LASSO

Localize, Attribute SatelliteS in Orbit

P20151

MULTIDISCIPLINARY SENIOR DESIGN

Kate Gleason College of Engineering

TDoA Algorithm

https://twitter.com/tanyaofmars/status/893892863485919232

The TDoA simulator

Agenda
1. Time Difference of Arrival (TDoA) Introduction.
2. The Satellite Problem
3. Finding the Best Solution on a Plane
4. Fitting a Line through the solution on each plane.
5. Unit Tests
6. Simulating the Inputs
7. Deriving the Uncertainty based on Simulated Inputs

Purpose Of Slides

- Inform reader about how the Time Difference of Arrival algorithm works.
- This is not designed as presentation slides. The slides are filled with information and comments. It is meant to be read like a paper (but with lots of pictures).

TDoA Basics

By Anthony Iannuzzi

TDoA data is receiver locations and time differences between each receiver.

$$\boldsymbol{R}_1 = [x_1, y_1, z_1]$$
$$\boldsymbol{R}_2 = [x_2, y_2, z_2]$$
$$\boldsymbol{R}_n = [x_n, y_n, z_n]$$

The receivers must all be measured in the same frame.

TD represents the Time Difference data. It is an upper triangular matrix. Each entry is the time difference between the receivers represented by that row and column number.

$$TD = \begin{array}{c} \\ 1 \\ 2 \\ \vdots \\ n \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & \cdots & n \end{array} \\ \begin{bmatrix} 0 & t_{12} & \cdots & t_{1n} \\ 0 & 0 & \ddots & t_{2n} \\ \vdots & \vdots & \ddots & \ddots \\ 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

The diagonals are zero as a station compared to itself has no time difference. The lower triangle contains redundant information and is set to 0.

TDoA data creates hyperbolas and hyperboloids.

In 2 Dimensions

$$\frac{4(x'')^2}{\delta^2} - \frac{4(y'')^2}{4D^2 - \delta^2} = 1$$

In 3 Dimensions

$$\frac{4(x'')^2}{\delta^2} - \frac{4(y'')^2}{4D^2 - \delta^2} - \frac{4(z'')^2}{4D^2 - \delta^2} = 1$$

where

δ is the distance difference

D is half the distance between stations

δ the distances differences is calculated by

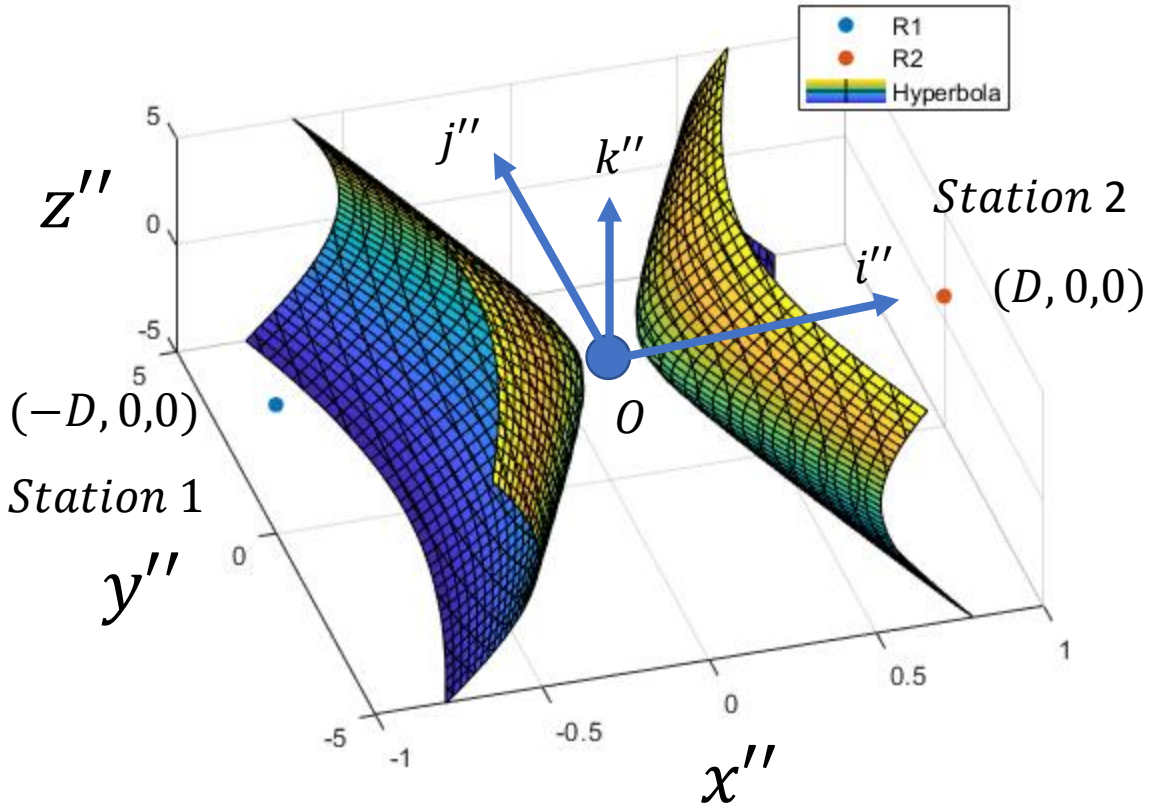
$$\delta = TD * 3e8$$

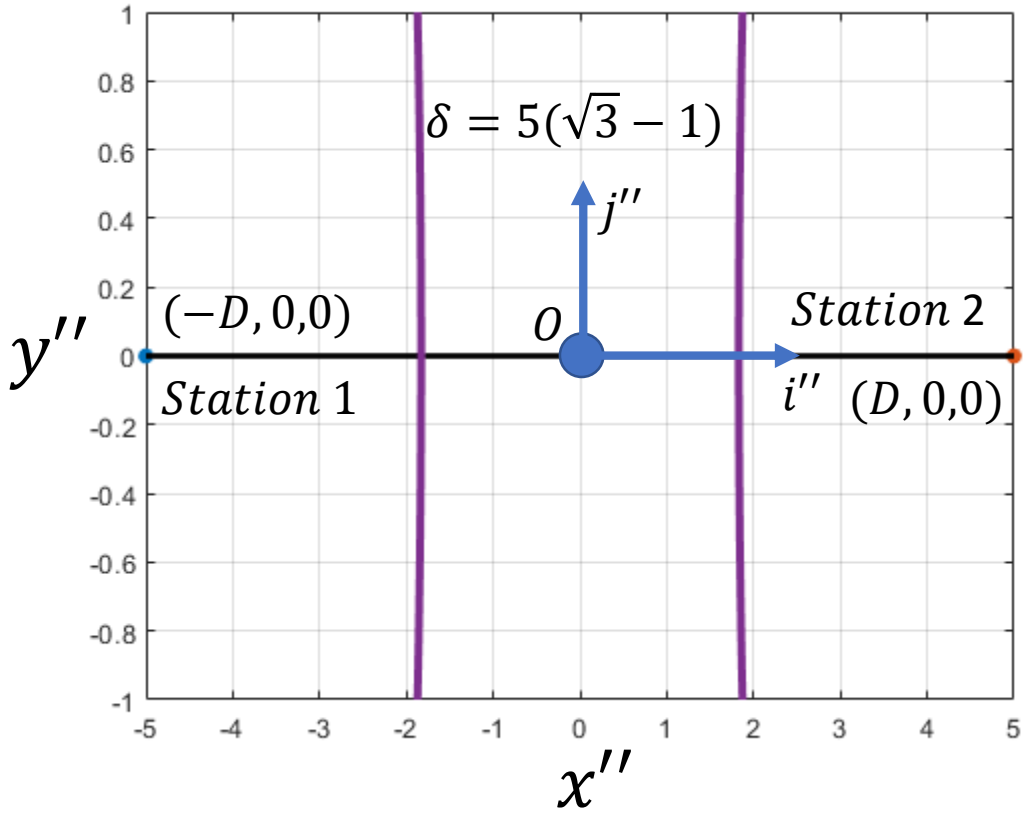
$3e8$ is the speed of light in m/s.

The half distance between stations is

$$D = \frac{||\boldsymbol{R}_2 - \boldsymbol{R}_1||}{2}$$

Graphically what hyperboloid and hyperbola look like.

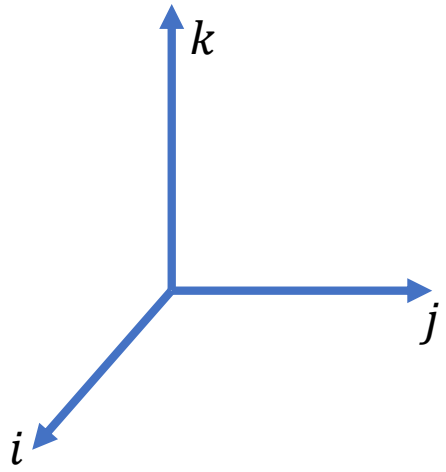


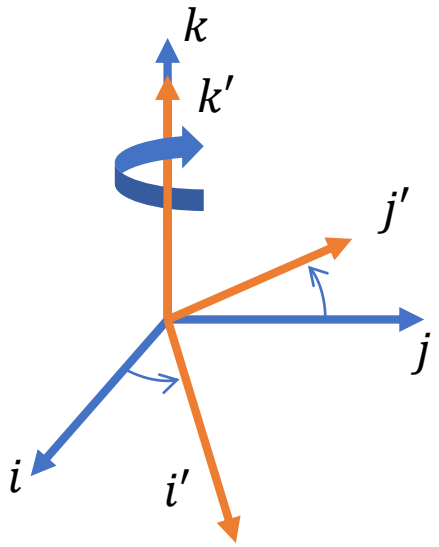


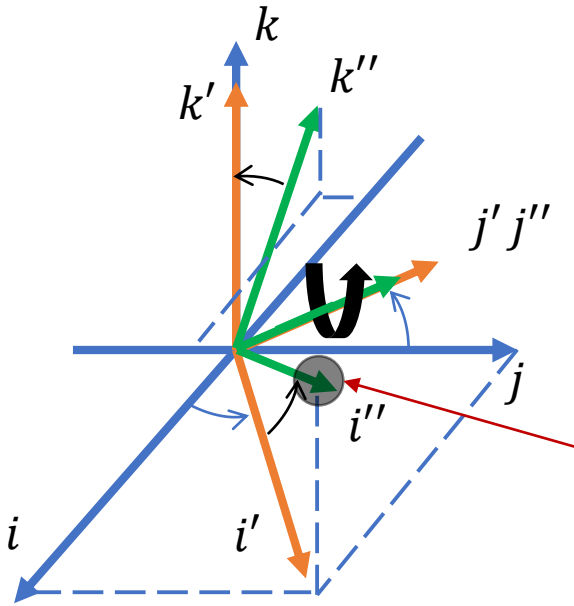
These Hyperbolas assume the stations are located at $(\pm D, 0,0)$. This is called the ***Body Frame***. We will now transform this hyperbola into the ***Fixed Frame***; the frame the stations are measured in.

Coordinate Transformation Introduction

Azimuth, Elevation Rotations







The point we want to transform

Fixed Frame

$$RM = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Intermediate Frame

$$RM_z = \begin{pmatrix} \cos a & -\sin a & 0 \\ \sin a & \cos a & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Body Frame

$$RM_y = \begin{pmatrix} \cos b & 0 & -\sin b \\ 0 & 1 & 0 \\ \sin b & 0 & \cos b \end{pmatrix}$$

A point measured in the body frame, r_b can be converted to the fixed frame, r_f by multiplying by the rotation matrix

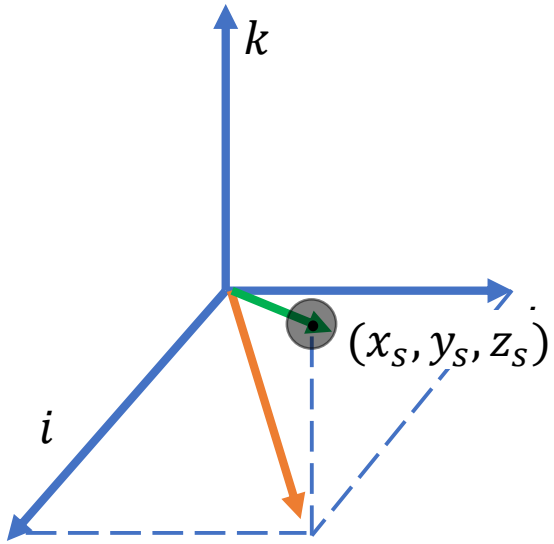
$$RM_{b\to f} = RM_z * RM_y$$

$$\boldsymbol{r}_f = RM_{b\to f} * \boldsymbol{r}_b$$

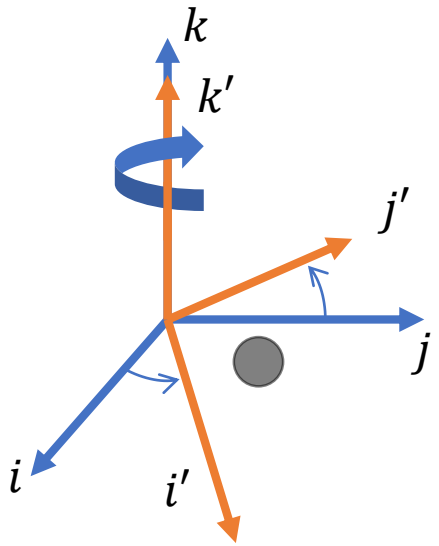
$$RM_{b\to f} = \begin{pmatrix} \cos a \cos b & -\sin a & -\cos a \sin b \\ \sin a \cos b & \cos a & -\sin a \sin b \\ \sin b & 0 & \cos b \end{pmatrix}$$

Calculating Azimuth and Elevation

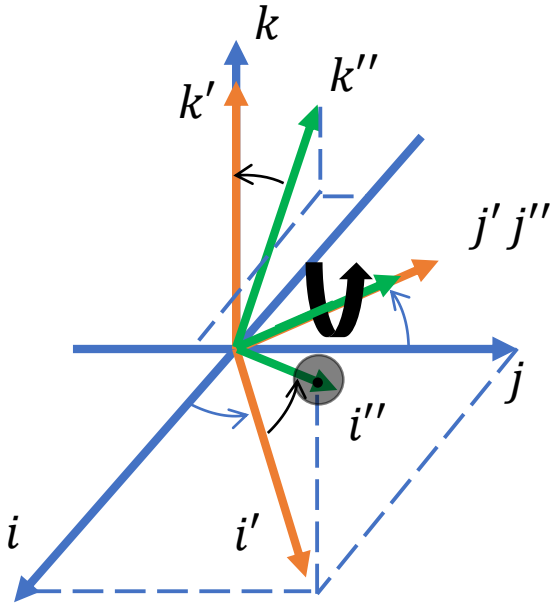
How to calculate Azimuth, Elevation Rotations



Fixed Frame



Intermediate Frame



Body Frame

$Given: [x_s, y_s, z_s]$

Azimuth

$$a = \tan^{-1}\frac{y_s}{x_s}$$

Elevation

$$b = \tan^{-1}\frac{z_s}{\sqrt{x_s^2 + y_s^2}}$$

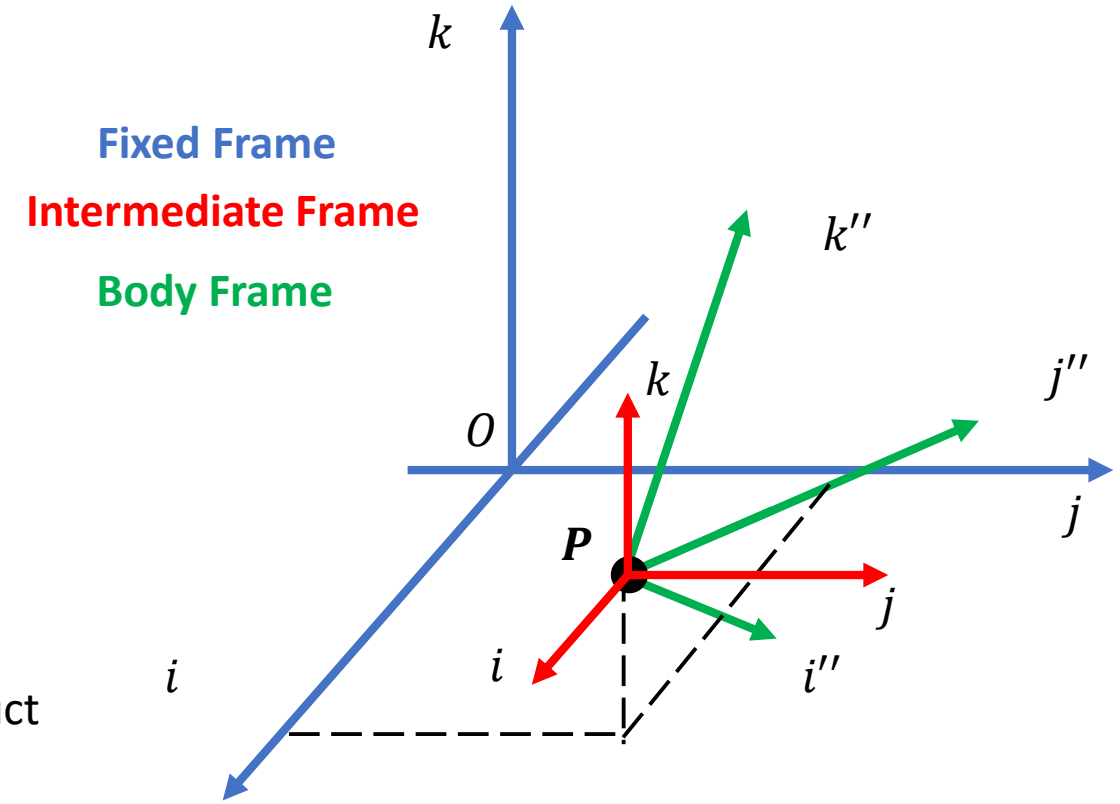
If we have the point measured in the fixed frame, then we can calculate the azimuth and elevation with respect to the fixed frame.

Transforming Coordinate Axes that have different origins.

1. Rotate to the Fixed coordinate frame orientation.
2. Then shift coordinate frame.

$Given$: $\boldsymbol{P} = [x_0, y_0, z_0]$

$$\boldsymbol{r}_f = \underbrace{RM_{b\to f} * \boldsymbol{r}_b^T}_{\boldsymbol{1.}} - \underbrace{\boldsymbol{P^T}}_{\boldsymbol{2.}}$$



Physically, each entry in the RM can be regarded as a dot product between each body and fixed frame axis:

$$RM_{b\to f} = \begin{array}{c|ccc} & i & j & k \\ i'' & \cos a \cos b & -\sin a & -\cos a \sin b \\ j'' & \sin a \cos b & \cos a & -\sin a \sin b \\ k'' & \sin b & 0 & \cos b \end{array}$$

Transforming the Hyperbola or Hyperboloid to the Fixed Frame. Known Data.

Given Absolute Position of Stations:

$$\boldsymbol{R}_1 = [x_1, y_1, z_1]$$

$$\boldsymbol{R}_2 = [x_2, y_2, z_2]$$

$$D = \frac{||\boldsymbol{R}_2 - \boldsymbol{R}_1||}{2}$$

$$\boldsymbol{O} = \boldsymbol{R}_1 + \frac{\boldsymbol{R}_2 - \boldsymbol{R}_1}{2}$$

x axis always points from station 1 to 2.

$$[x_s, y_s, z_s] = \boldsymbol{R}_2 - \boldsymbol{O}$$

Transforming the Hyperbola or Hyperboloid to the Fixed Frame. The new Equation

Let $\boldsymbol{r} = [x - x_0, y - y_0, z - z_0]^T$, then the hyperboloid becomes:

$$\frac{4(\boldsymbol{r} \cdot RM(:,1))^2}{\delta^2} - \frac{4(\boldsymbol{r} \cdot RM(:,2))^2}{4D^2 - \delta^2} - \frac{4(\boldsymbol{r} \cdot RM(:,3))^2}{4D^2 - \delta^2} = 1$$

dot product

Substitute $[x'', y'', z'']$ for

$RM * \boldsymbol{r}^T$ split amongst the terms.

Hyperboloid in Fixed Frame

Let $\boldsymbol{r} = [x - x_0, y - y_0, z - z_0]^T$, then the hyperboloid becomes:

$$\frac{4(\boldsymbol{r} \cdot RM(:,1))^2}{\delta^2} - \frac{4(\boldsymbol{r} \cdot RM(:,2))^2}{4D^2 - \delta^2} - \frac{4(\boldsymbol{r} \cdot RM(:,3))^2}{4D^2 - \delta^2} = 1$$

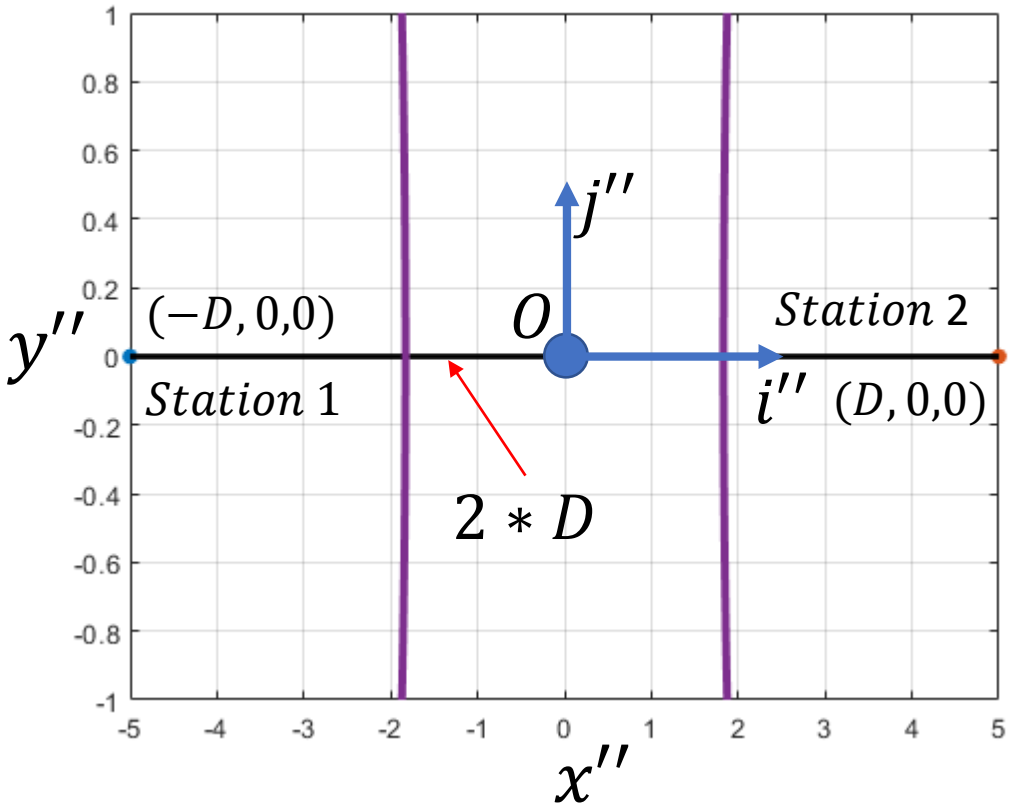
dot product

Substitute $[x'', y'', z'']$ for

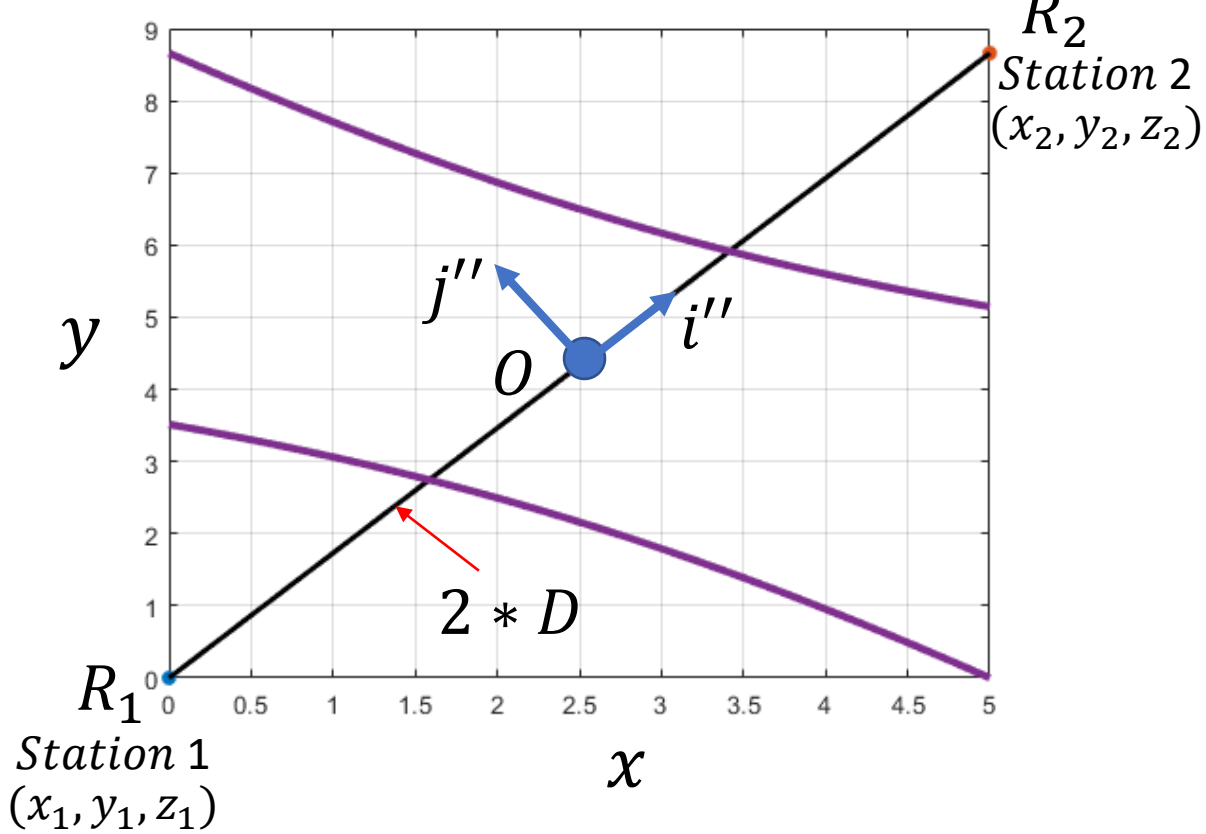
$RM * \boldsymbol{r}^{\boldsymbol{T}}$ split amongst the terms.

