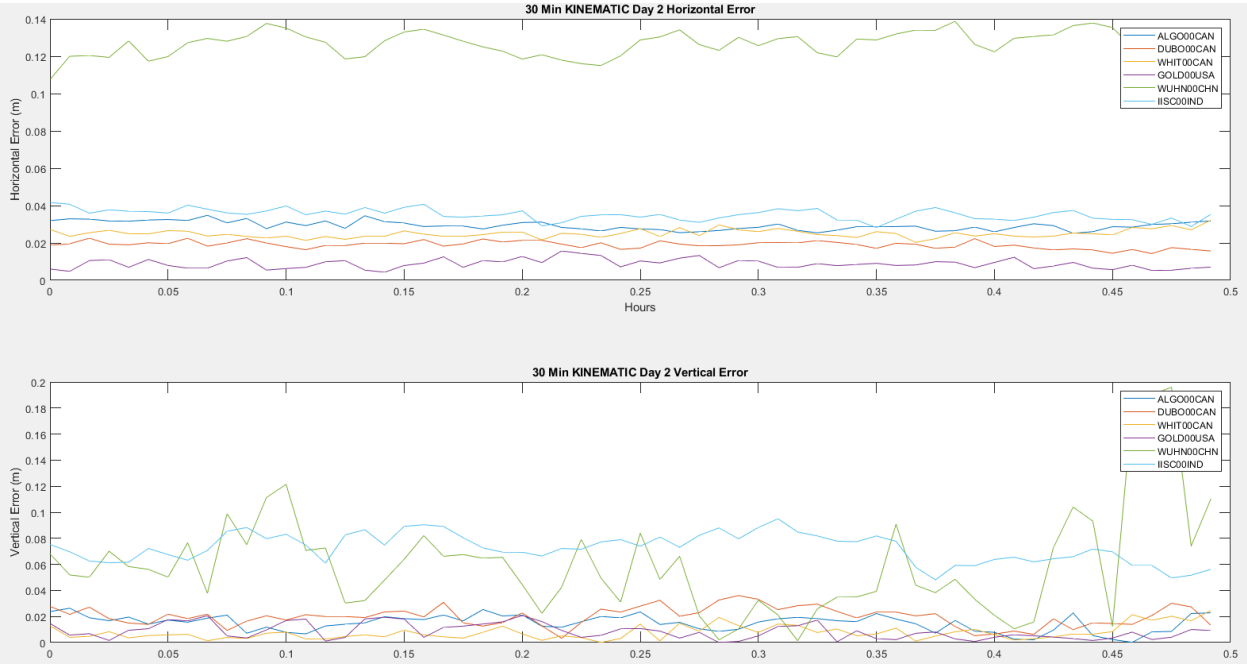

Figure 15– Chart of 30 Minute Kinematic's Horizontal and Vertical Error for Day 2

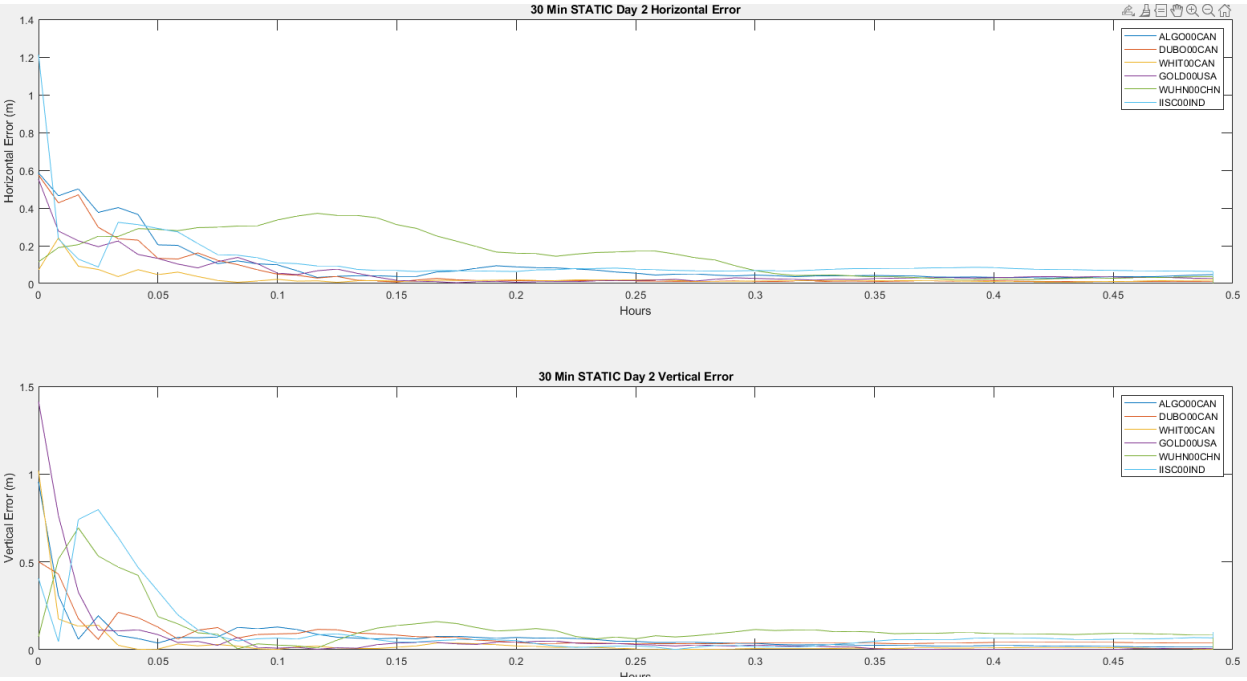

Figure 16 – Chart of 30 Minute Static's Horizontal and Vertical Error for Day 2

The charts for 24 hour and 30 minute for Day 2 are almost identical, just with different data arcs. Again, Kinematic is not consistent as it doesn't consolidate within the 30 min. Static consolidates after at most, 18 minutes (0.3 hours).