Transforming the Hyperbola or Hyperboloid to the Fixed Frame. Visualization

Body Frame



Fixed Frame



Creating the Hyperboloids and Hyperbolas algorithm.

Inputs is *TDoA Data*

- Receiver Locations in the ***Fixed Frame***
- Distance Difference

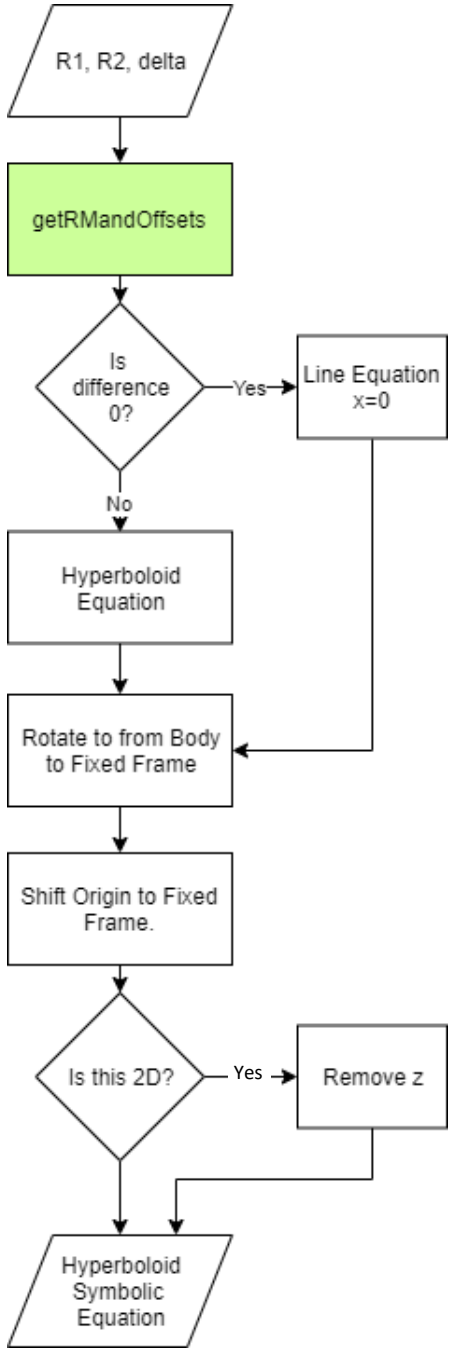
Algorithm:

1. Calculate $\boldsymbol{RM_{b\to f}}$ and offset $\boldsymbol{P}$
2. Create Hyperboloid in ***Body Frame***.
3. Transform to ***Fixed Frame***.

Outputs:

- Symbolic Equation of Hyperboloid

$$\frac{4(\boldsymbol{r}\cdot RM(:,1))^2}{\delta^2}-\frac{4(\boldsymbol{r}\cdot RM(:,2))^2}{4D^2-\delta^2}-\frac{4(\boldsymbol{r}\cdot RM(:,3))^2}{4D^2-\delta^2}=1$$



Changing to a 1 sided Hyperbola

2 sided Hyperbola

$$\frac{4(x'')^2}{\delta^2} - \frac{4(y'')^2}{4D^2 - \delta^2} - \frac{4(z'')^2}{4D^2 - \delta^2} = 1$$

1 sided Hyperbola (solve for x'')

$$x'' = \delta \sqrt{\frac{1}{4} + \frac{(y'')^2}{4D^2 - \delta^2} + \frac{(z'')^2}{4D^2 - \delta^2}}$$

where

δ is the distance difference

D is half the distance between stations

Choose the Hyperbola side based on the sign of the distance difference.

$$|\delta| = 5\left(\sqrt{3} - 1\right) = -3.66$$

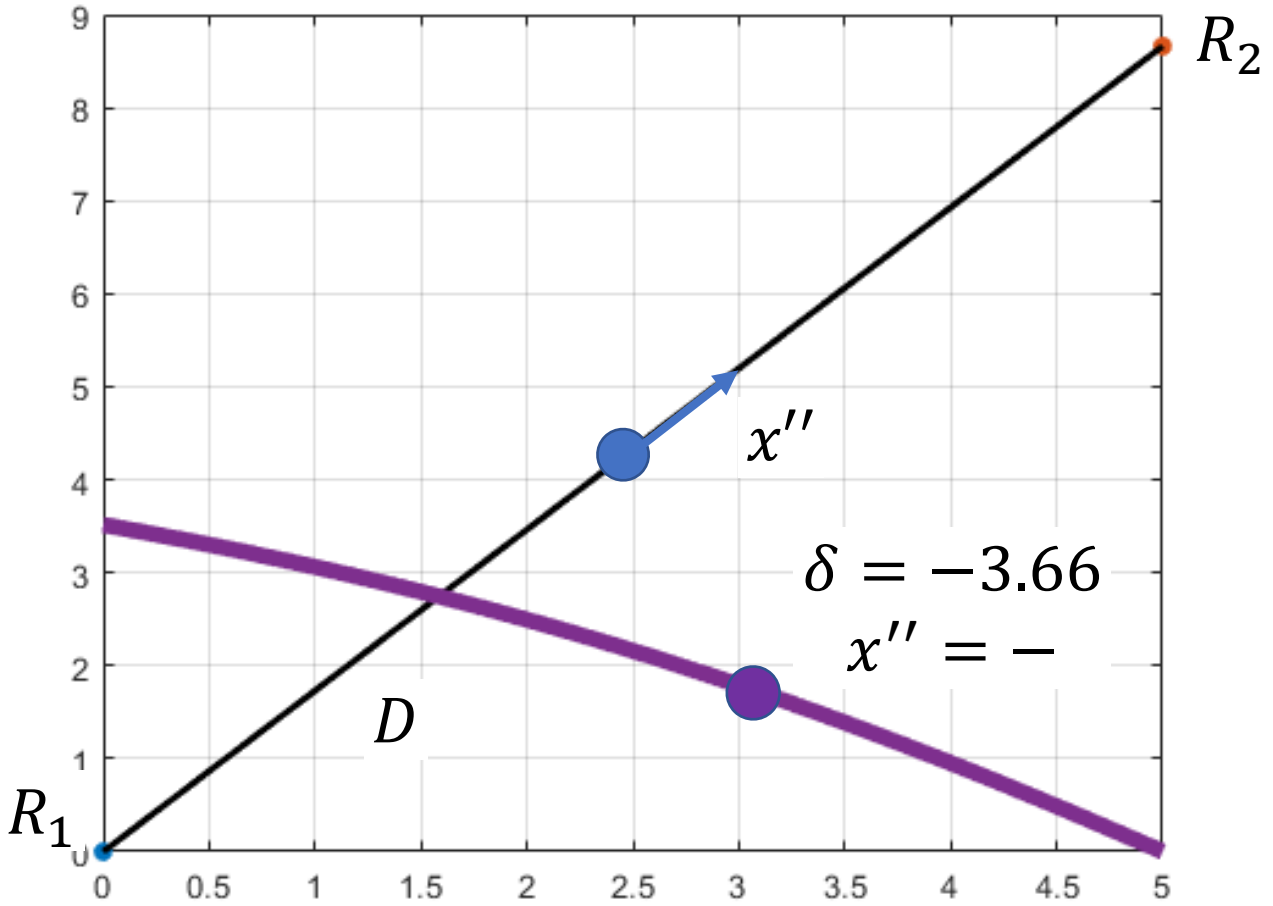
The sign of x'' is wholly dictated by sign of delta.

$$x'' = \delta\sqrt{\frac{1}{4} + \frac{(y'')^2}{4D^2 - \delta^2} + \frac{(z'')^2}{4D^2 - \delta^2}}$$

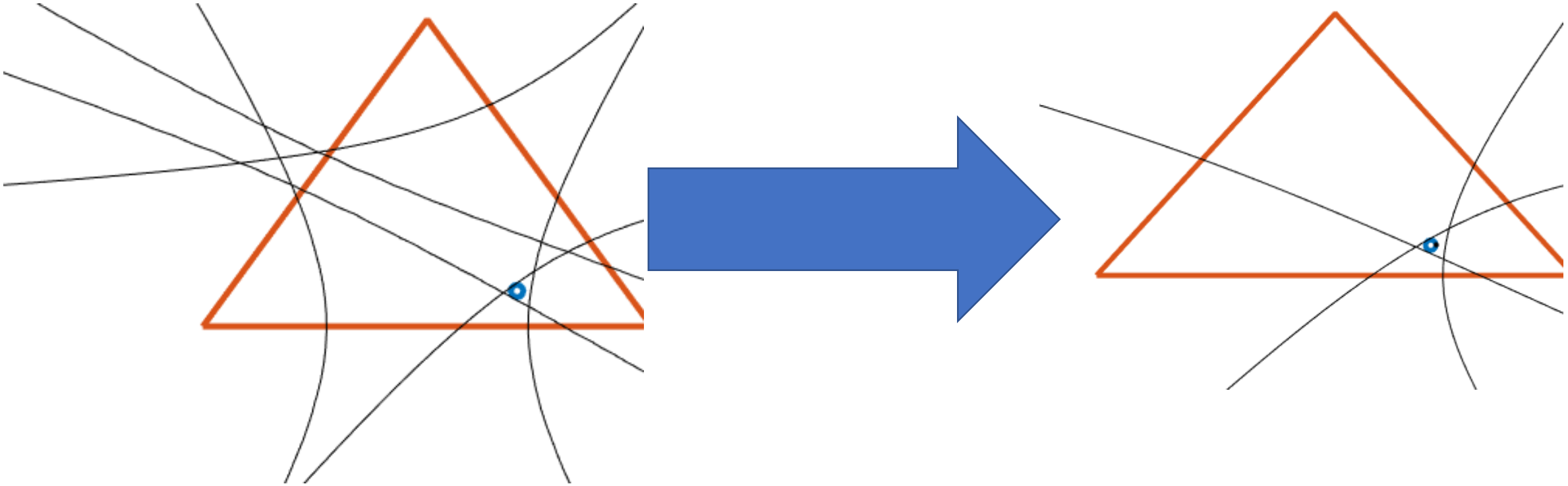
where

δ is the distance difference

D is half the distance between stations



Solution is much less ambiguous with 1 sided Hyperbolas



But how many hyperbolas are created from n number of stations?
How do we find the “best solution”?

TDoA data creates $\frac{n(n-1)(n-2)}{6}$ hyperboloids.

Recall the TDoA Data

$$\boldsymbol{R}_1 = [x_1, y_1, z_1]$$

$$\boldsymbol{R}_2 = [x_2, y_2, z_2]$$

$$\boldsymbol{R}_n = [x_n, y_n, z_n]$$

$$TD = \begin{matrix} \\ 1 \\ 2 \\ \vdots \\ n \end{matrix} \begin{matrix} \begin{matrix} 1 & 2 & \cdots & n \end{matrix} \\ \begin{bmatrix} 0 & t_{12} & \cdots & t_{1n} \\ 0 & 0 & \ddots & t_{2n} \\ \vdots & \vdots & \ddots & \ddots \\ 0 & 0 & 0 & 0 \end{bmatrix} \end{matrix}$$

For $n = 3$, we have 3 Hyperboloids.
For $n = 4$, we have 6 Hyperboloids.

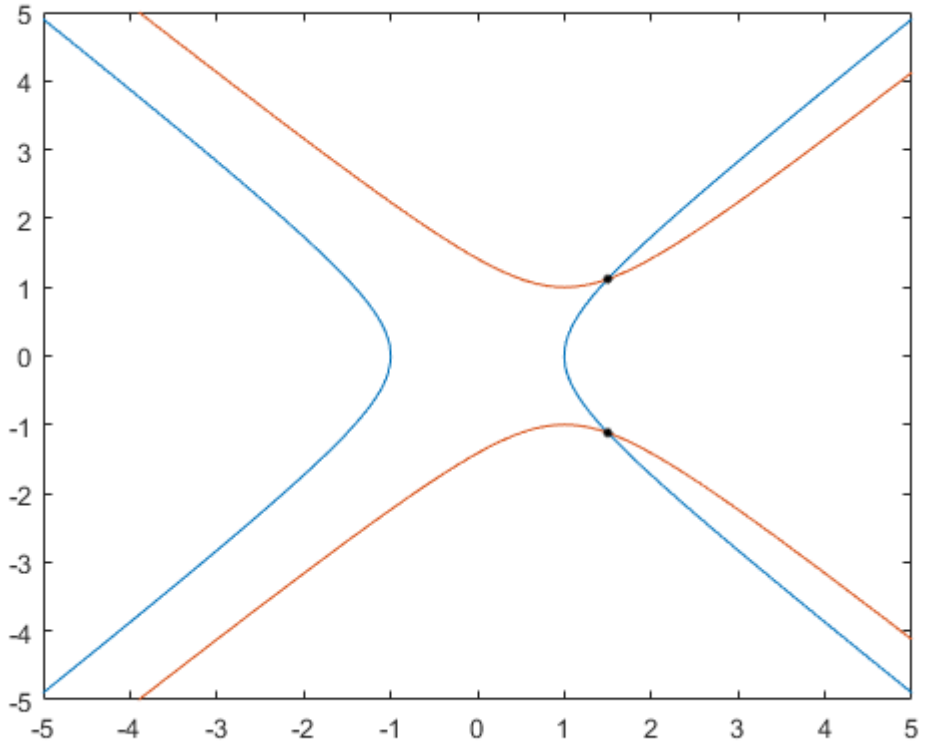
Number of unique Hyperboloids.
Given $n = 3$, we have

$$H_{12}, \mathrm{H}_{13}, \mathrm{H}_{23}$$

Only 2 of these hyperboloids are unique.

For $n = 4$, we have 3 unique Hyperboloids.

Intersecting Hyperbolas in 2D.



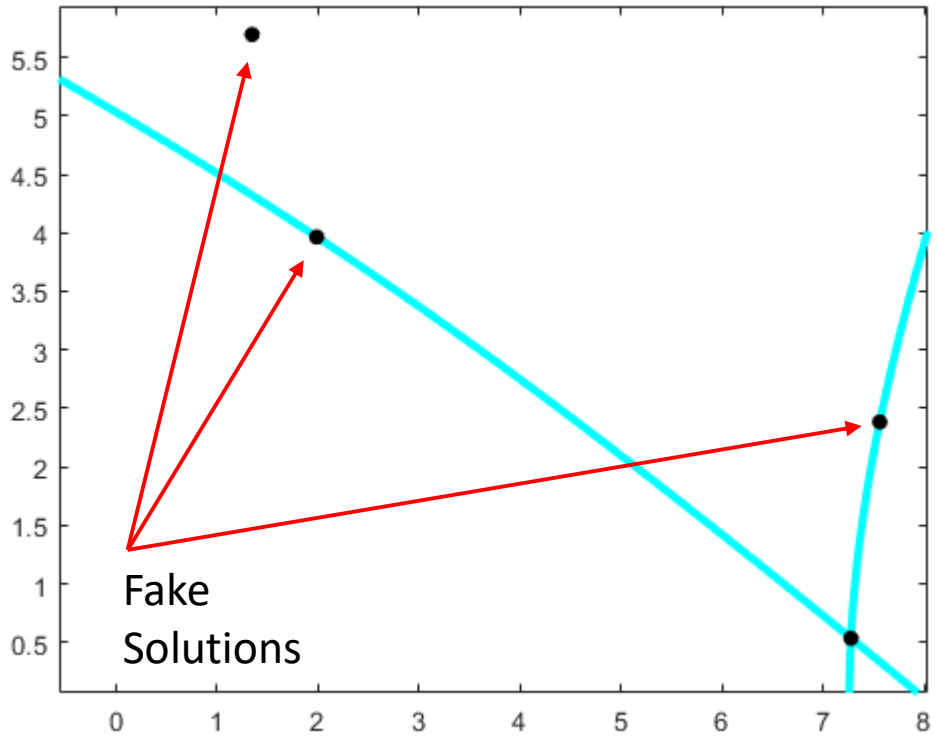
$$H_A = \frac{4(\boldsymbol{r_A} \cdot RM_A(:,1))^2}{\delta_A^2} - \frac{4(\boldsymbol{r_A} \cdot RM_A(:,2))^2}{4D_A^2 - \delta_A^2} = 1$$

$$H_B = \frac{4(\boldsymbol{r_B} \cdot RM_B(:,1))^2}{\delta_B^2} - \frac{4(\boldsymbol{r_B} \cdot RM_B(:,2))^2}{4D_B^2 - \delta_B^2} = 1$$

Intersection is $H_A - H_B = 0$

- This works for both one-sided and two-sided Hyperbolas.
- For one-sided Hyperbolas, must remove fake solutions by plugging in all solutions back into H_A and H_B to verify they are indeed on the Hyperbola.

Removing the Fake Solutions for One-sided Hyperbolas.



$$\boldsymbol{r_A} \cdot RM_A(:,1) = \delta_A \sqrt{\frac{1}{4} + \frac{(\boldsymbol{r_A} \cdot RM_A(:,2))^2}{4D_A^2 - \delta_A^2}}$$

$$\boldsymbol{r_B} \cdot RM_B(:,1) = \delta_B \sqrt{\frac{1}{4} + \frac{(\boldsymbol{r_B} \cdot RM_B(:,2))^2}{4D_B^2 - \delta_B^2}}$$

Intersection is $H_A - H_B = 0$

- For one-sided Hyperbolas, must remove fake solutions by plugging in all solutions back into H_A and H_B to verify they are indeed on the Hyperbola.
- Fake solutions come from squaring both sides when going to solve the one-sided equation.

The one-sided method breaks down when the sign of the Time Difference is questionable.

- The one-sided method requires signed time differences.
- Consider the following time differences:

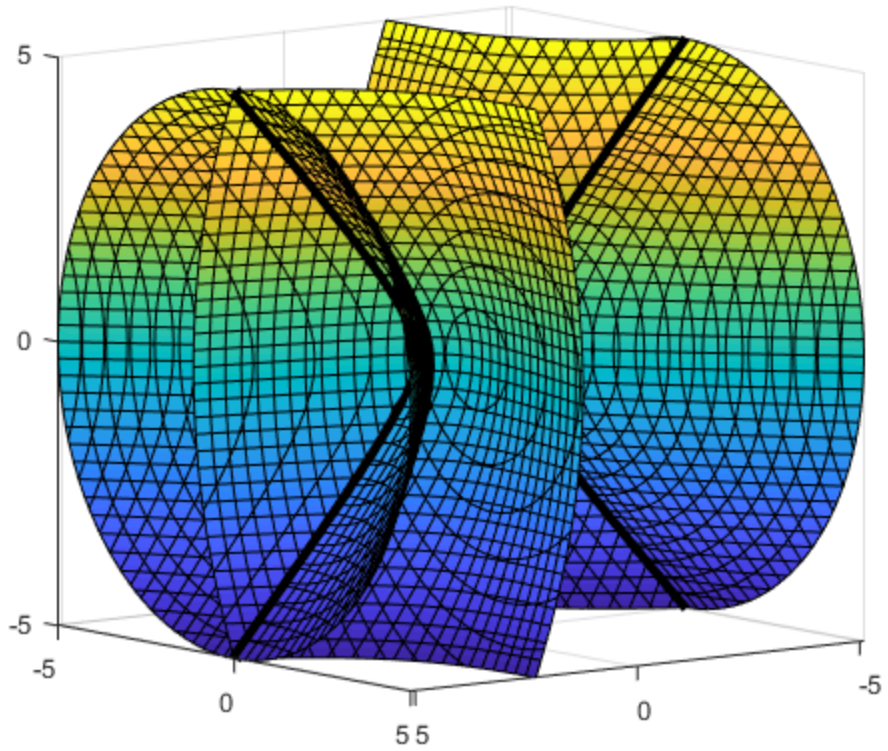
$$\text{(A)}\ \ TD = 1.2us \pm 0.1us$$

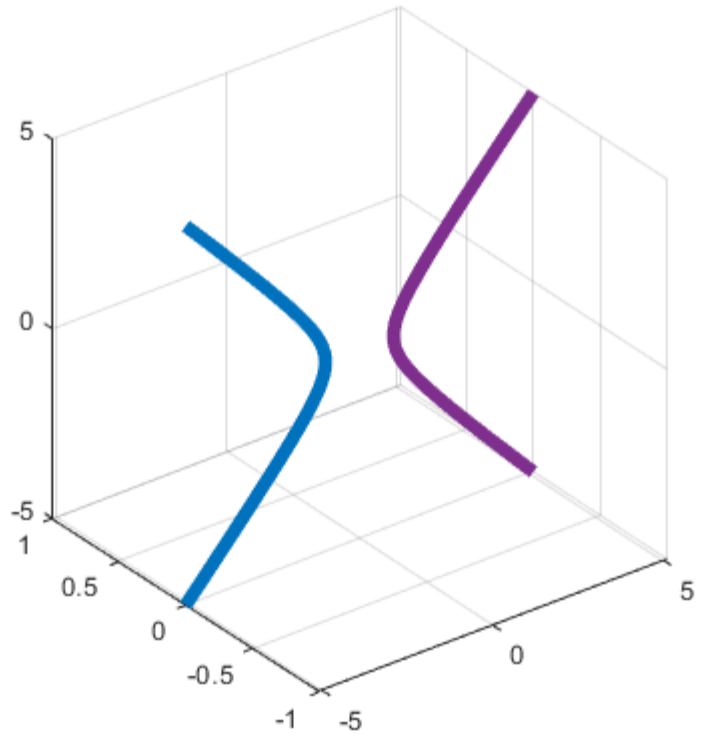
$$\text{(B)}\ \ TD = 0.05us \pm 0.1us$$

- The sign of TD is unambiguous in case (A).
- The sign of TD is ambiguous in case (B) with respect to its uncertainty value.