Figure 17 – Chart of 24 Hour Kinematic's Horizontal and Vertical Error for Day 3

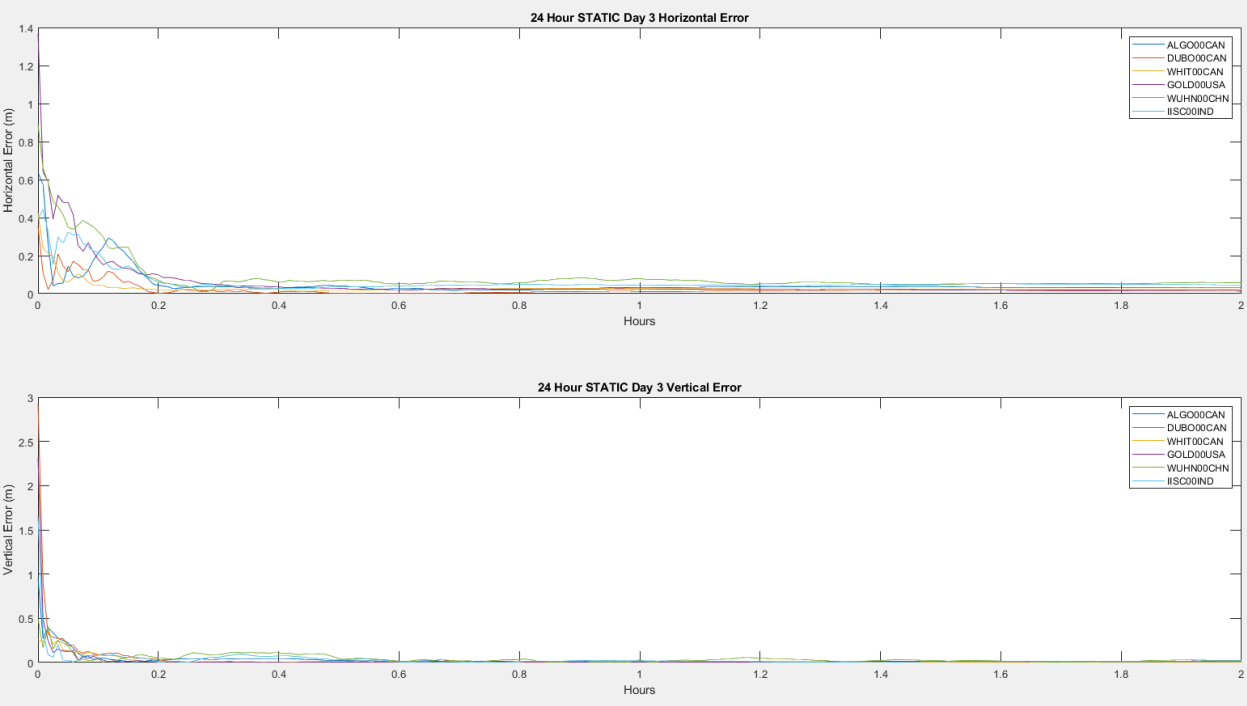

Figure 18 – Chart of 24 Hour Static's Horizontal and Vertical Error for Day 3

The data for the 3rd day is very similar to the 2nd day for both kinematic and static 24 hour scenarios. Kinematic is still seems noisy with WUHN00CHN being the noisiest. The most noticeable difference is that day 3's static chart consolidates earlier than day 2's. It consolidates at around 2 hours vs 5 hours.

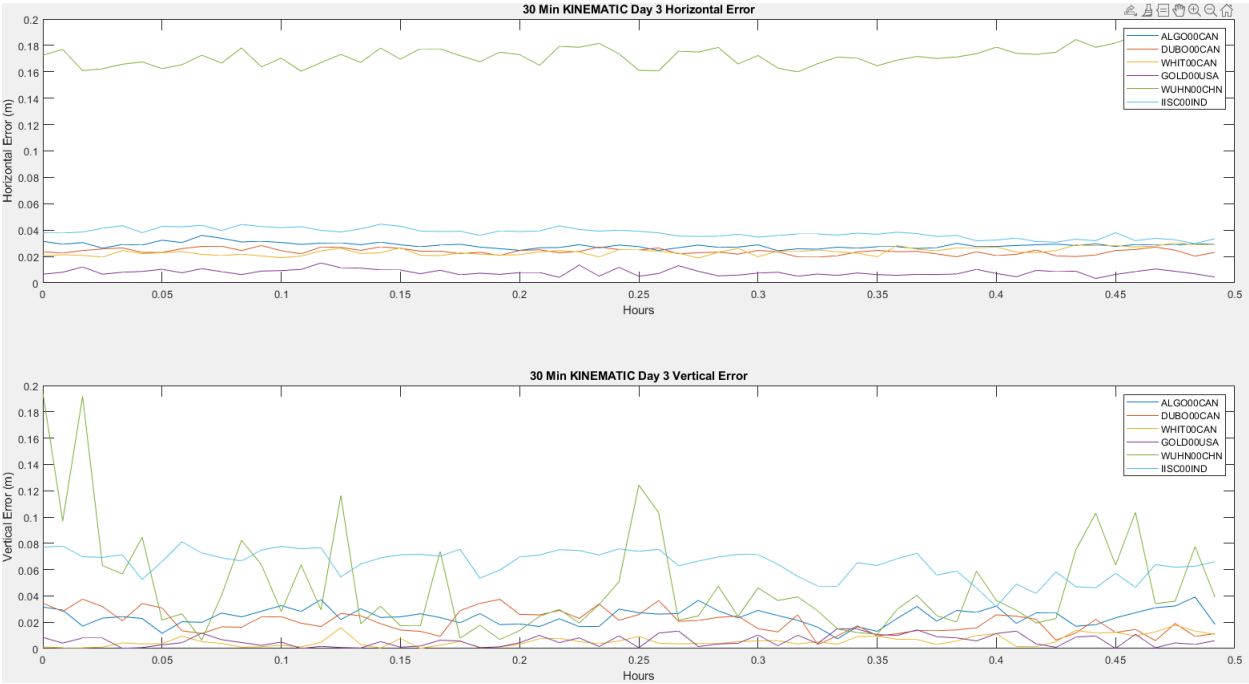

Figure 19 – Chart of 30 Minute Kinematic's Horizontal and Vertical Error for Day 3

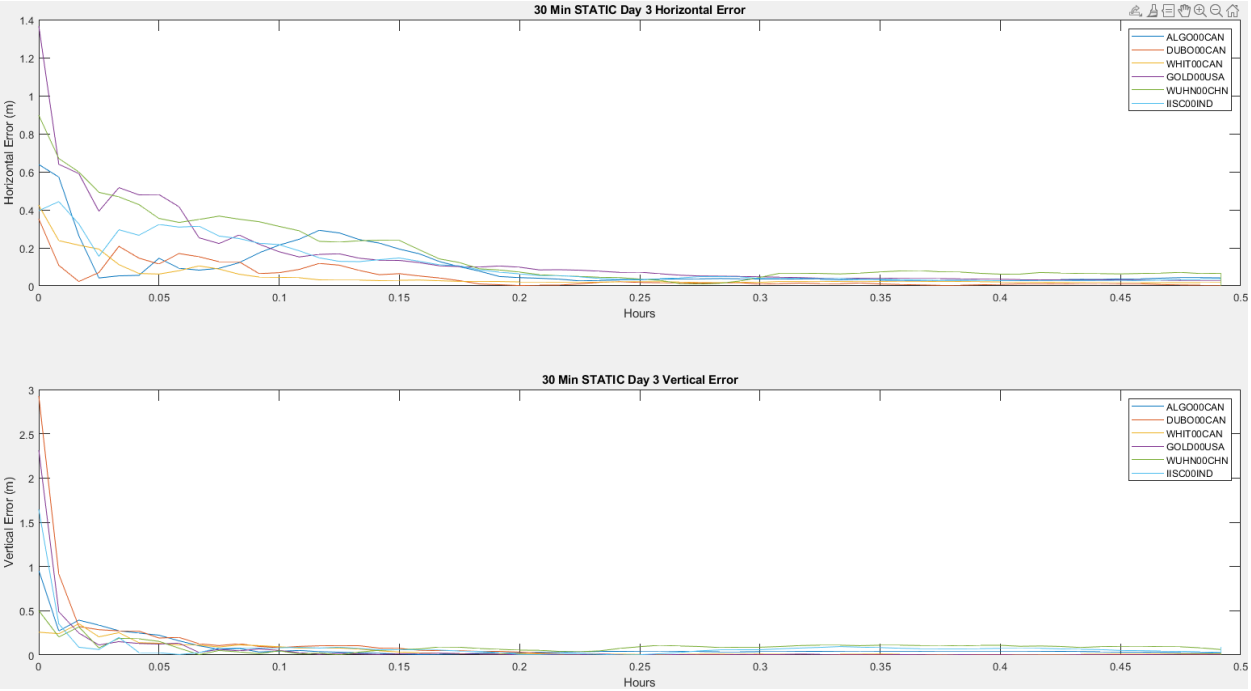

Figure 20 – Chart of 30 Minute Static's Horizontal and Vertical Error for Day 3

The 30 minute chart for day 3's kinematic and static are almost identical to day 2's 30 minute charts for kinematic and static. Day 3's kinematic vertical error seem to be less noisy than day 2's vertical error.