Time differences approach zero as the emitter goes directly in between two receivers. TD Error becomes more significant at these points and the one-sided method may begin breaking down.

Intersecting 2 Hyperboloids results in a Hyperbola.





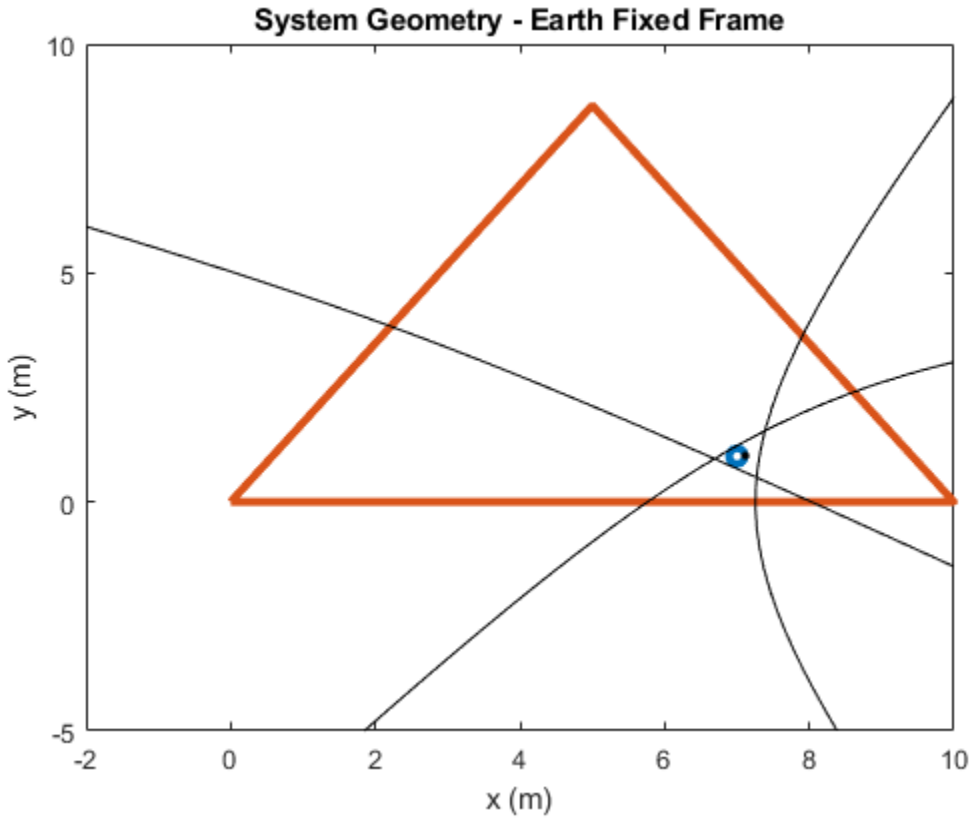
If the Hyperboloids are one-sided, the result is a one-sided Hyperbola.
Still working on whether we can mathematically derive this intersection hyperbola.

Finding Best Solution

By Anthony Iannuzzi

For one-sided Hyperbolas, finding the best solution is the average of 3 points.

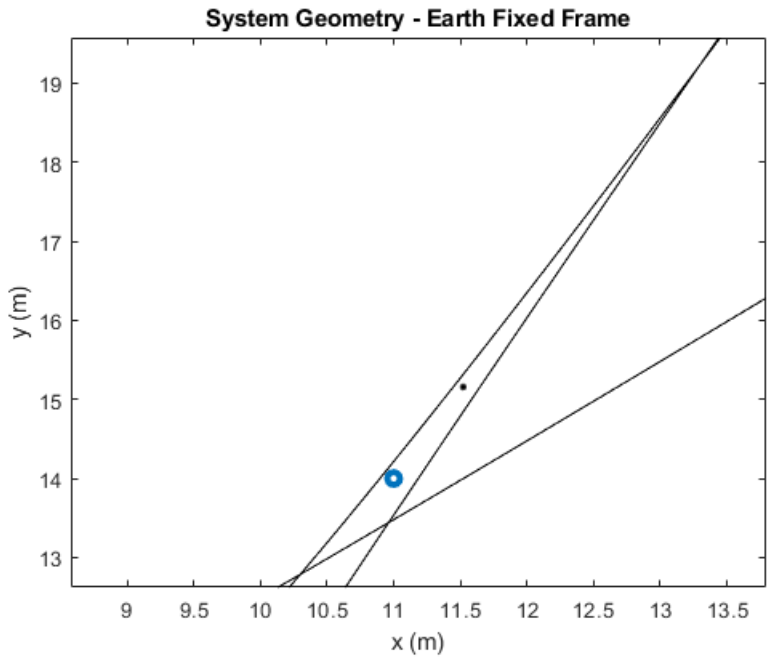
Three one-sided hyperbolas intersect at 3 points only.

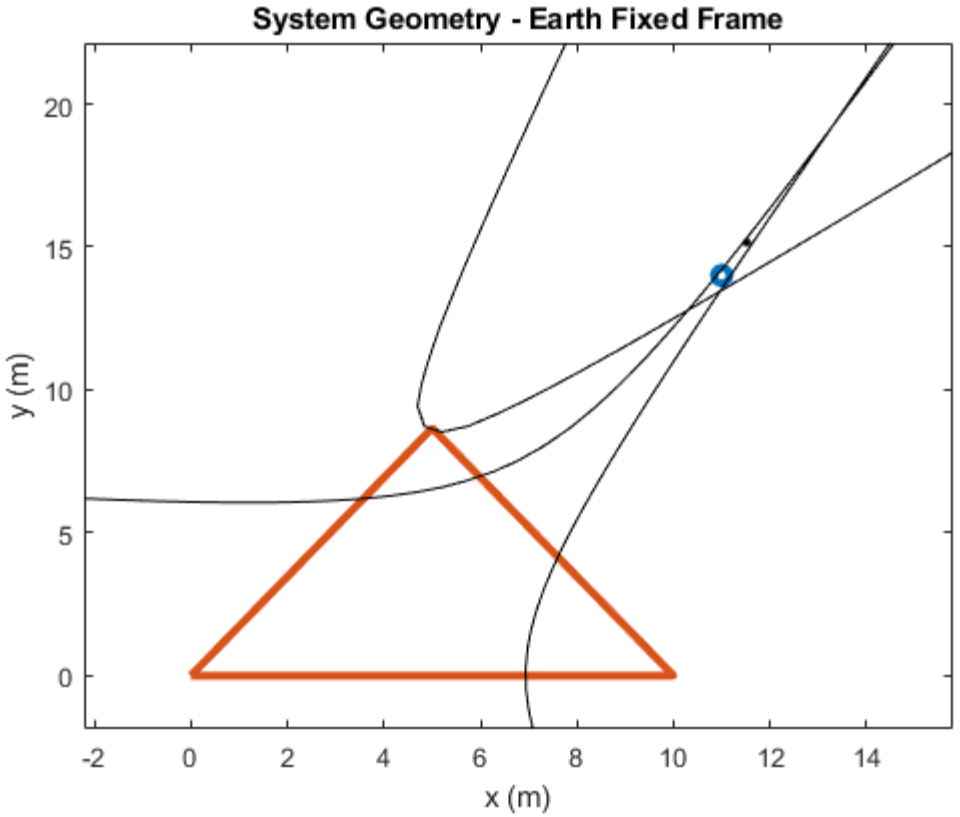


- The blue circle is the correct answer.
- The black dot in the center of the enclosed triangular shape is the “best solution”.

TDoA in 2D is very sensitive to errors outside its “triangle of receivers”.

TDoA error generally looks like this:





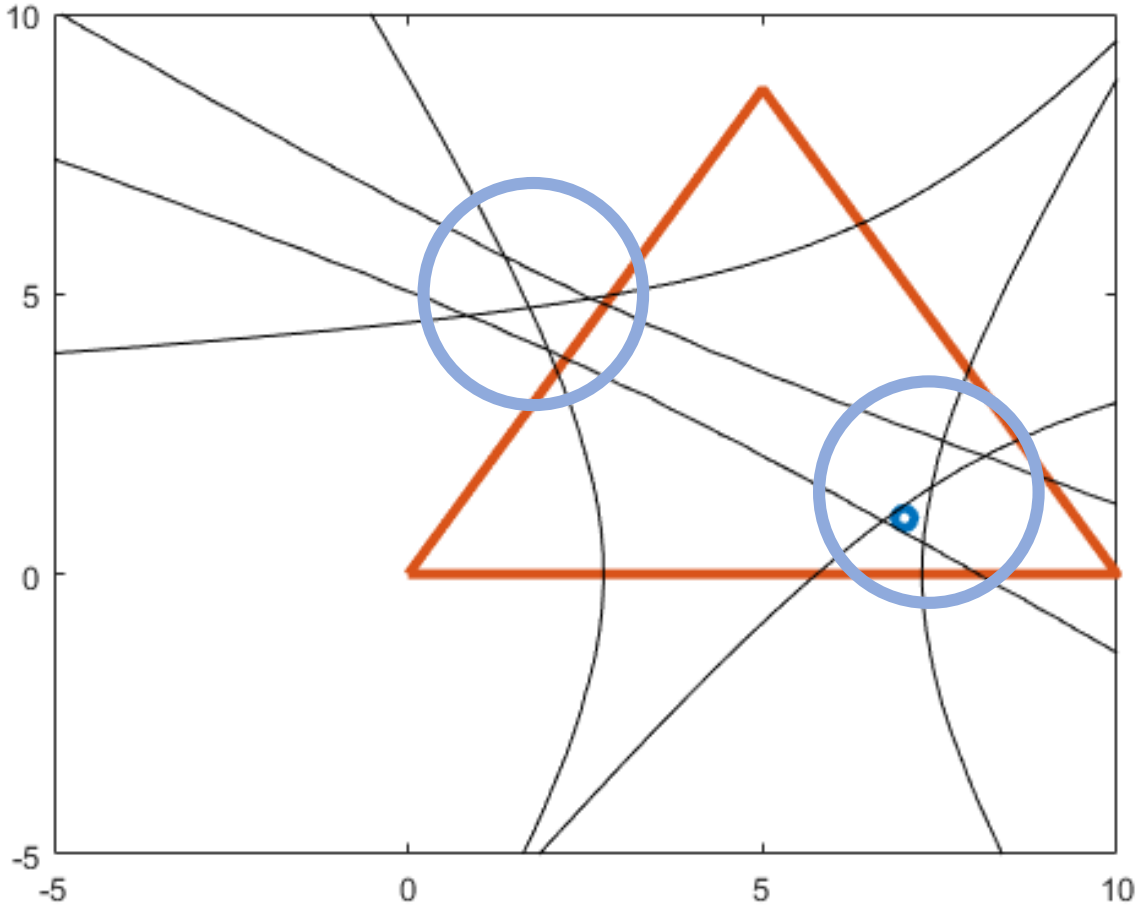
- The blue circle is the correct answer.
- The black dot in the center of the enclosed triangular shape is the “best solution”.

Position error is large, but the direction is still accurate.

Finding the best solution for two-sided hyperbolas is significantly harder.

- Two-sided hyperbolas intersect in 12 different locations.
- Which cluster is the best solution?
- The one with the smallest area?

Approach:
Find all clustered locations by intersecting the hyperbolas 2 at a time.



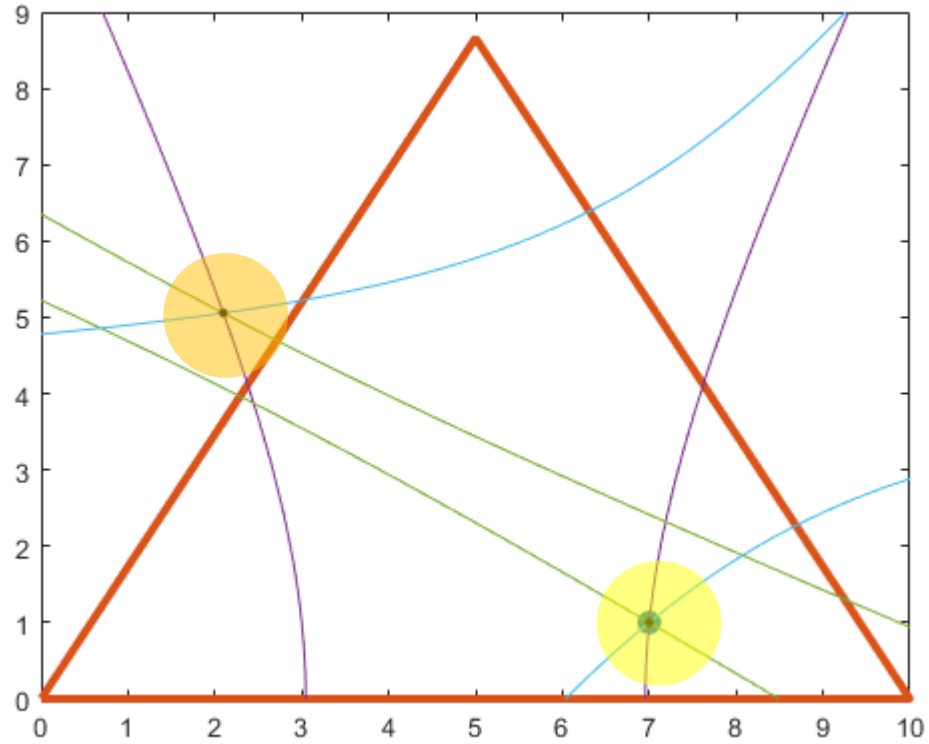
Example with no error: 2 clusters are close, but they still form a triangle.

1	7.0001	0.9998
	7.1886	2.3197
	2.4457	3.8727
	2.0964	5.0562

2	1.9290	-5.5781
	7.0000	1.0001
	2.0963	5.0564
	9.3175	9.0656

3	6.9888	0.9970
	8.0753	1.8799
	0.6839	4.8581
	2.0965	5.0565

- Numerical precision prevents the 3 points from lining up $exactly$.
- Intersecting 3 hyperbolas results in no solution.

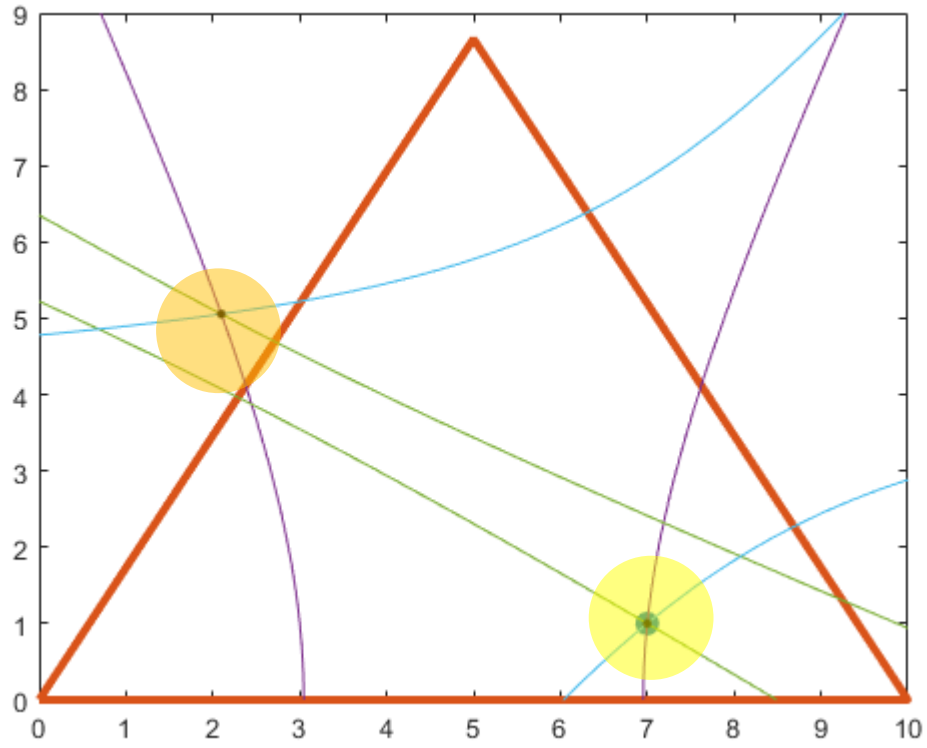


$$H_A - H_B = 0$$

Where $H_A, \mathrm{H_B}$ are equations of Hyperboloids.

First sort the data with respect to x then y. Differentiate the data. Single out the data below 0.1 difference.

	x sorted	**y sorted**	difference	difference
1	0.6839	4.8581	1.2451	-10.4362
2	1.9290	-5.5781	0.1673	10.6345
3	2.0963	5.0564	0.0001	-0.0002
4	2.0964	5.0562	0.0001	0.0003
5	2.0965	5.0565	0.3492	-1.1838
6	2.4457	3.8727	4.5431	-2.8757
7	6.9888	0.9970	0.0112	0.0031
8	7.0000	1.0001	0.0001	-0.0003
9	7.0001	0.9998	0.1885	1.3199
10	7.1886	2.3197	0.8866	-0.4399
11	8.0753	1.8799	1.2423	7.1858
12	9.3175	9.0656		



Decide how many solutions there are by looking for which singled out points are clustered.

Index	Diff - 1	x sorted	y sorted	difference	difference
1		0.6839	4.8581	1.2451	-10.4362
2		1.9290	-5.5781	0.1673	10.6345
3	0	2.0963	5.0564	0.0001	-0.0002
4	2	2.0964	5.0562	0.0001	0.0003
5		2.0965	5.0565	0.3492	-1.1838
6		2.4457	3.8727	4.5431	-2.8757
7	0	6.9888	0.9970	0.0112	0.0031
8		7.0000	1.0001	0.0001	-0.0003
9	0	7.0001	0.9998	0.1885	1.3199
10		7.1886	2.3197	0.8866	-0.4399
11		8.0753	1.8799	1.2423	7.1858
12		9.3175	9.0656		

- Collect the indices of all singled out differences.

$$\boldsymbol{Indices} = [3,4,7,8]$$

- Take the differences of these differences.

$$diff(Indices) = [1,3,1]$$

$$\boldsymbol{diff(Indices) - 1 = [0,2,0]}$$

- The number of solutions is equal to the number of non-zero terms.

The critical points to collect are the beginning, end, and all non-zero terms in Diff-1.

Index	Diff - 1	x sorted	y sorted	difference	difference
1		0.6839	4.8581	1.2451	-10.4362
2		1.9290	-5.5781	0.1673	10.6345
3	0	2.0963	5.0564	0.0001	-0.0002
4	2	2.0964	5.0562	0.0001	0.0003
5		2.0965	5.0565	0.3492	-1.1838
6		2.4457	3.8727	4.5431	-2.8757
7	0	6.9888	0.9970	0.0112	0.0031
8		7.0000	1.0001	0.0001	-0.0003
9	0	7.0001	0.9998	0.1885	1.3199
10		7.1886	2.3197	0.8866	-0.4399
11		8.0753	1.8799	1.2423	7.1858
12		9.3175	9.0656		

$$\boldsymbol{diff(Indices) - 1 = [0, 2, 0]}$$

- The number of solutions is equal to the number of non-zero terms.
- The critical points identify switching between solutions.
- The first critical point is always 0.
- The last critical point is the number of singled out points. (4 in this ex.)

$$criticalPoints = [0,2,4]$$

Use critical points to identify the indices of the solution

Index	Diff - 1	x sorted	y sorted	difference	difference
1		0.6839	4.8581	1.2451	-10.4362
2		1.9290	-5.5781	0.1673	10.6345
3	0	2.0963	5.0564	0.0001	-0.0002
4	2	2.0964	5.0562	0.0001	0.0003
5		2.0965	5.0565	0.3492	-1.1838
6		2.4457	3.8727	4.5431	-2.8757
7	0	6.9888	0.9970	0.0112	0.0031
8		7.0000	1.0001	0.0001	-0.0003
9		7.0001	0.9998	0.1885	1.3199
10		7.1886	2.3197	0.8866	-0.4399
11		8.0753	1.8799	1.2423	7.1858
12		9.3175	9.0656		

$$Indices = [3,4,7,8]$$

$$criticalPoints = [0,2,4]$$

We now iterate through critical points, collecting all data between the previous and current critical point:

$$Indices\big(split(i) + 1 : split(i+1)\big)$$

For the first solution we have

$$Indices(1:2) = [3,4]$$

For the second solution we have

$$Indices(3:4) = [7,8]$$

The last point in the cluster is the point directly above the identified ranges.

Index	Diff - 1	x sorted	y sorted	difference	difference
1		0.6839	4.8581	1.2451	-10.4362
2		1.9290	-5.5781	0.1673	10.6345
3	0	2.0963	5.0564	0.0001	-0.0002
4	2	2.0964	5.0562	0.0001	0.0003
5		2.0965	5.0565	0.3492	-1.1838
6		2.4457	3.8727	4.5431	-2.8757
7	0	6.9888	0.9970	0.0112	0.0031
8		7.0000	1.0001	0.0001	-0.0003
9		7.0001	0.9998	0.1885	1.3199
10		7.1886	2.3197	0.8866	-0.4399
11		8.0753	1.8799	1.2423	7.1858
12		9.3175	9.0656		

For the first solution we have

$$Indices(1:2) = [3,4]$$

For the second solution we have

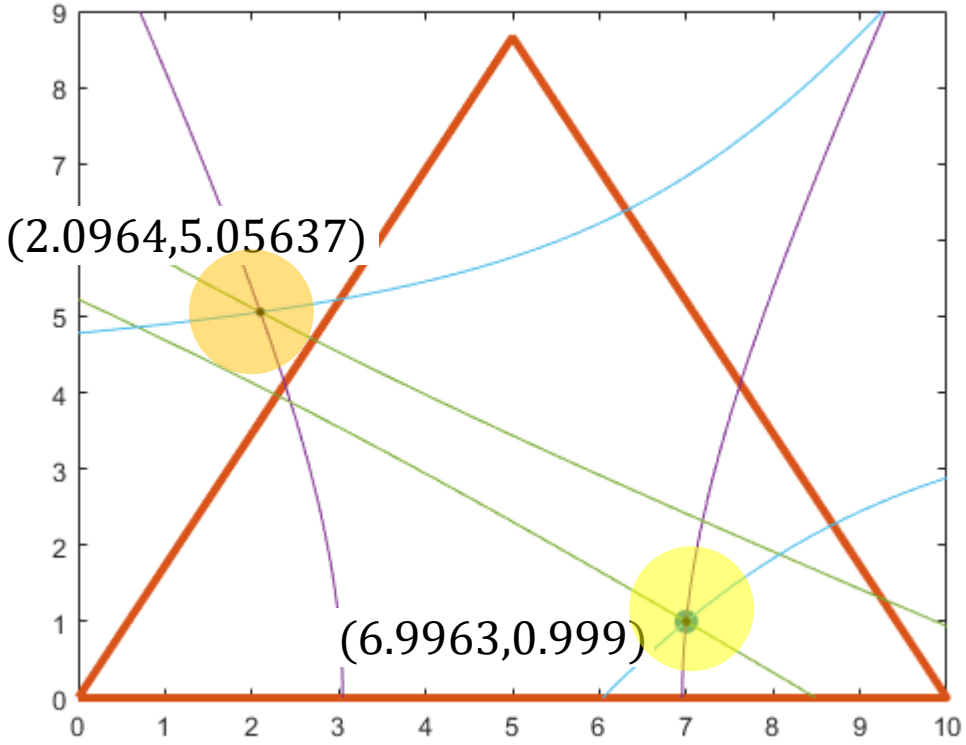
$$Indices(3:4) = [7,8]$$

The diff function lags one element behind. We add the next consecutive element to each solution:

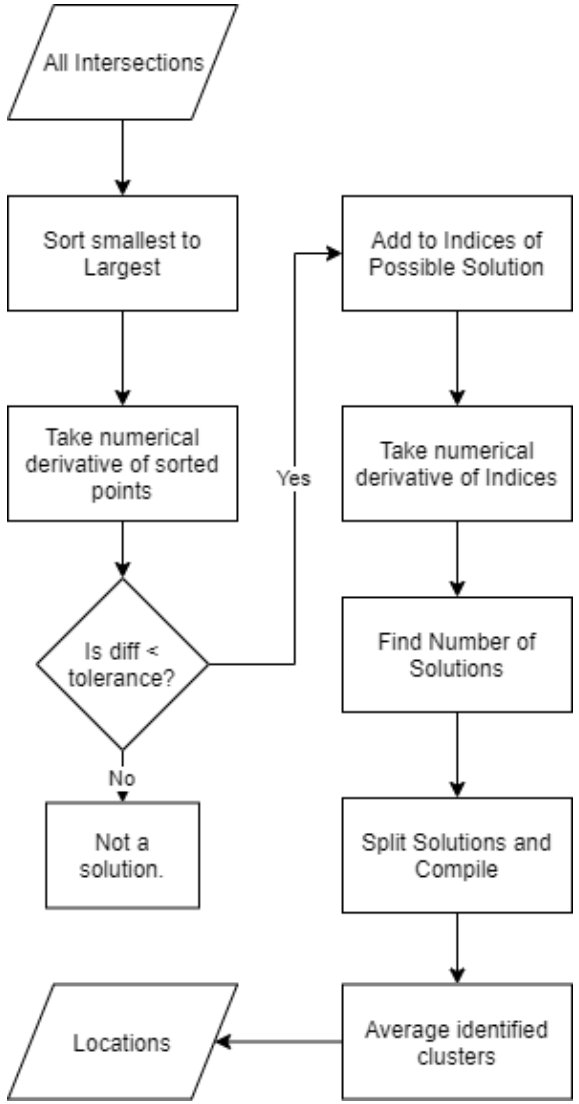
$$soln1 = [3,4,5]$$
$$soln2 = [7,8,9]$$

Finally, average the solution together.

Index	Diff - 1	x sorted	y sorted	difference	difference
1		0.6839	4.8581	1.2451	-10.4362
2		1.9290	-5.5781	0.1673	10.6345
3	0	2.0963	5.0564	0.0001	-0.0002
4	2	2.0964	5.0562	0.0001	0.0003
5		2.0965	5.0565	0.3492	-1.1838
6		2.4457	3.8727	4.5431	-2.8757
7	0	6.9888	0.9970	0.0112	0.0031
8		7.0000	1.0001	0.0001	-0.0003
9		7.0001	0.9998	0.1885	1.3199
10		7.1886	2.3197	0.8866	-0.4399
11		8.0753	1.8799	1.2423	7.1858
12		9.3175	9.0656		

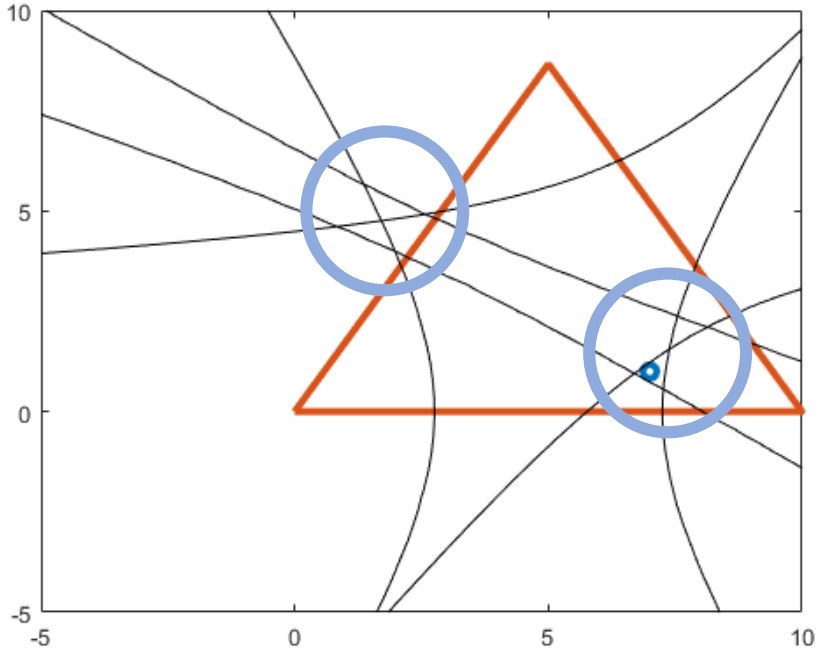


Choose the “Best” Solution Flowchart



Limitations to this algorithm

- This algorithm works well for 2 clusters of points.
- It does not work well when there are 4 clusters of points with roughly the same size:

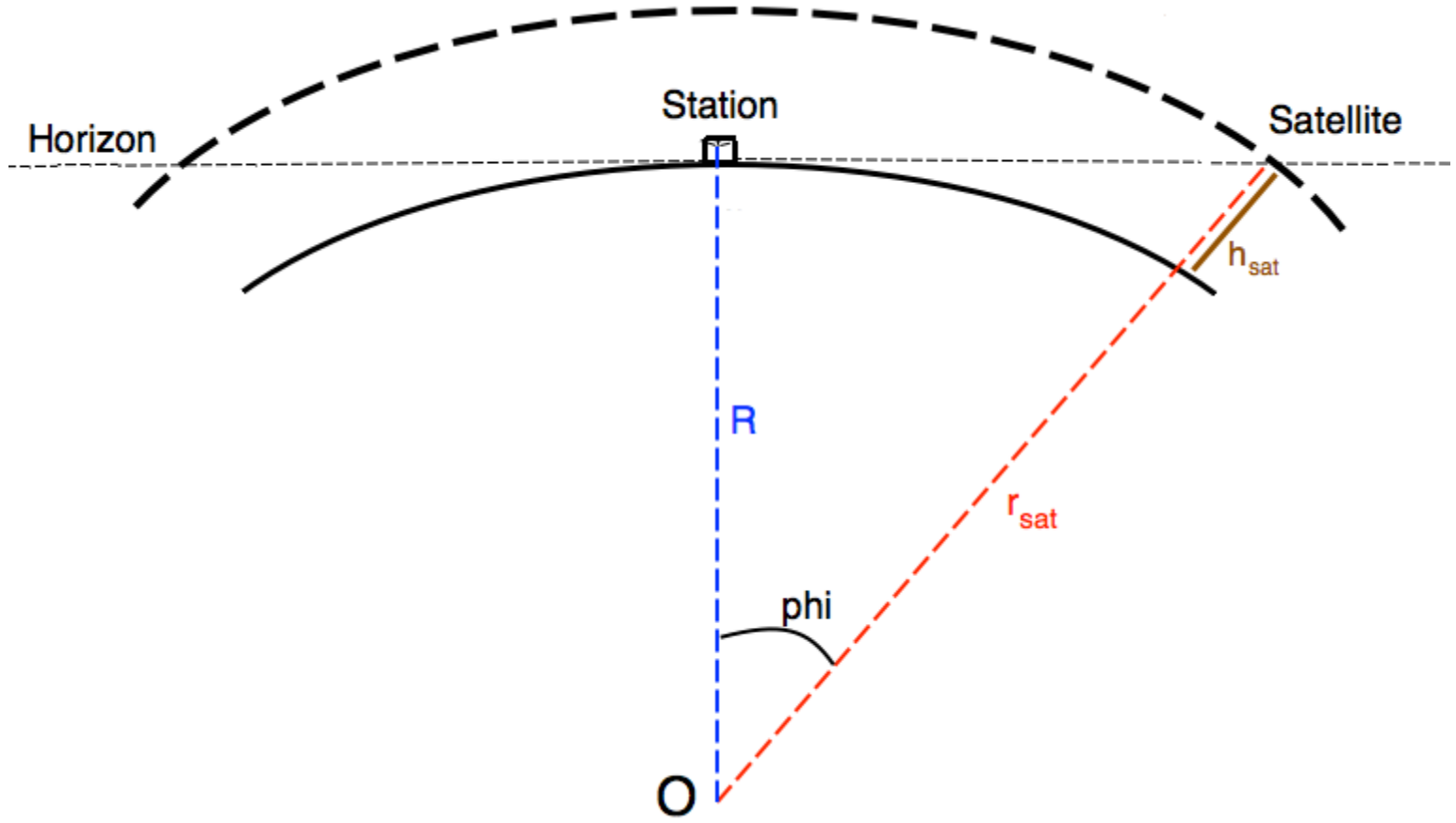


- Averaging the points makes sense for a near equilateral triangle. It might not for an isosceles triangle.

The Satellite Problem

By Anthony Iannuzzi

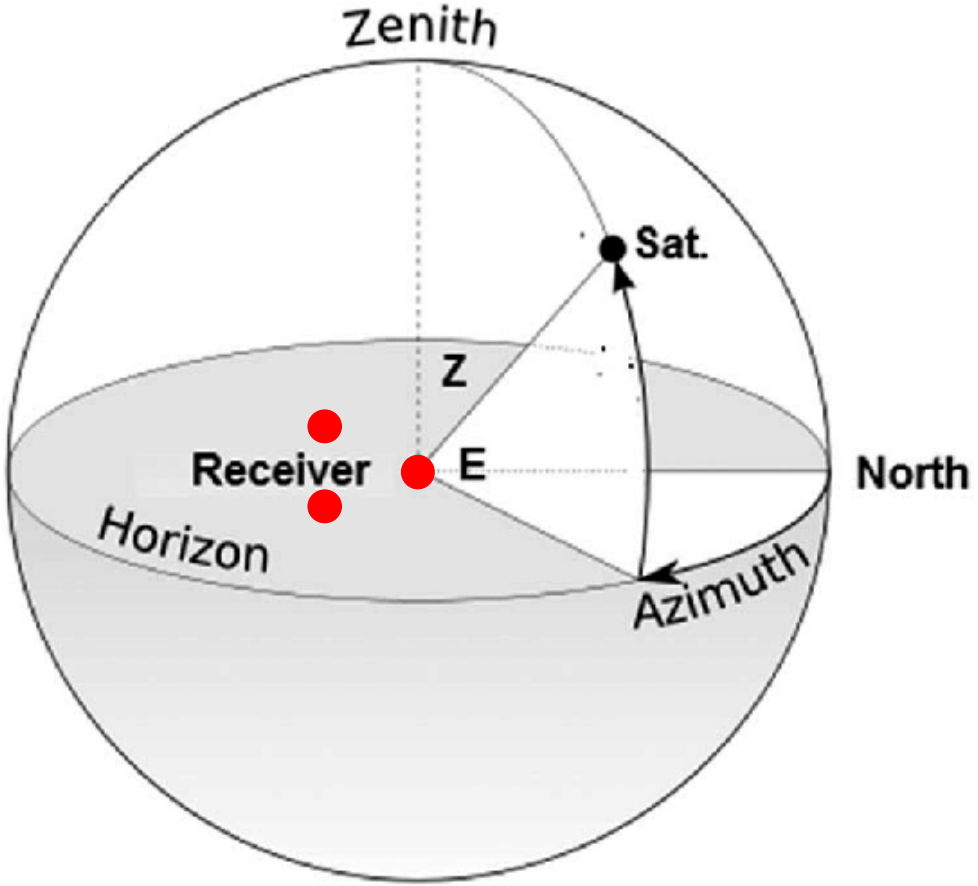
The satellite is much closer to the station near elevation 90° than at lower elevations.



http://deepearthscience.blogspot.com/2015/06/diy-satellite-pass-planner.html

The satellite is far away from the plane of the receivers except when elevation is $\sim 0°$

- The satellite is far away from the receivers.
- For the case of elevation $\sim 0°$, the satellite is far outside the “triangle”.
- For any other elevation, the satellite is on a different plane.



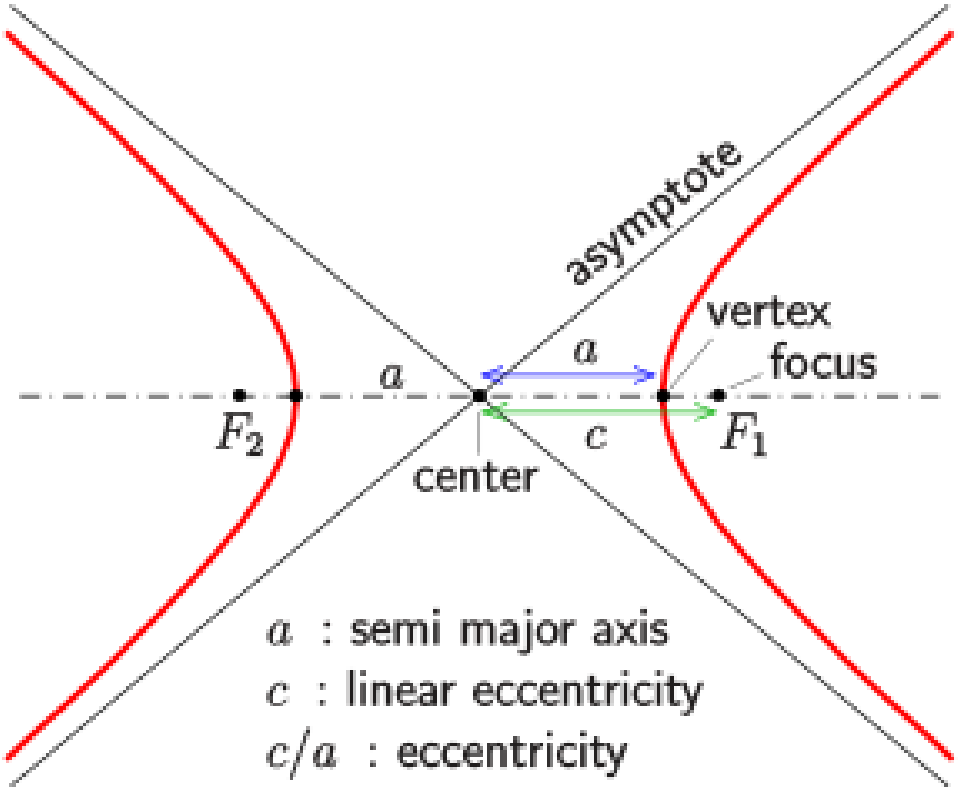
https://www.researchgate.net/figure/Satellite-Azimuth-and-elevation-angle_fig1_334197029

Far away, hyperbolas look like cones.

- Hyperbolas approach a line as x or y approach infinity.
- A 2D one-sided hyperbola is:

$$x'' = \delta\sqrt{\frac{1}{4} + \frac{(y'')^2}{4D^2 - \delta^2}}$$

If y'' is large, then the ¼ can be neglected.



https://en.wikipedia.org/wiki/Hyperbola

Far away, hyperbolas look like cones.

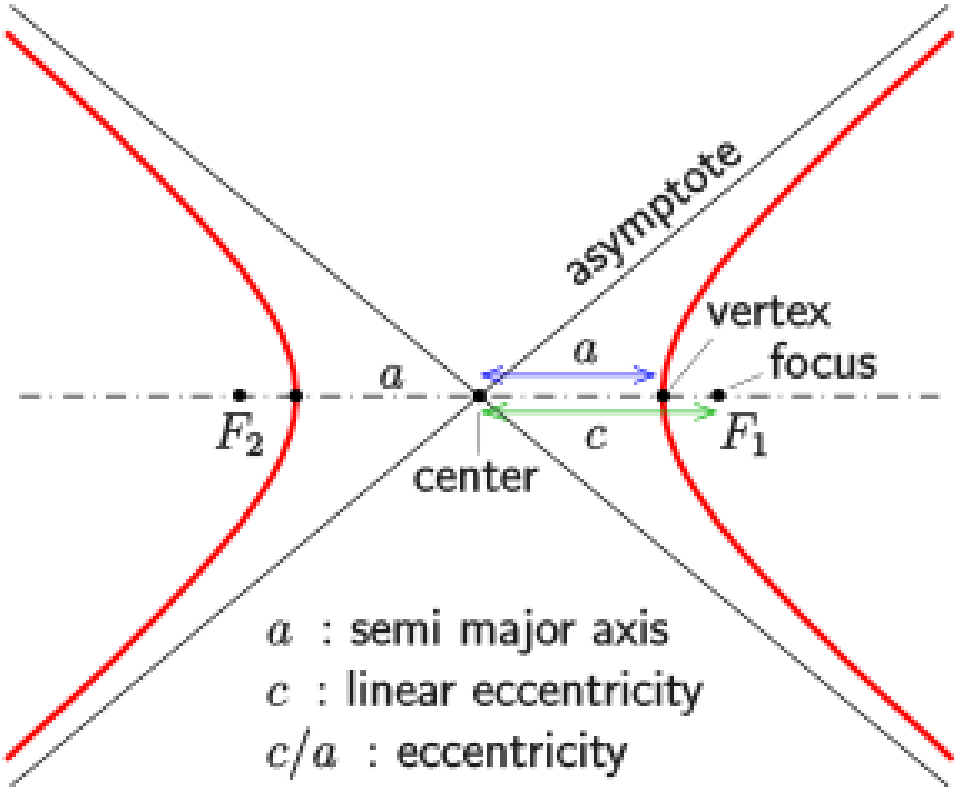
$$x'' = \delta\sqrt{\frac{1}{4} + \frac{(y'')^2}{4D^2 - \delta^2}}$$

If y'' is large, then the ¼ can be neglected.

$$x'' = \delta\sqrt{\frac{(y'')^2}{4D^2 - \delta^2}}$$

$$x'' = \delta|y''|\sqrt{\frac{1}{4D^2 - \delta^2}}$$

- This is the equation of the asymptotes.



https://en.wikipedia.org/wiki/Hyperbola

The line equation is

$$x = \frac{a}{b}|y|$$

Where $b^2 = 4D^2 - \delta^2$

The strategy for getting the satellite direction.

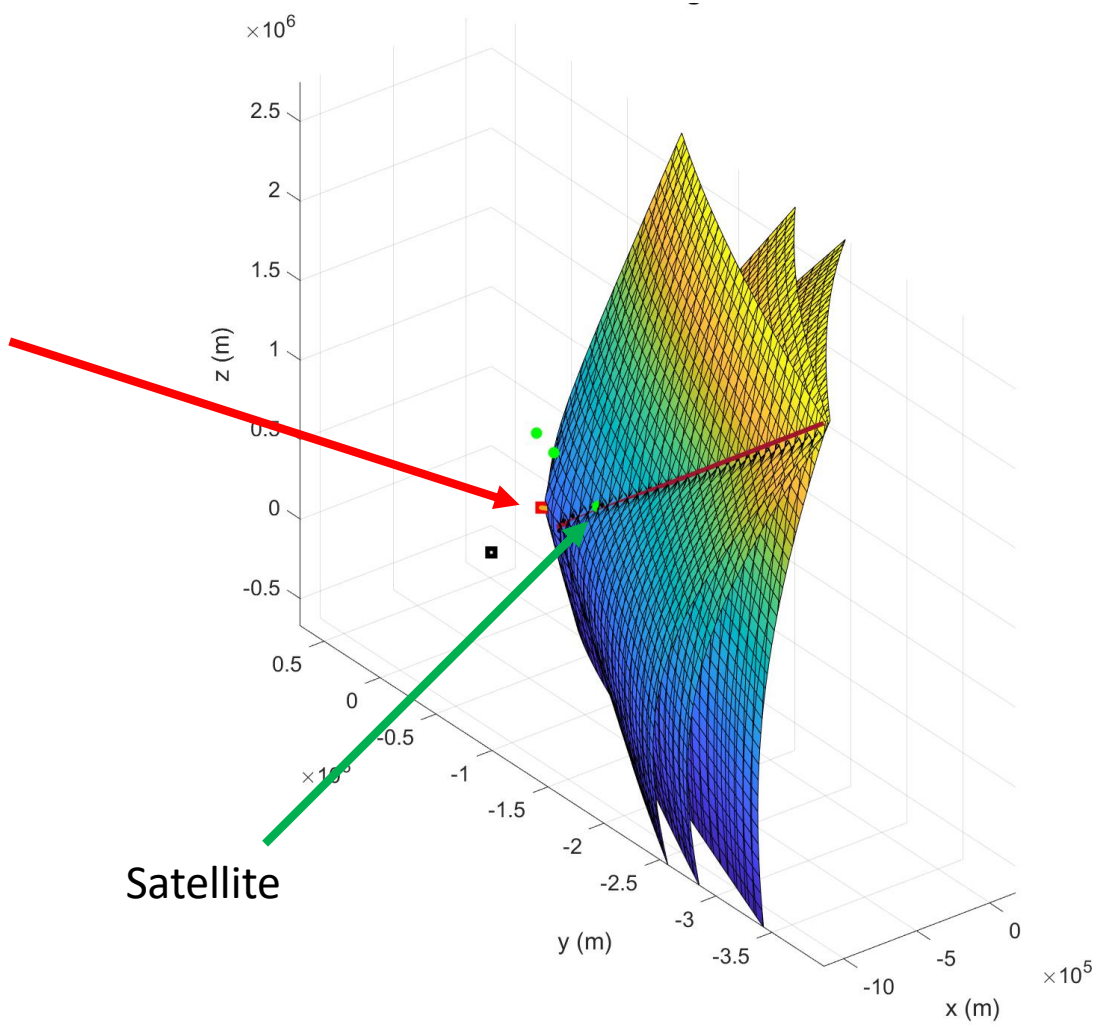
- 2 Hyperboloids intersect on a hyperbola.
- Far away from the foci, a hyperbola looks like a line.
- The direction of the satellite can be derived from a line.

1. Draw the hyperboloids.
2. Solve multiple 2D problems on different planes.
3. Fit a line to the 2D solutions.

More on specifics in a later section.