The data for Kinematic and Static modes for 24 hours were to our expectations. We believe that if the stations were considered to be kinematic, its chart would be noisy due to the GNSS filter making estimates every epoch. Static would be less noisy because the GNSS filter would take the average of the estimates of several epochs. Due to expectation of Kinematic being noisy, the results aren't consistent especially with WUHN00CHN. Static is more consistent, we zoomed in on the charts for static because after the first couple of hours, the errors for both horizontal and vertical would mellow out. We also wanted to focus more on the discrepancies during the first couple of hours.

The data for Kinematic and Static modes for 30 minutes were also to our expectations. Comparing to 24 hours, the charts weren't as noisy. Kinematic didn't have huge spikes in error and the static charts consolidated quicker than the 24 hour charts. Through the analysis of kinematic vs static, static produces better results but that **doesn't necessarily mean it's accurate** because the stations do move due to things like plate tectonic shifts and mean sea level changes.

There were no differences in the data comparing day 2 vs day 3. The effect that data arc length has on the results is noise. With a shorter time frame, it's uncommon to find a huge discrepancy during those 30 minutes. Within 24 hours, it is very likely to discover a spike or discrepancy.

4 Notable Error in Data

While processing the data, most WUHN00CHN datasets processed in kinematic mode appeared to be outliers. This can be seen in figures 6, 7, 10, and 11, where two datasets have values greater than the largest bin: WUHN00CHN on day 2 and day 3. The exact results can be seen in the tables provided in Appendix 5.0, however the solutions often did not converge for over 23 hours. For the day 2, 30-minute, kinematic dataset, the solution never reached 5cm of accuracy. This error may be attributed to inaccurate reference coordinates, as they were not readily available.

WUHN00CHN caused us some trouble in developing our time series error data. It was specifically the WUHN00CHN station that didn't have matching matrix dimensions to be plotted on the charts. We had to manually add one extra element to both its data arc, horizontal, and vertical error matrices. For the extra data arc length, the element we added was the last element of the data arc length. For the horizontal and vertical errors, we added a 0 element.

```
if i==5
        hor_error(numel(hor_error)+1) = 0;
        vert_error(numel(vert_error)+1) = 0;
        hour(numel(hour)+1) = hour(numel(hour));
    end
```

5 Conclusions

In conclusion, there were many observed trends. First, in terms of position accuracy, it was observed that the processing mode impacted both the total horizontal and vertical RMSE of a dataset, as well as the time to converge to 5cm accuracy. Kinematic processing generally results in a lower RMSE that static processing and reaches 5cm of accuracy much faster. Secondly, it was observed that a 24-hour data arc dramatically reduces the total RMSE in both horizontal and vertical direction but has no impact on the time to reach 5cm accuracy.

Additionally, when observing the evolution of horizontal and vertical error as a function of time, static produced better results than kinematic. However, static isn't exactly correct itself because static assumes the stations are stationary when there are factors that say that it's not. Kinematic has results that are more truthful to reality, but they're not as precise and accurate in PPP. The longer a data arc is for kinematic scenarios, the more imprecise it becomes. The longer a data arc is for static scenarios, the longer it takes to become precise and will be more accurate.

6 Division of Labour

Tim	Ryan
- Obtained and converted all day 3 files - readPos.m modification - main.m development - XYZ2enu.m development - Data result tables (Appendix 5.0) - Histograms	- Obtained and converted all day 2 files - main.m development - Plots with evolution of error - Map of stations

7 Resources

[1] Latitude,Longitude,Height to/from ECEF (X,Y,Z). (n.d.). Retrieved March 13, 2021, from https://www.oc.nps.edu/oc2902w/coord/llhxyz.htm

[2] Precise Point Positioning (PPP). (n.d.). Retrieved March 20, 2021, from https://novatel.com/an-introduction-to-gnss/chapter-5-resolving-errors/precise-point-positioning-ppp