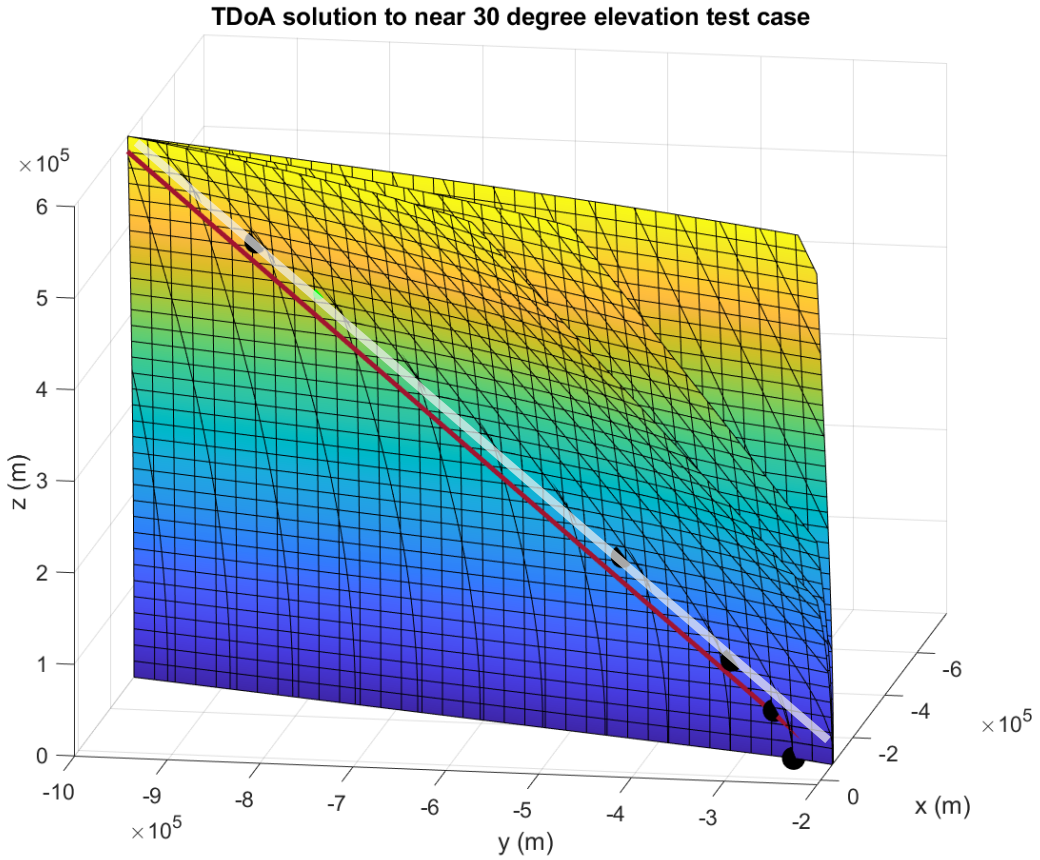
What do the hyperboloids look like for the satellite problem?

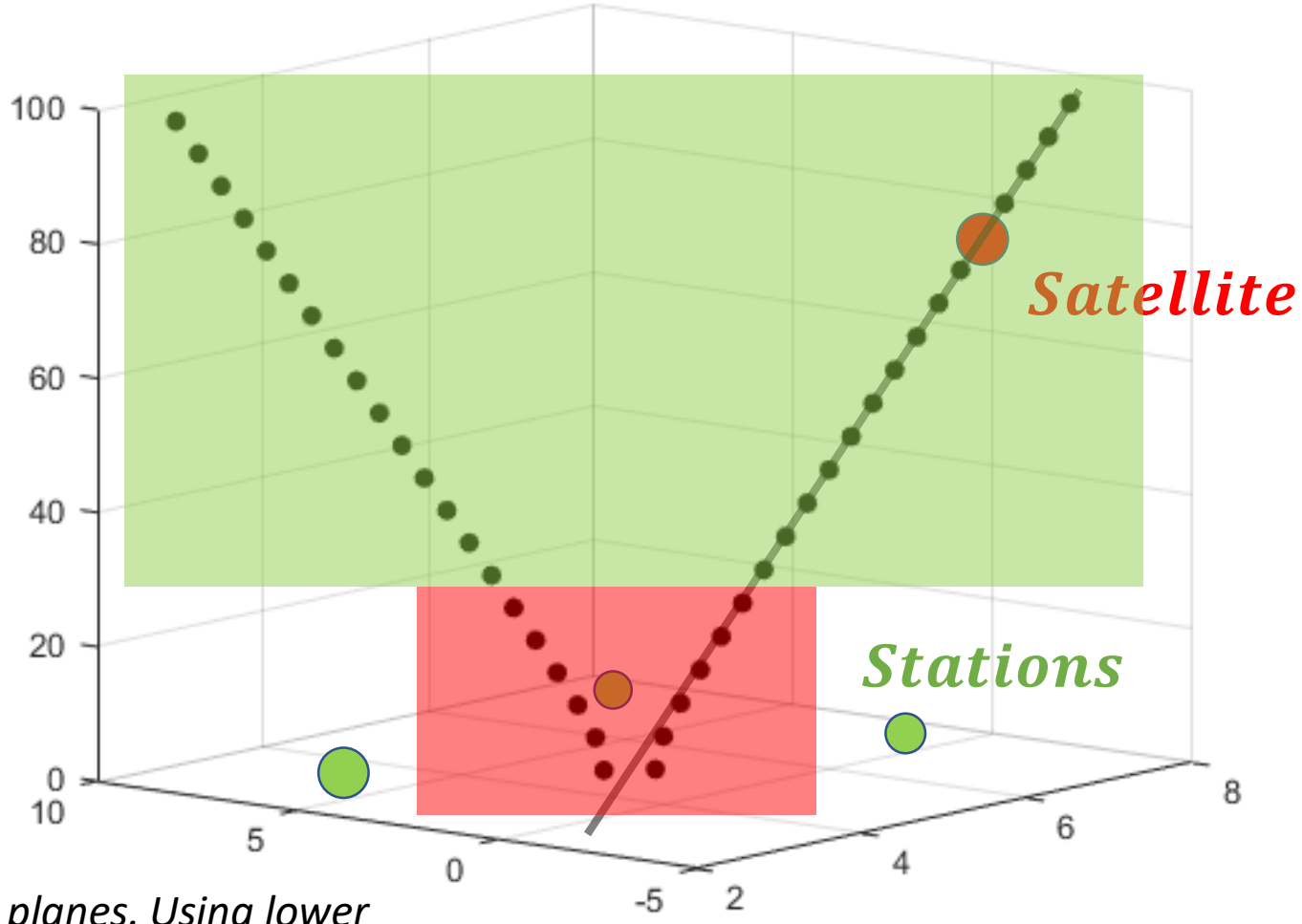
Receiving stations



Satellite

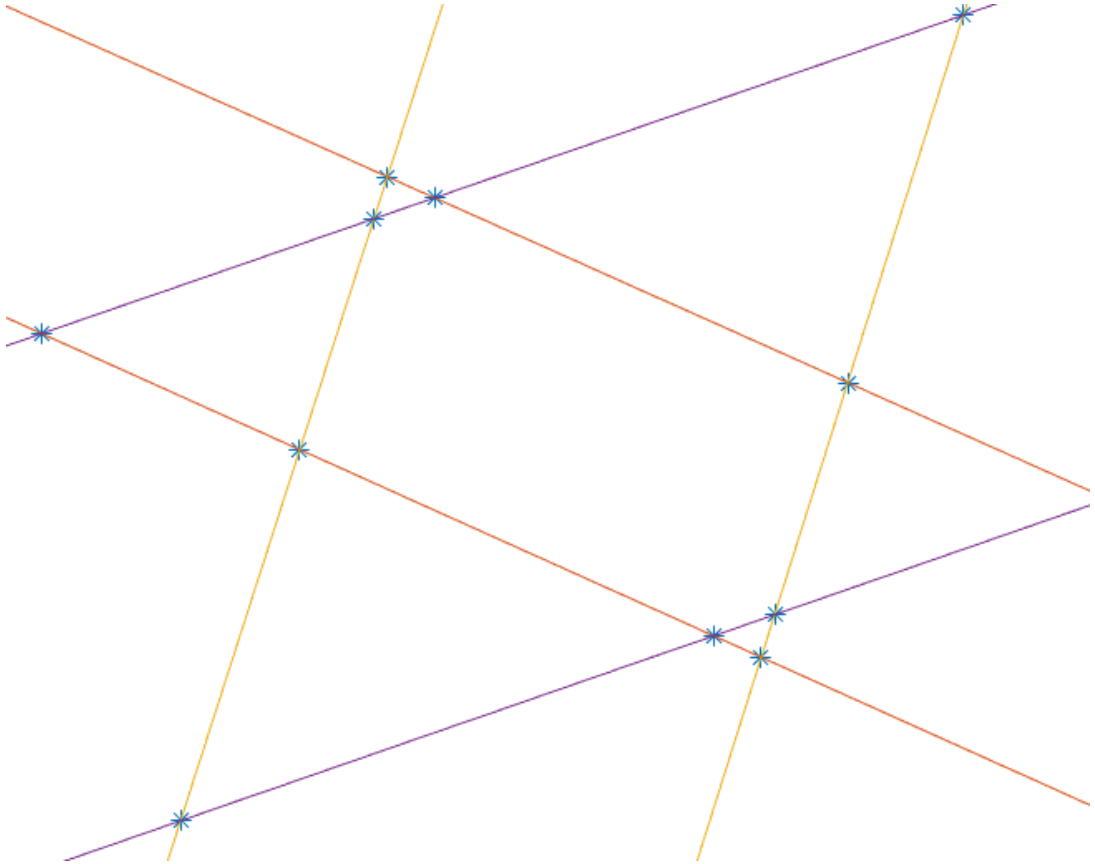
Hyperboloids intersect on a hyperbola. We can fit a line because the satellite is FAR away from the plane of stations.



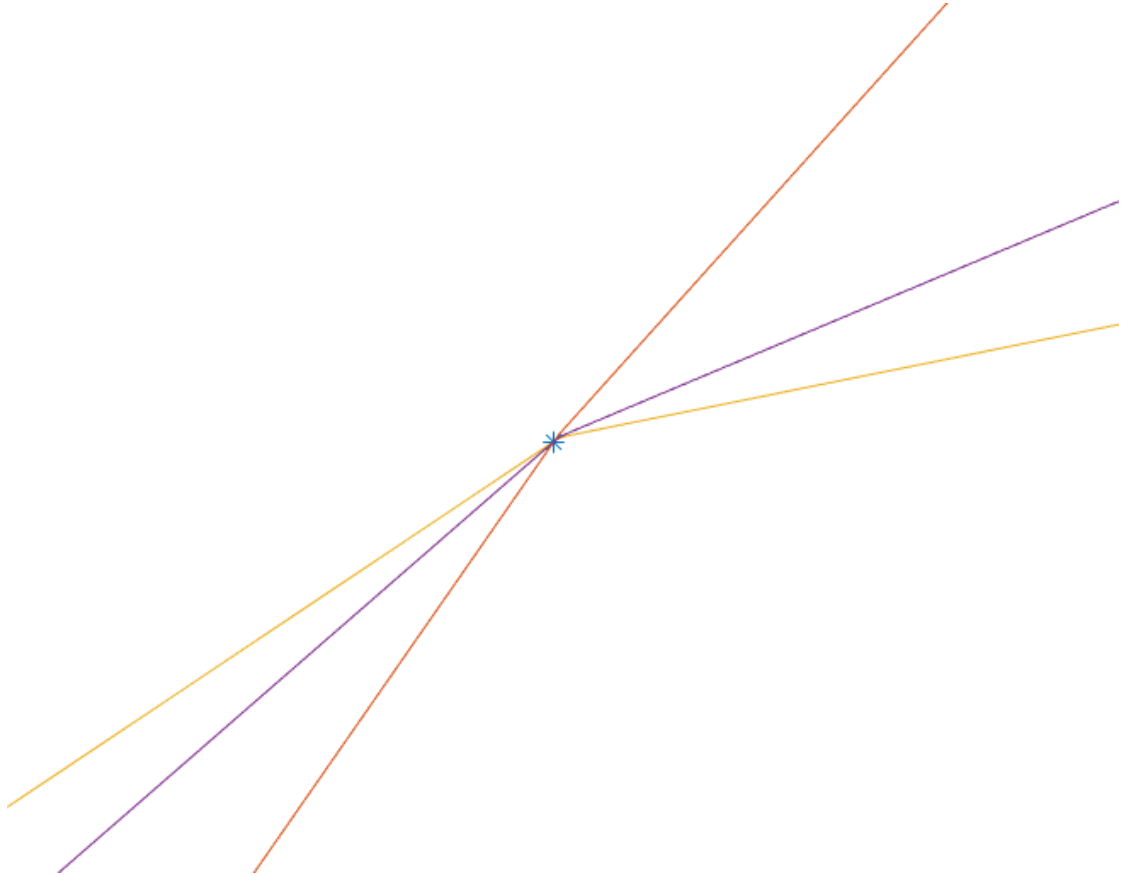


White line fits using 3 planes at least 100km above the planes. Using lower planes pull the line down (red)

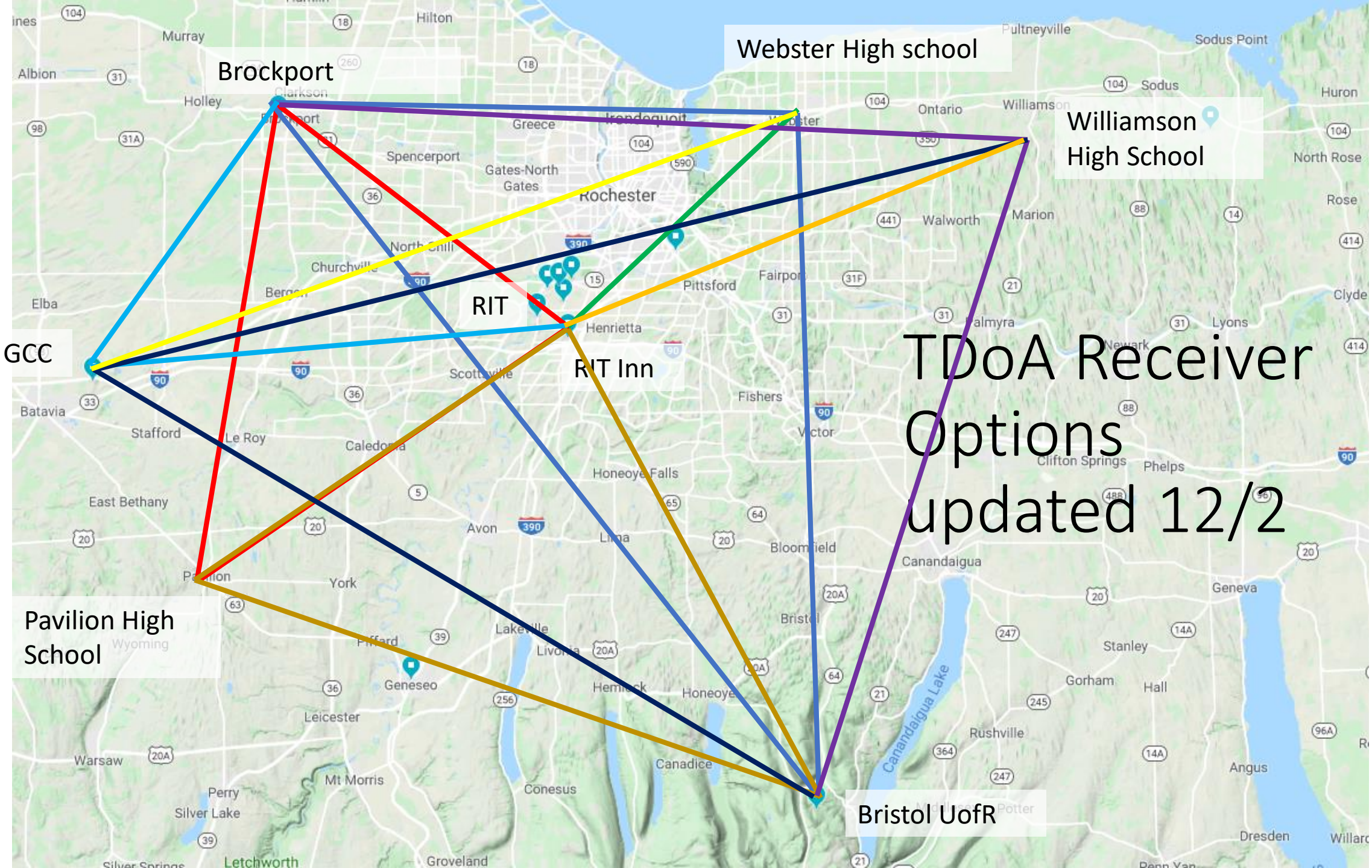
What does the localizing the satellite look like on a plane far away from the receivers?



2-sided Hyperbolas



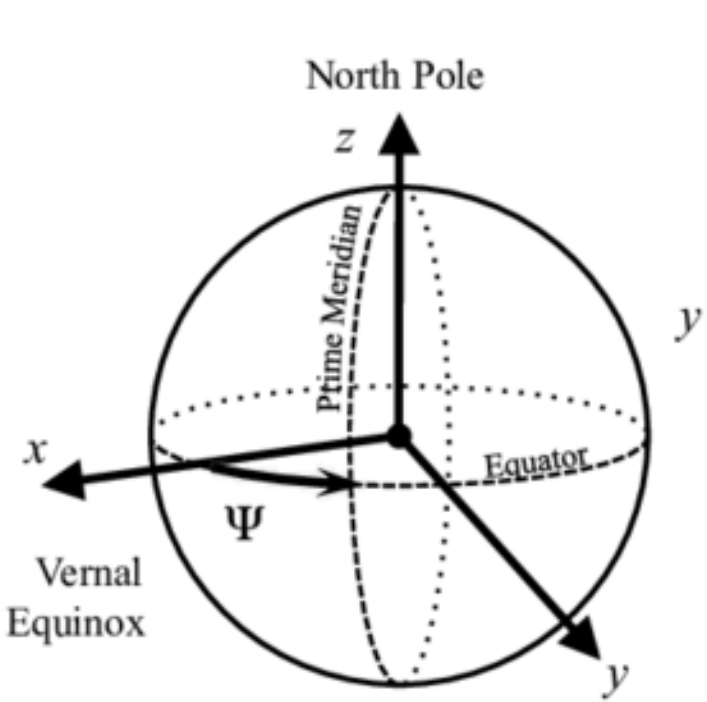
1-sided Hyperbolas


Webster High school
Brockport
Williamson High School
GCC
RIT
RIT Inn
Pavilion High School
Bristol UofR
TDoA Receiver Options updated 12/2

Coordinate Transformations Overview.

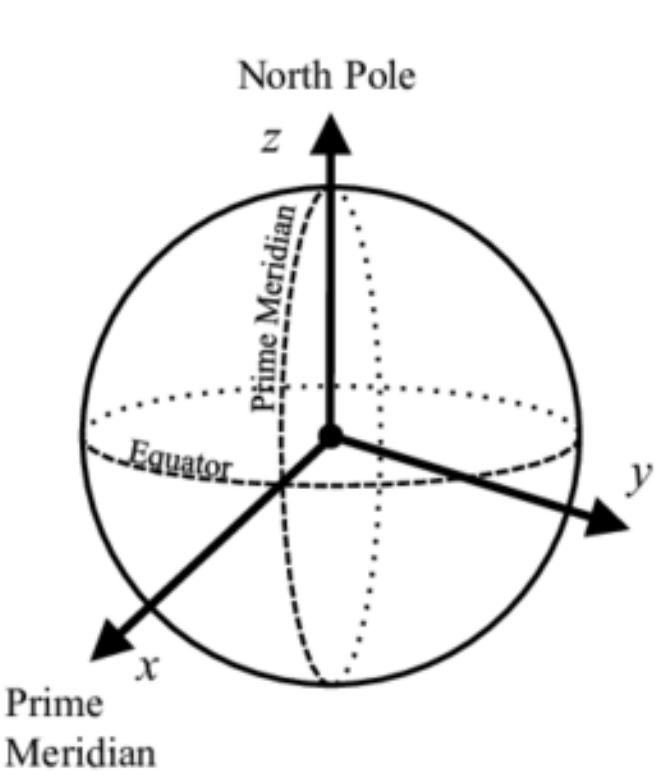
- Receivers are usually in geodetic coordinates
$$latitude, longitude, altitude$$
- Satellite coordinates are usually in orbital plane or an Inertial frame, like J2000.
$$x_f, y_f, z_f$$
- To convert the receivers to x,y,z coordinates, the shape of the Earth must be taken into account.
- To convert the satellite to Earth fixed coordinates, the rotation rate and orientation of the Earth must be taken into account.

The 3 Earth frames are used for different parts of the system.



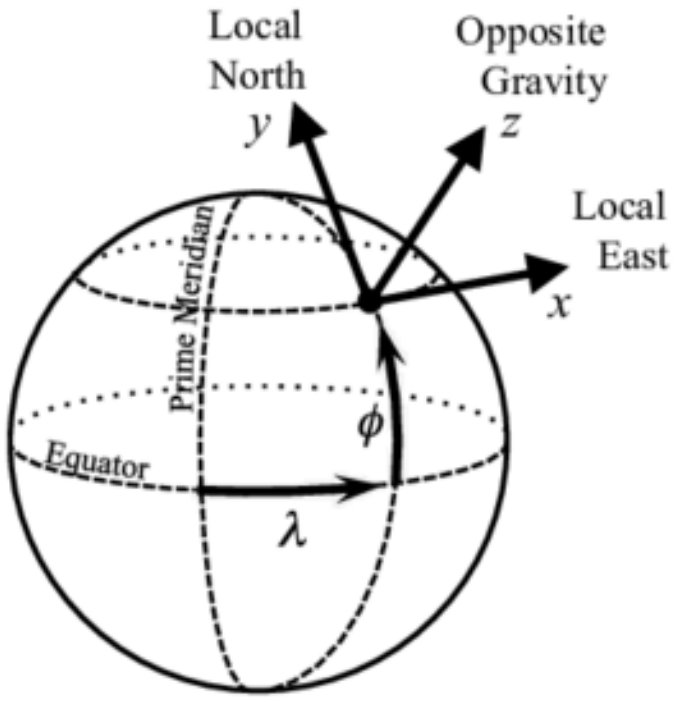
EARTH CENTRED INERTIAL: *I*

Where the satellite is measured.



EARTH CENTRED FIXED: *F*

Ground station coordinates



TOPOCENTRIC: *T*

Satellite coordinates with respect to a station.

Transforming from Geodetic to Earth Fixed requires the shape of the Earth. We use WGS84.

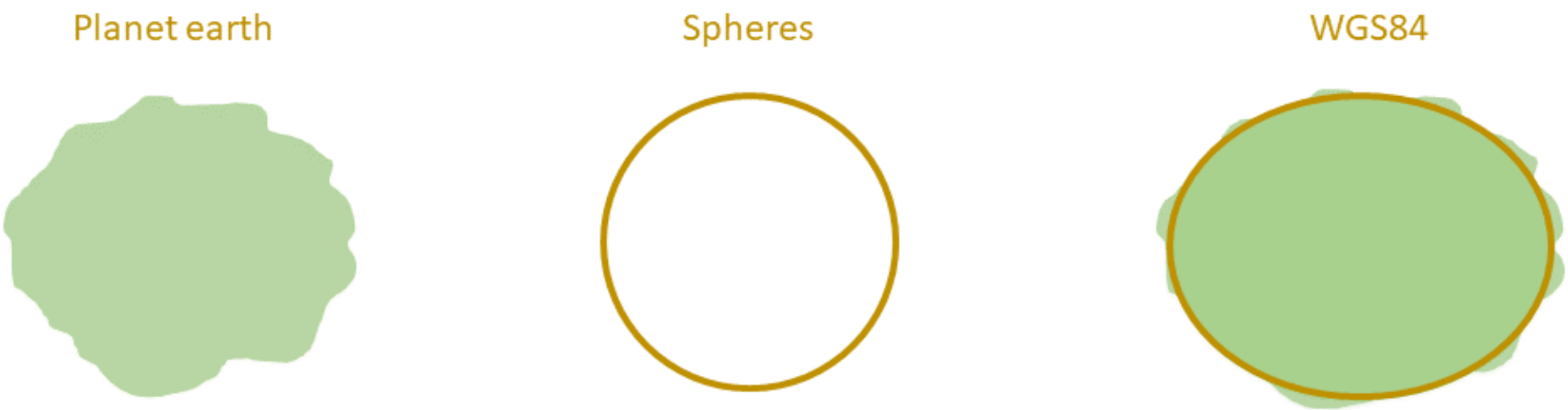
Spherical Model

Analytical (Lat,Long) -> (x,y,z)

WGS84 Model

More accurate model.

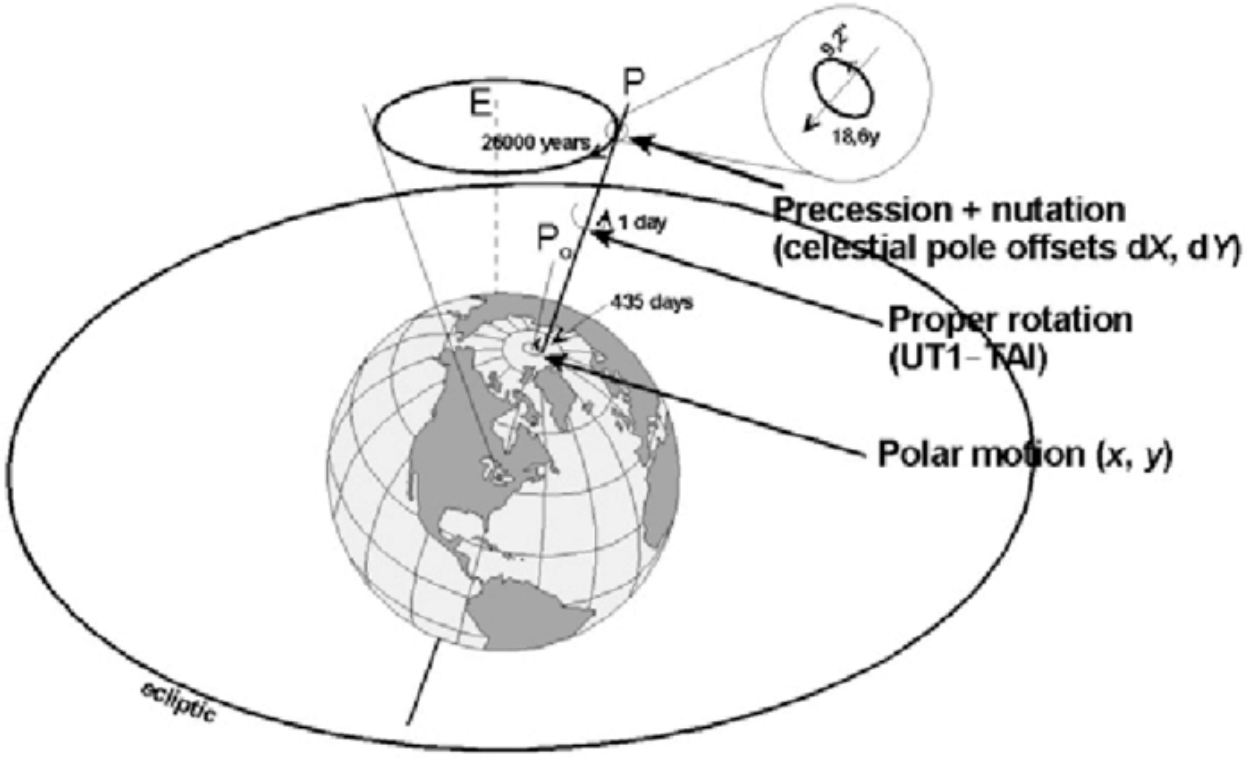
Matlab and Orekit Built-in functions available. Analvtical solution is more



http://support.virtual-surveyor.com/en/support/solutions/articles/1000261351-what-is-wgs84-

Transforming Satellite from Earth Inertial to Earth Fixed.

- The Earth rotates once in about 24 hours
- It undergoes one precession period in 26,000 years.
- It undergoes one nutation period every 18.6 years
- The poles wander one full period in 435 days.



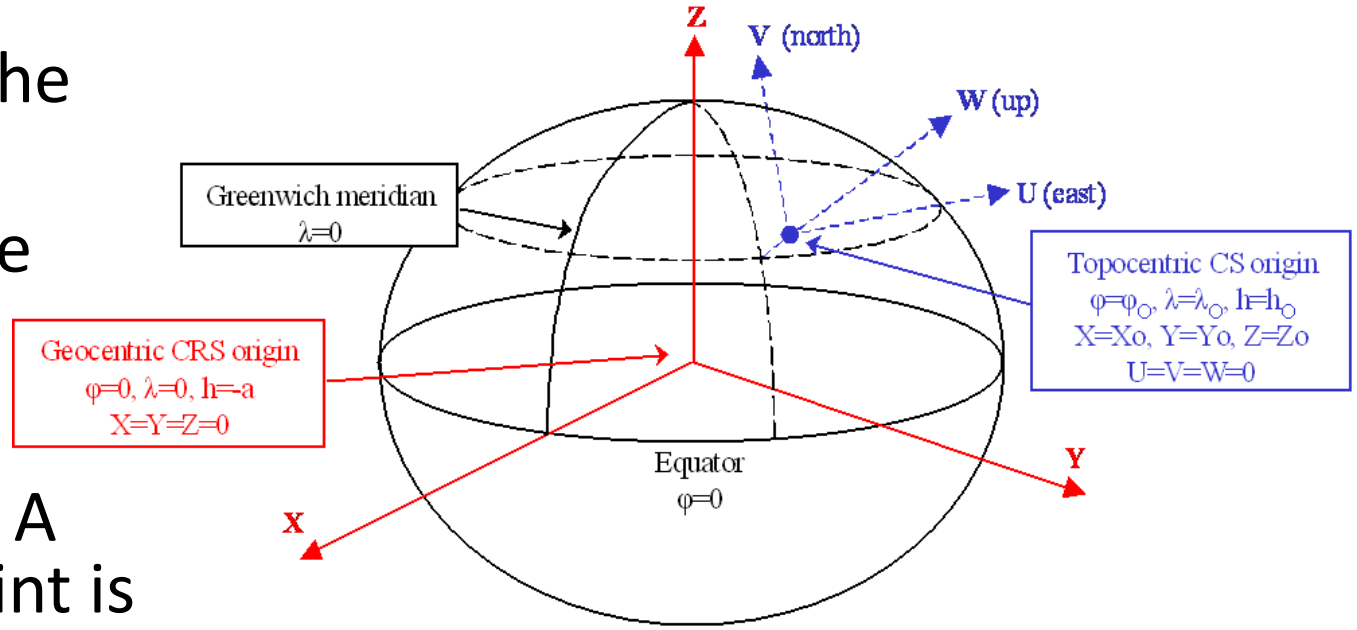
https://www.researchgate.net/figure/Earth-orientation-parameters_fig1_289388318

Orekit in Java handles these transformations. https://www.orekit.org/

Transforming from Earth Fixed to Topocentric

Topocentric frame orientation:

- The z-axis points normal to the surface
- The y-axis points towards true north
- The x-axis points east.
- Frame describes the horizon. A positive zenith means the point is in-view.
- The Azimuth/Elevation conversions can be applied to these coordinates to get a direction.



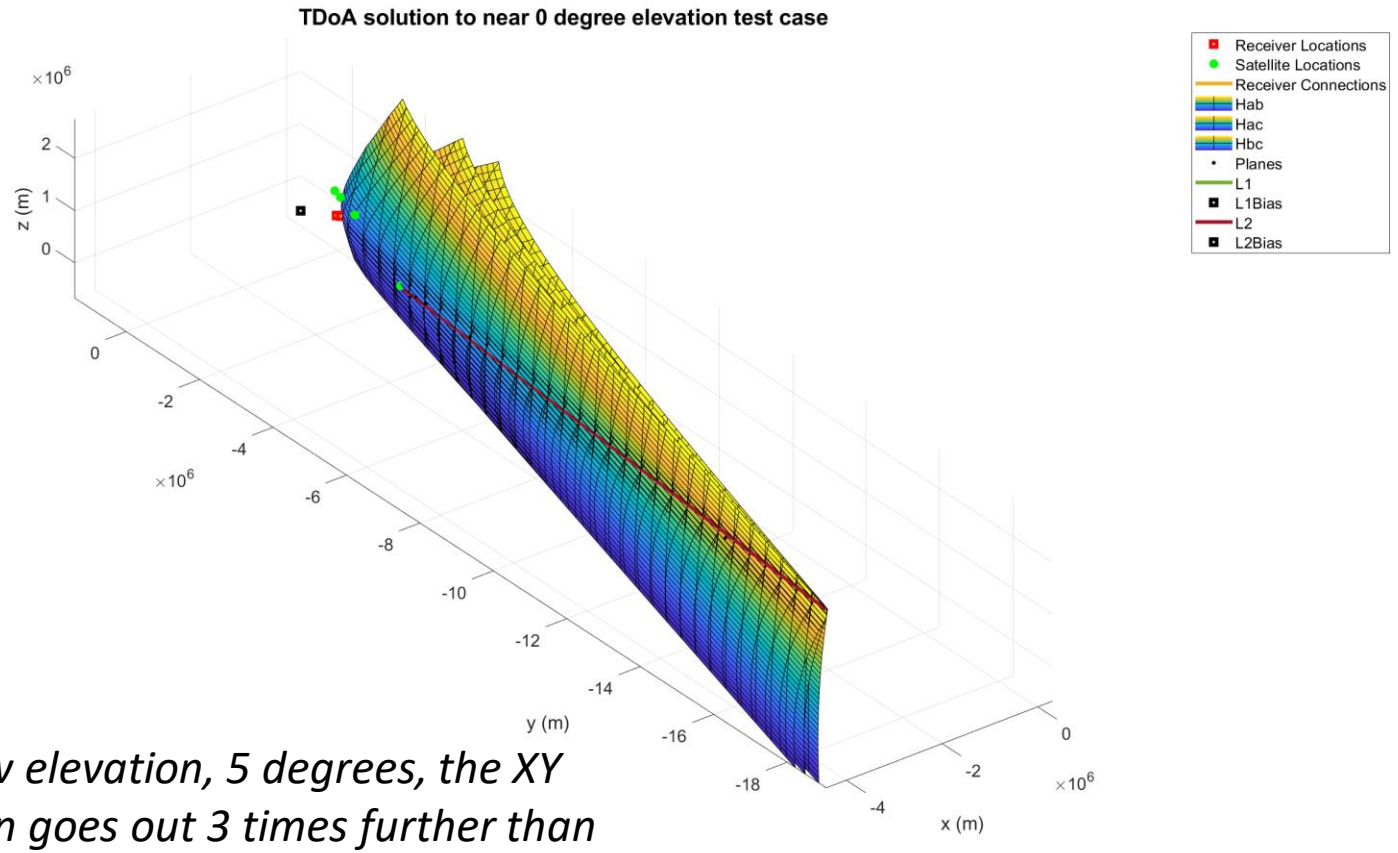
http://www.hydrometronics.com/ecef.html

Finding the Direction

By Anthony Iannuzzi

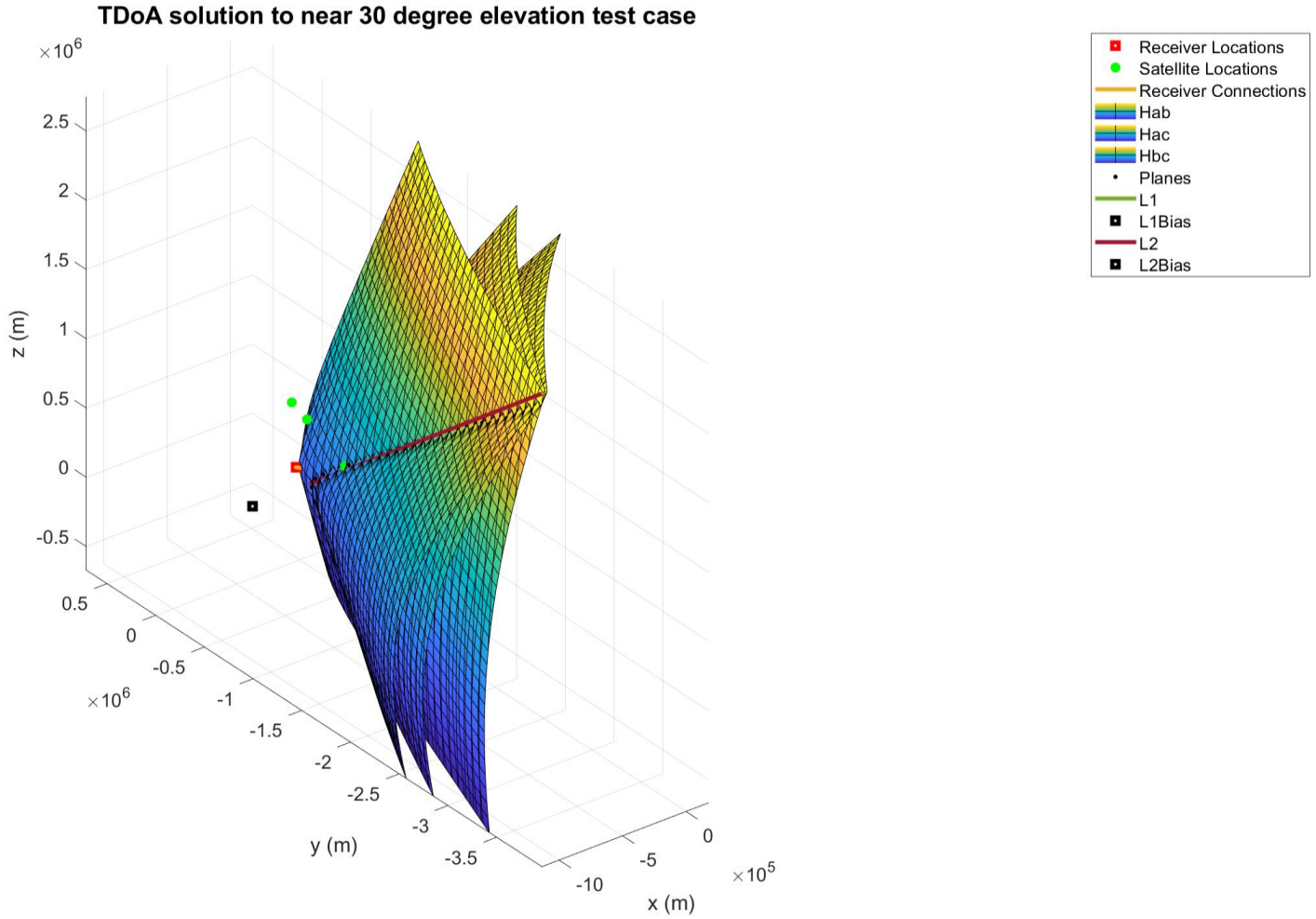
We solve TDoA on 50km, 400km, and 1200km planes.

- This was a compromise to minimize elevation error.
- For low elevation satellites, solving a 2D TDoA problem on high z values does not make sense: the XY values begin to diverge.



For this low elevation, 5 degrees, the XY intersection goes out 3 times further than the satellite for the 1200km plane.

For 30 degree elevation, the fit goes out to 2x the satellite position.



We fit a parametric line to the planes we solve TDoA on.

Parametric Equation of a Line: $r = r_0 + tr_d$

Where

$r = [x, y, z]$

$r_0 = [x_0, y_0, z_0]$ starting point of line

$r_d = [x_d, y_d, z_d]$ direction vector

n is number of points

$D = 2$ for a linear line fit

$y = mx + b$

We fit three linear lines

- *t vs. x*
- *t vs. y*
- *t vs. z*

Fit a line to the data points from each plane.

For each dimension, do a linear regression fit

$$y = Mw$$

$$x = (M^T M)^{-1} * M^T * y$$

For 3 points:

$$\underset{nx1}{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}} = \underset{nxD}{\begin{bmatrix} 1 & t_1 \\ 1 & t_2 \\ 1 & t_3 \end{bmatrix}} * \underset{Dx1}{\begin{bmatrix} b \\ m \end{bmatrix}}$$

To get Azimuth and Elevation from a line, we need to choose a reference point on that line.

For simplicity, we use r_0 as our reference.

Parametric Equation of a Line: $r = r_0 + tr_d$

Where

$r = [x, y, z]$

$r_0 = [x_0, y_0, z_0]$ starting point of line

$r_d = [x_d, y_d, z_d]$ direction vector

$$az = \tan^{-1}\frac{x_{east}}{y_{north}}$$

$$el = \tan^{-1}\frac{z_{up}}{\sqrt{x_{east}^2 + y_{north}^2}}$$

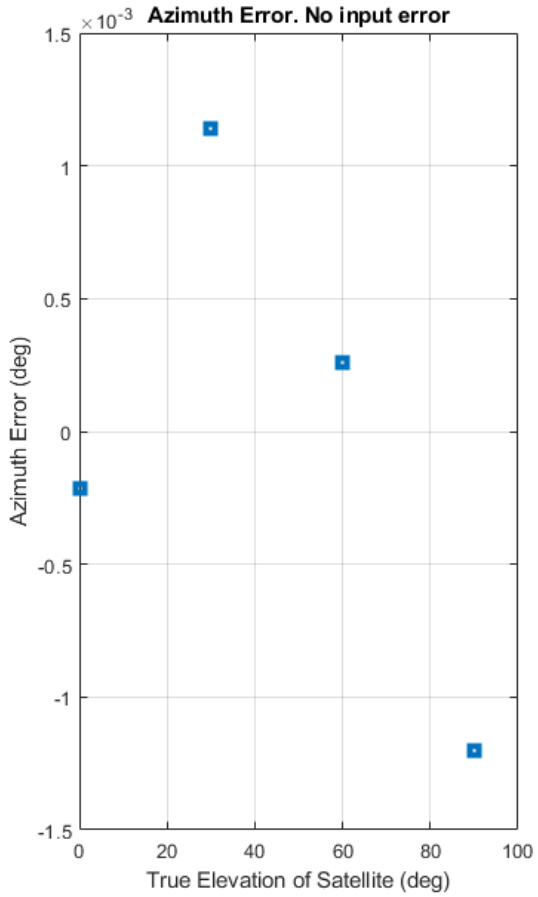
Notice azimuth is X/Y. Azimuth for ground tracks is measured from north, not east!

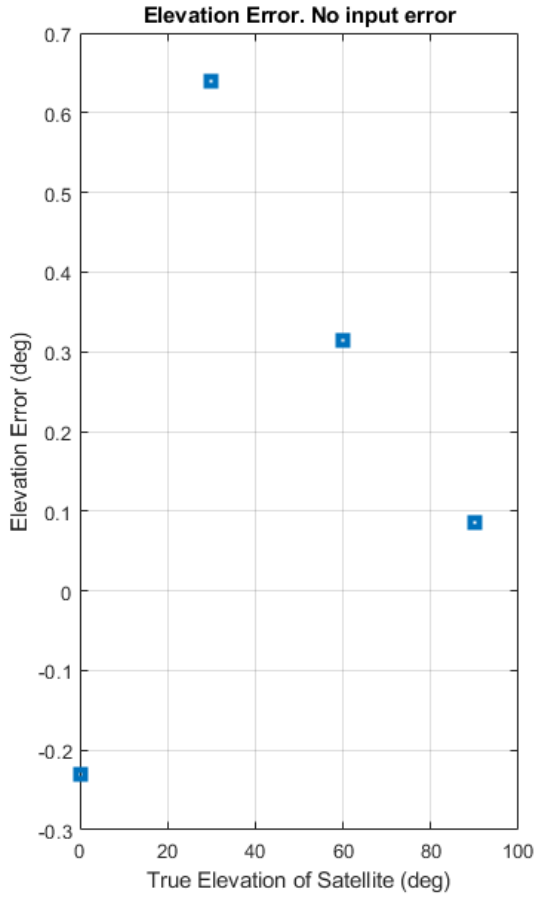
r_0 and r_d are measured wrt to station 1.

1. Convert r_0 and $r_0 + r_d$ to the Earth Fixed Frame.
2. Measure $r_0 + r_d$ in a topocentric frame centered at r_0

   Result: $(x_{east}, y_{north}, z_{up})$ wrt to r_0
3. Convert coordinates to azimuth and elevation.

Algorithm inherent error is $\pm 0.7°$ based on a 4-point Ground Track and 6 planes.

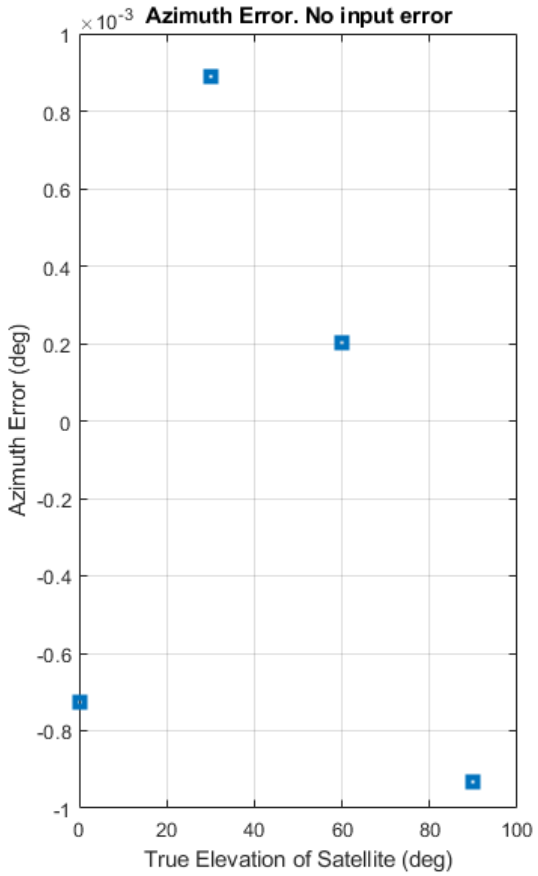


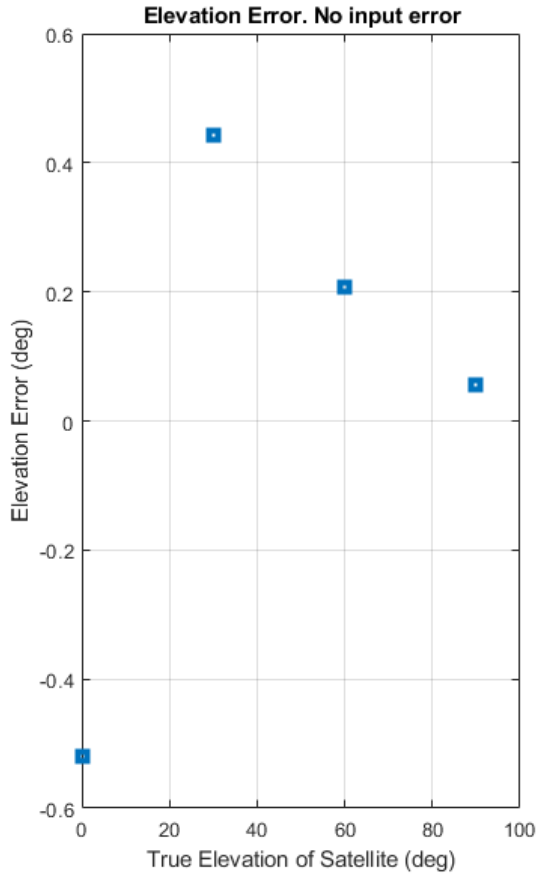


Line Fit to:
0, 50km, 100km,
200km, 500km,
2000km planes

Algorithm inherent error is $\pm 0.6°$ based on a 4-point Ground Track and 3 planes.

- Error in Elevation decreases for elevations 30 degrees and above.
- Error in Elevation increases below that.





Line Fit to:
50km, 400km, 1200km planes

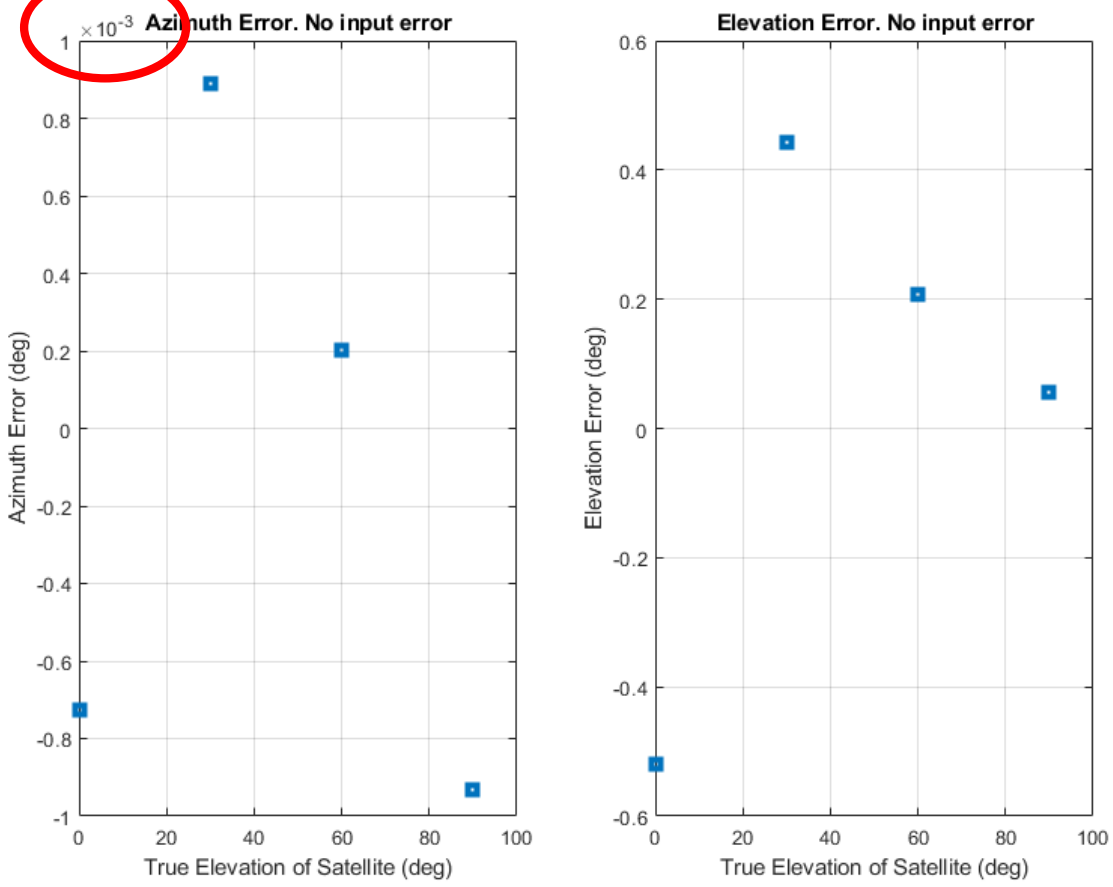
We can reduce TDoA intrinsic error with a few strategies.

Lower planes decrease error at low elevations. Higher planes decrease error at high elevations.

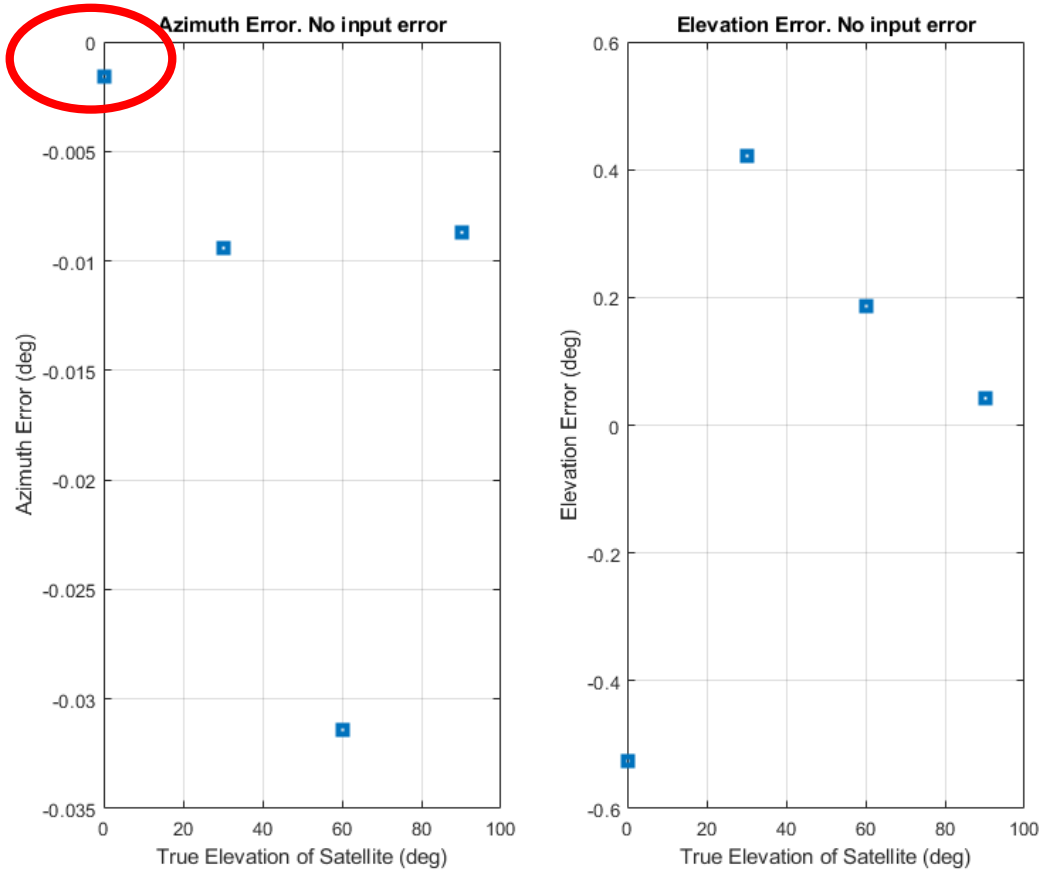
- By fitting a hyperbola instead of a line to the planes.
  - This is the most exact we can get.
  - This has yet to be tried.
- By solving with cones instead of hyperboloids.
  - A cone approximation of a hyperboloid uses the hyperboloid's asymptotes.
  - This turns out to be less accurate. See below slides.
- Changing the planes we solve on based on satellite elevation.
  - Would require we know something about satellite.
  - Could be an iterative method, find the direction, based on that direction change planes we solve on and find direction again. Continue until converged.
  - This has yet to be tried.

Using cones instead of hyperboloids does not improve results.

Hyperboloid

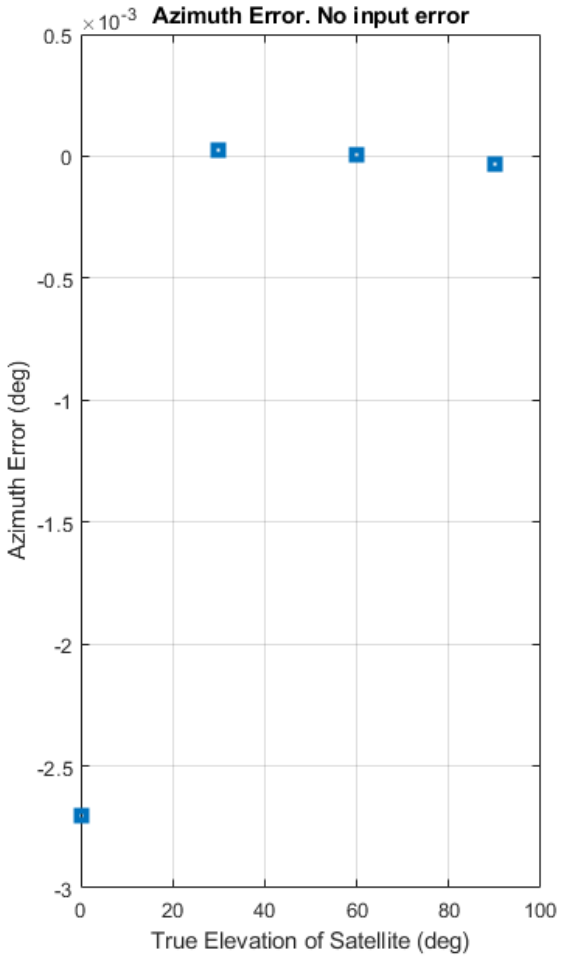


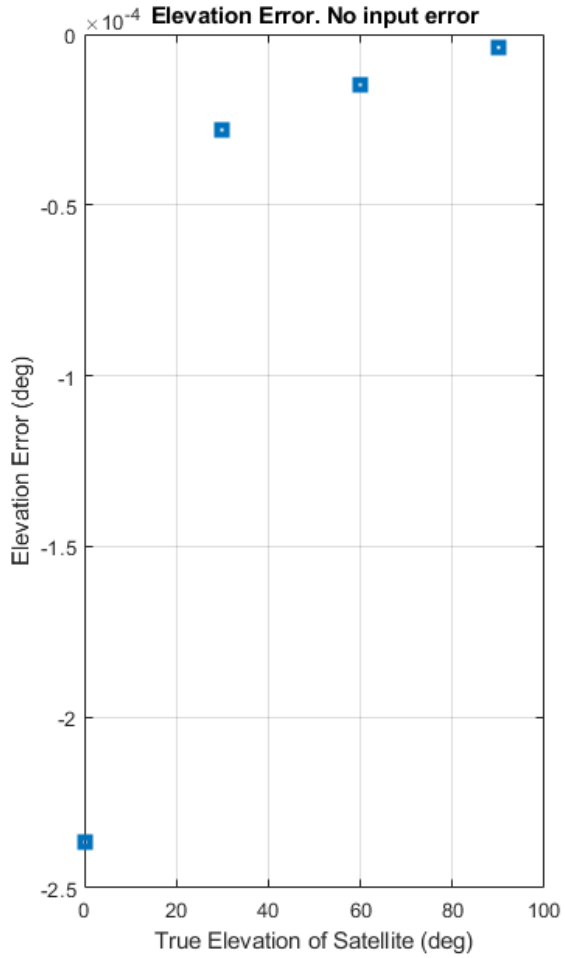
Cone



Cone causes larger azimuth error overall. It does not affect elevation error.

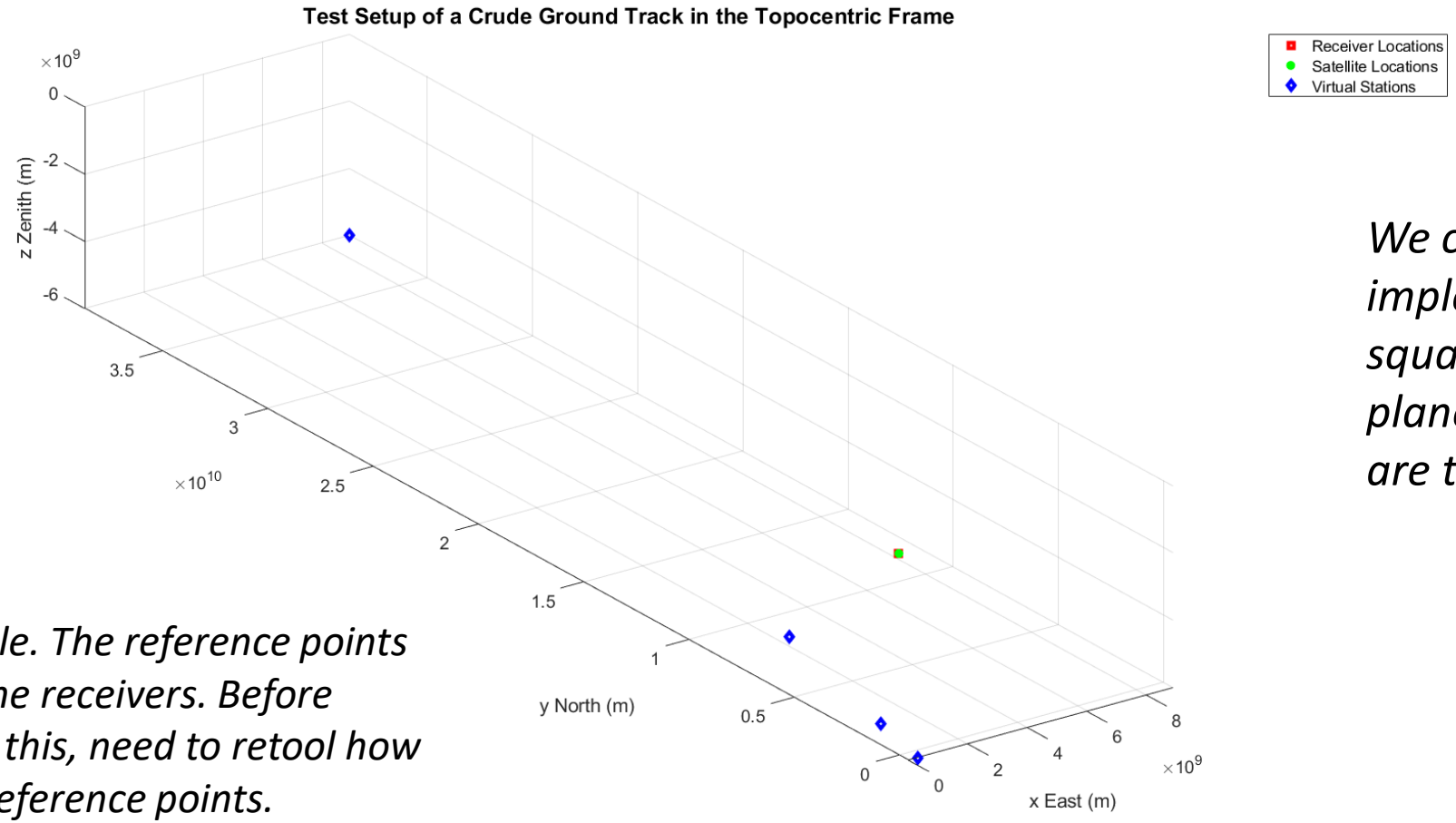
With higher planes, error decreases to ± 0.0003 degree in Elevation and ± 0.0025 degree in Azimuth.





Line Fit to:
0.5Gm, 4Gm, 12Gm planes

The downside to solving on planes Gigameters away is the reference point.

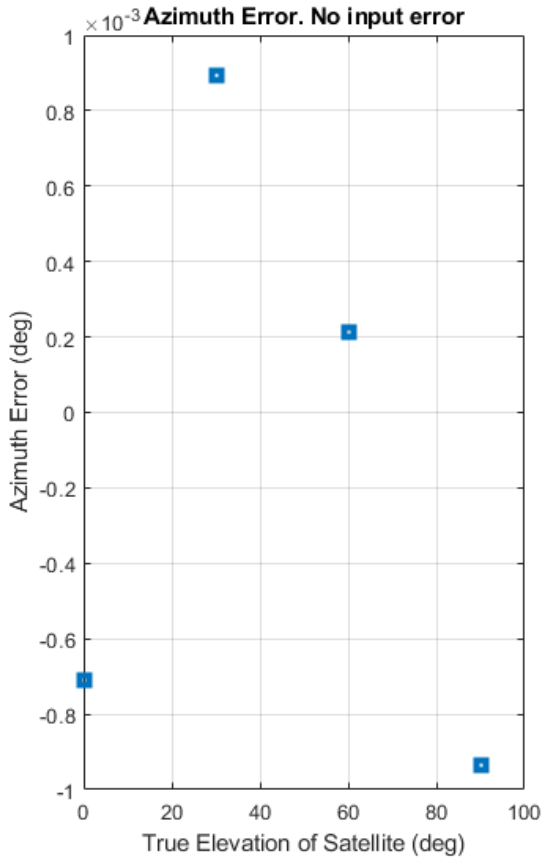


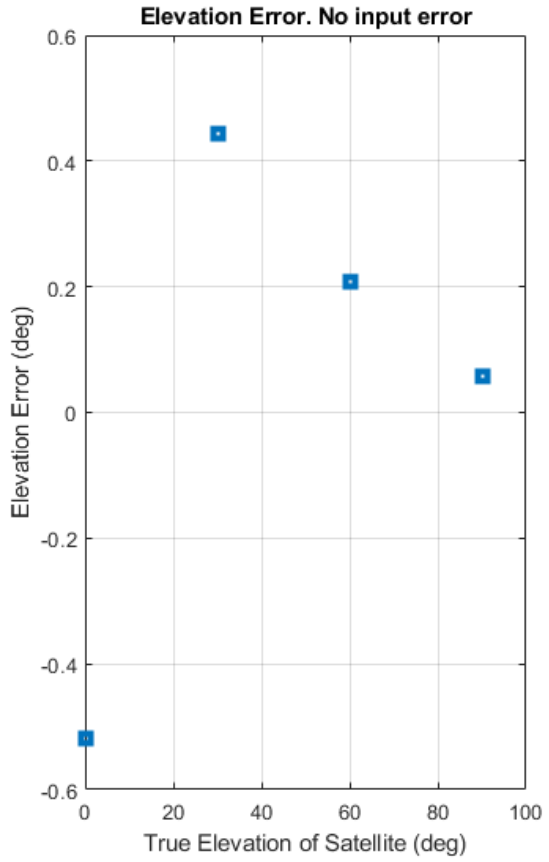
We cannot implement least squares with Gm planes – the numbers are too large.

Notice the scale. The reference points are far from the receivers. Before implementing this, need to retool how we calculate reference points.

Solving with Least Squares is significantly faster than solving symbolically.

- Error remains unchanged with no input noise.
- Solves ~60 times faster.
- 9 hours -> 9 minutes for Monte Carlo Simulation.





Least Squares uses an optimization function and minimizes a cost function.

- Least squares uses Matlab's fminunc function
  - Fminunc estimates the derivative of the cost function for faster convergence.
- Cost function:

$$cost = \sqrt{\Delta H_1^2 + \Delta H_2^2 + \Delta H_3^2}$$

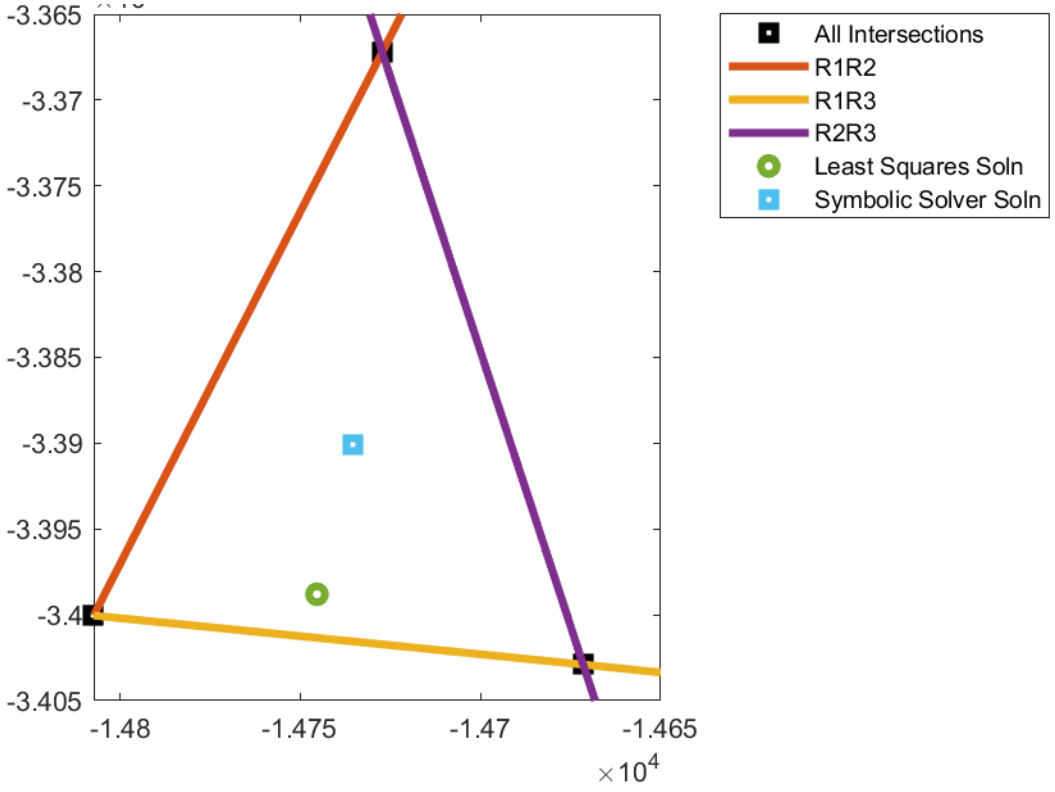
Where

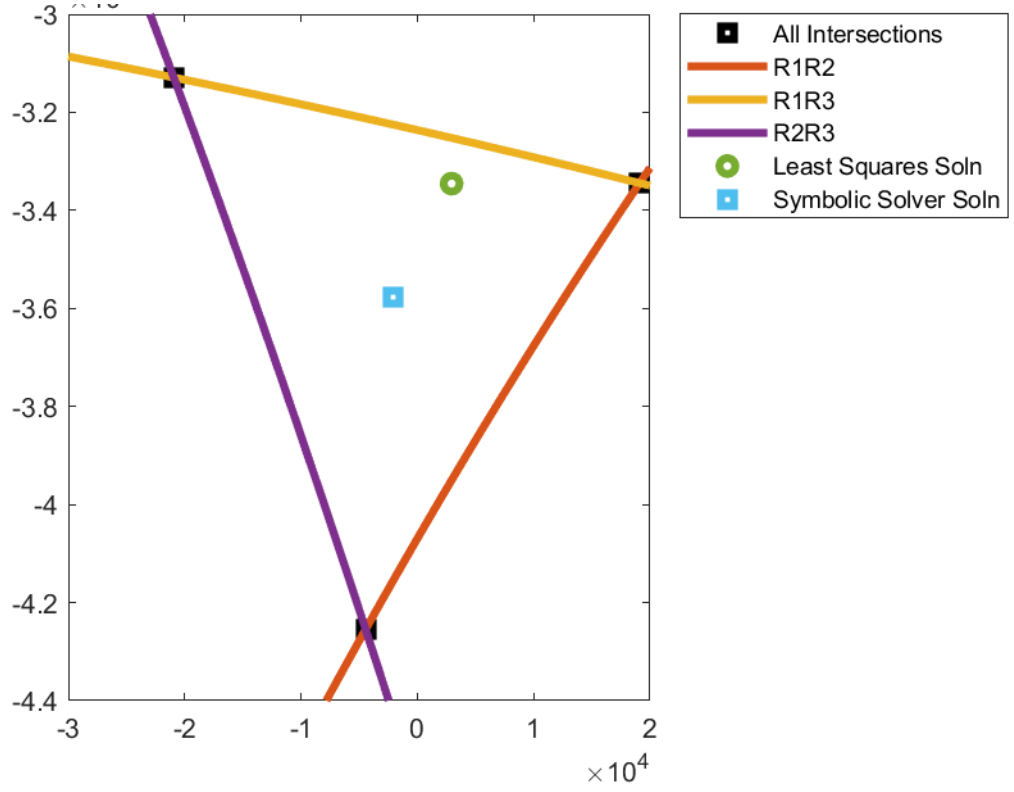
$$\Delta H_1 = a_1 \sqrt{\frac{y^2}{b_1} + \frac{z^2}{b_1} + 1} - x$$

ΔH_1 measures how close to the hyperboloid the current guess, (x, y, z), is. If the current guess is on the hyperboloid, then ΔH_1 is 0.

Therefore, this cost function represents how close the point is being on all 3 hyperboloids. Minimizing this yields the point closest to all 3 hyperboloids.

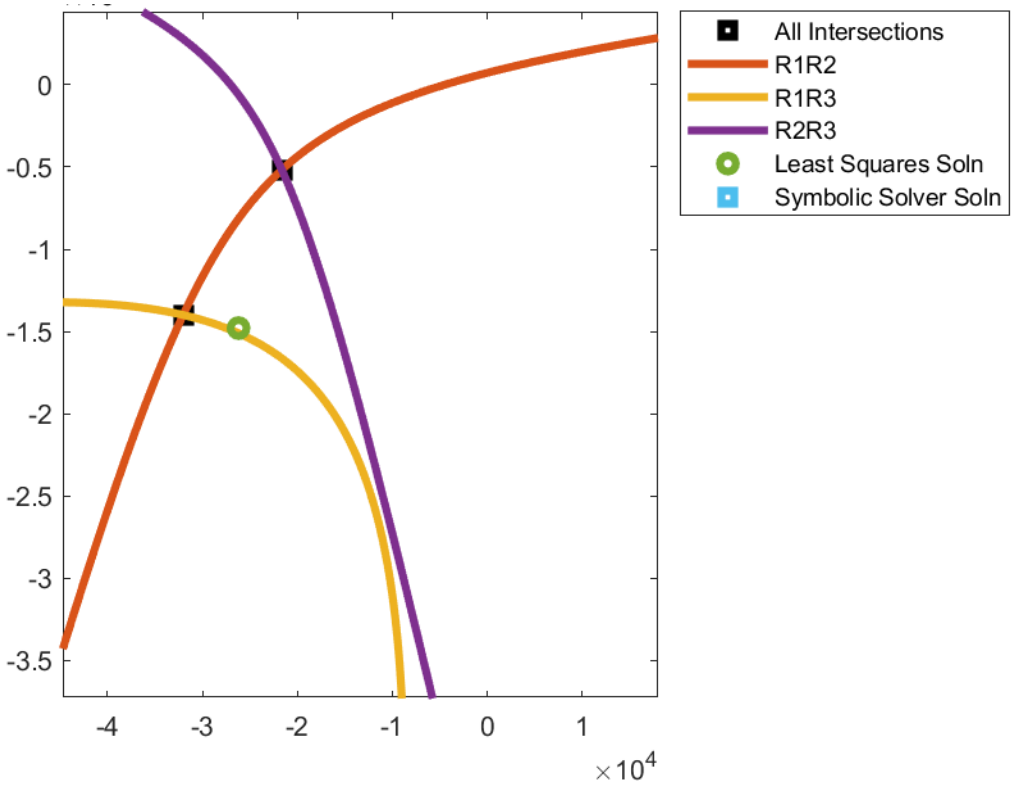
Least Squares tends to find the solution closest to the bottom leg of the isosceles triangle that TDoA creates.

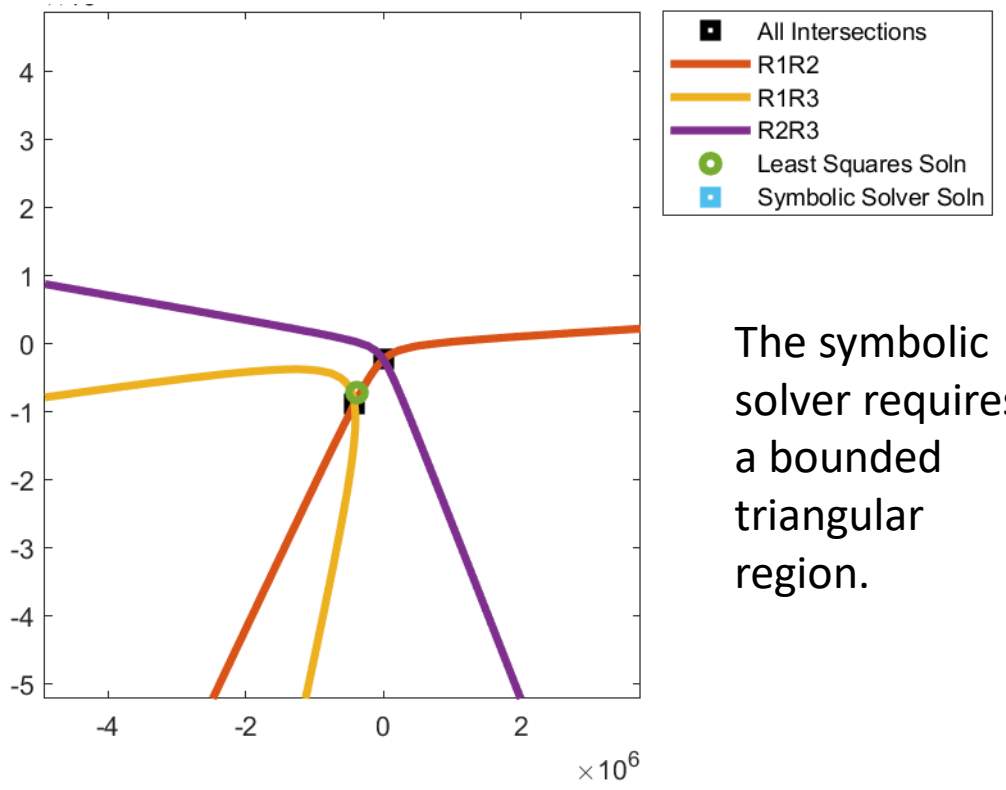




An Azimuth of 180 and Elevation of 45 is a stable point. Even 5 microsecond error does not cause crazy triangle shapes.

Least Squares can find a solution where the Symbolic Solver cannot.

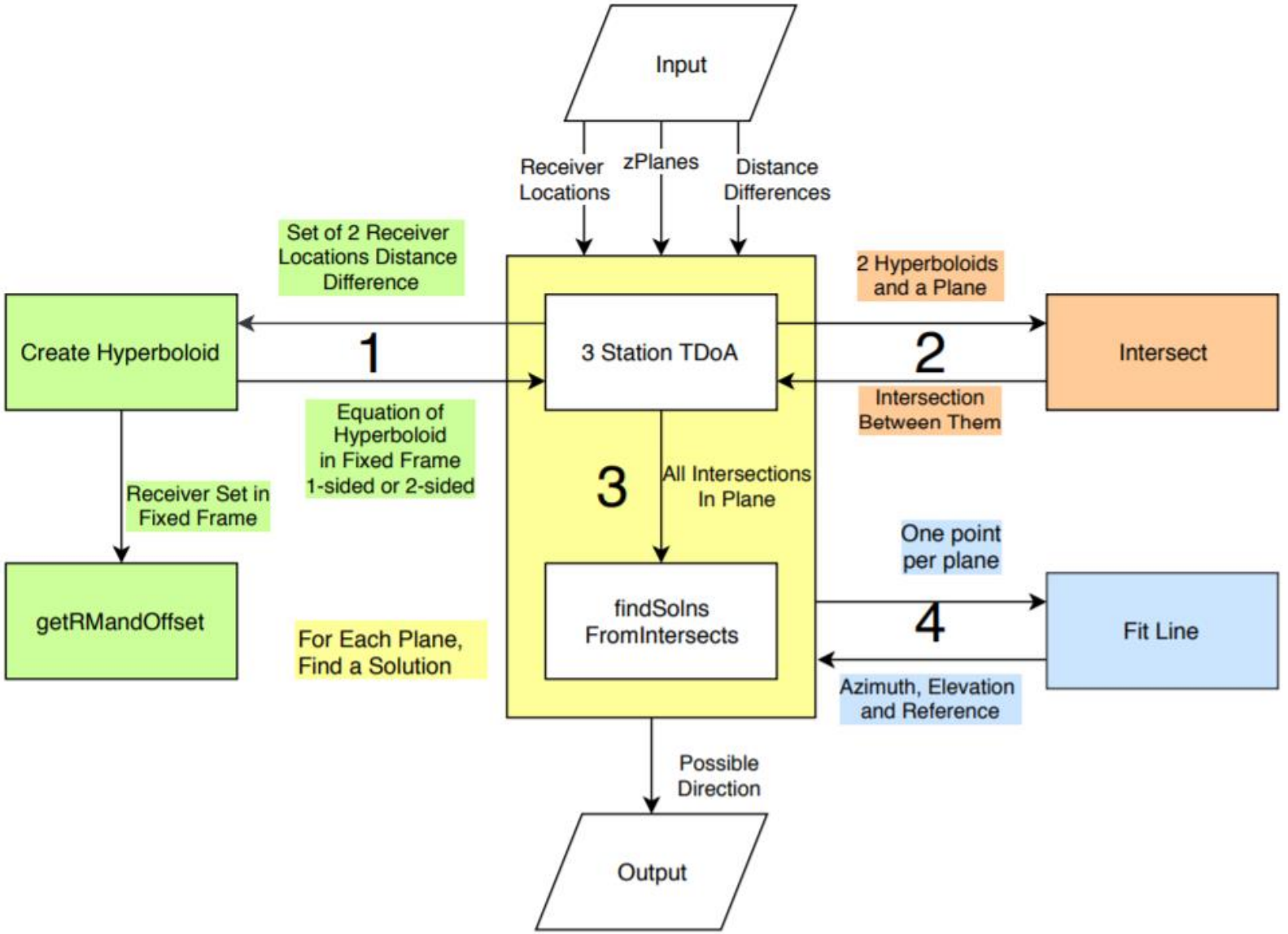




The symbolic solver requires a bounded triangular region.

An Azimuth of 180 and Elevation of 5 is an unstable point. Five microsecond error shifts the hyperbolas enough that they don't intersect. Only least squares gets a solution here.

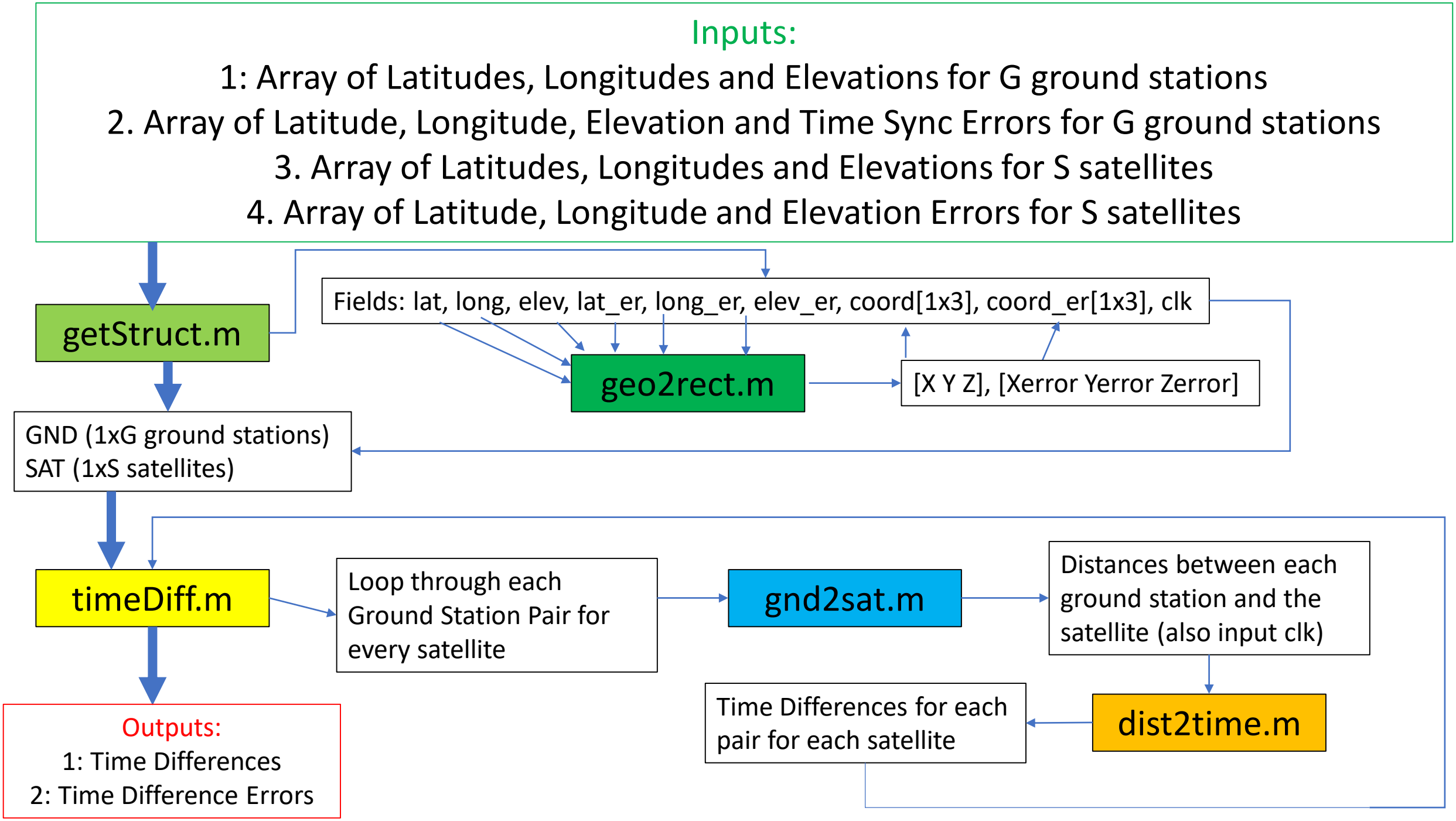
3 Station TDoA Simulator High level diagram.



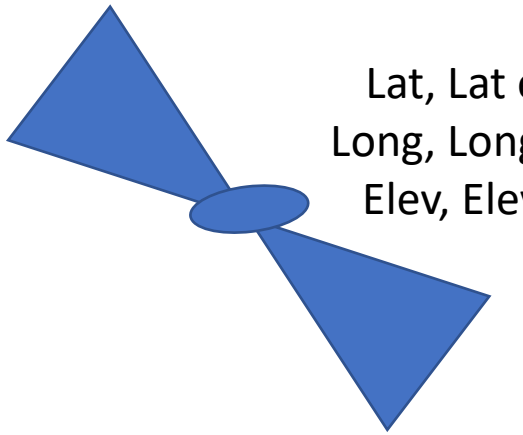
Generating Inputs to the TDOA simulator

Original by Andrew DeVries

Modified by Anthony Iannuzzi


Inputs:
1: Array of Latitudes, Longitudes and Elevations for G ground stations
2. Array of Latitude, Longitude, Elevation and Time Sync Errors for G ground stations
3. Array of Latitudes, Longitudes and Elevations for S satellites
4. Array of Latitude, Longitude and Elevation Errors for S satellites
getStruct.m
Fields: lat, long, elev, lat_er, long_er, elev_er, coord[1x3], coord_er[1x3], clk
geo2rect.m
[X Y Z], [Xerror Yerror Zerror]
GND (1xG ground stations)
SAT (1xS satellites)
timeDiff.m
Loop through each Ground Station Pair for every satellite
gnd2sat.m
Distances between each ground station and the satellite (also input clk)
dist2time.m
Time Differences for each pair for each satellite
Outputs:
1: Time Differences
2: Time Difference Errors

Inputs:



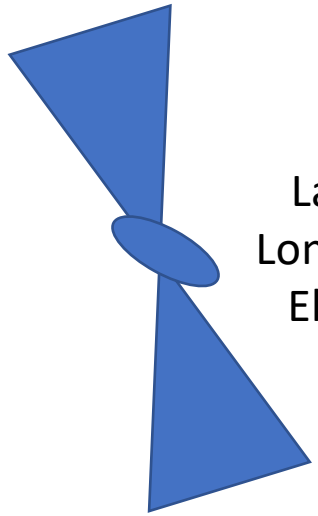
S1
Lat, Lat error [deg]
Long, Long error [deg]
Elev, Elev error [m]

A
Lat, Lat error [deg]
Long, Long error [deg]
Elev, Elev error [m]
Clk sync [s]

C
Lat, Lat error [deg]
Long, Long error [deg]
Elev, Elev error [m]
Clk sync [s]

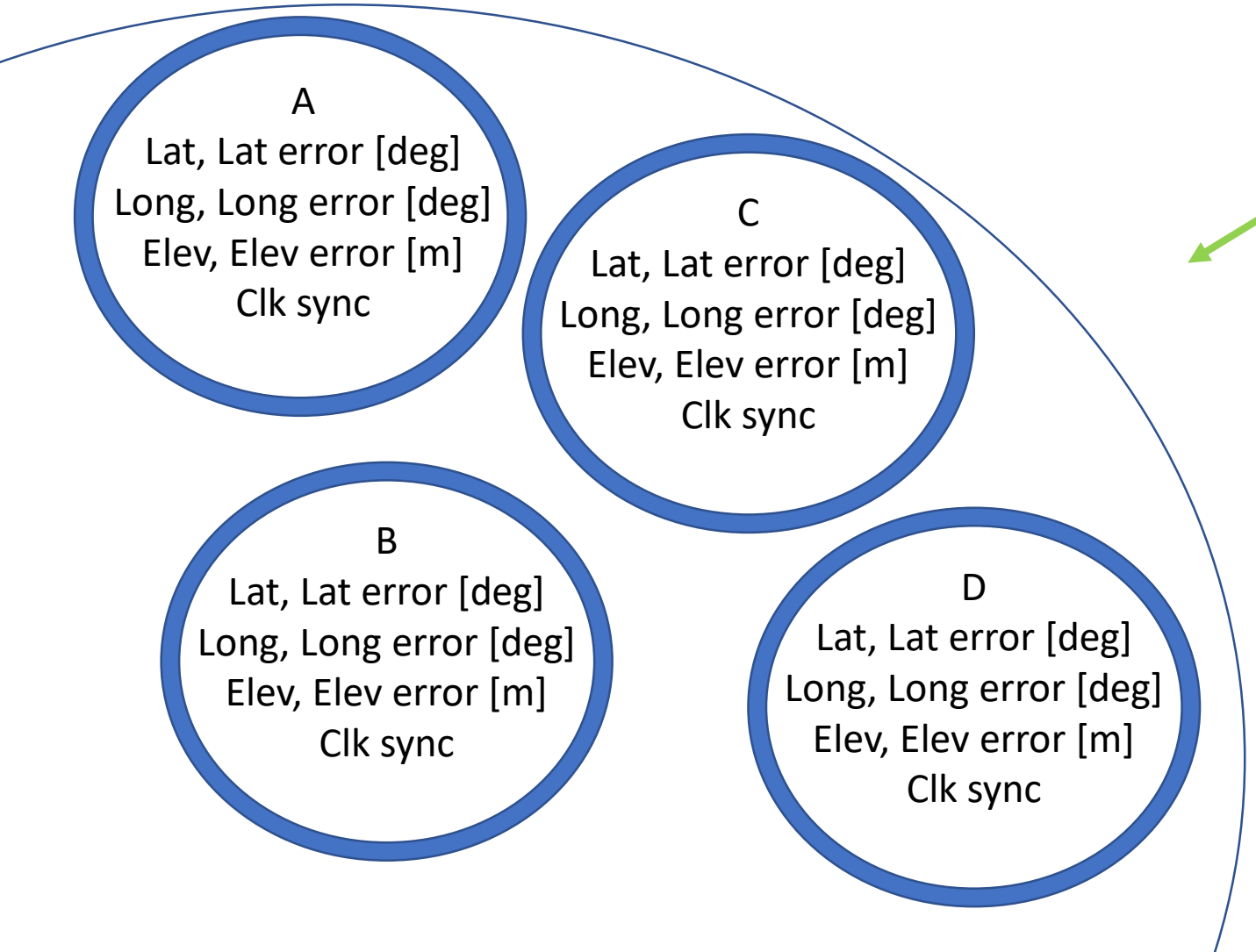
B
Lat, Lat error [deg]
Long, Long error [deg]
Elev, Elev error [m]
Clk sync [s]

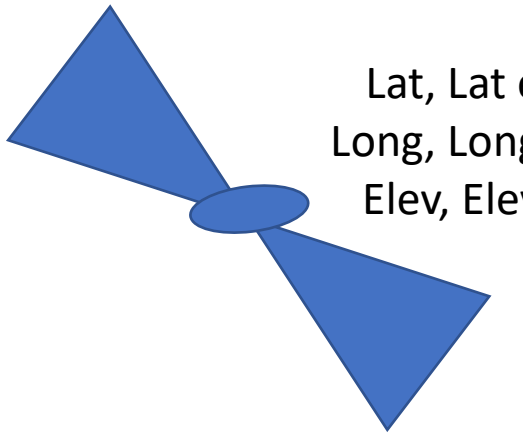
D
Lat, Lat error [deg]
Long, Long error [deg]
Elev, Elev error [m]
Clk sync [s]



S2
Lat, Lat error [deg]
Long, Long error [deg]
Elev, Elev error [m]

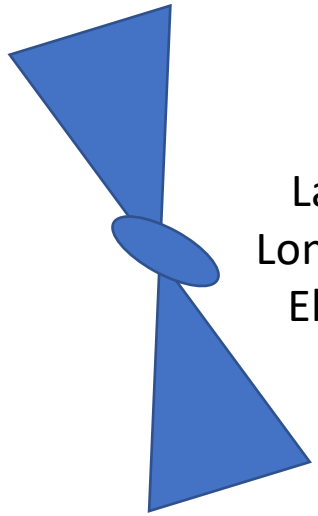
All information captured in GND structure!





S1
Lat, Lat error [deg]
Long, Long error [deg]
Elev, Elev error [m]

All information captured in SAT structure!



S2
Lat, Lat error [deg]
Long, Long error [deg]
Elev, Elev error [m]

Outputs: Time Difference Matrix

	S1	S2	…	SN
Time Difference [s] Time Difference Error [s]	A-B A-B error	A-B A-B error		A-B A-B error
	A-C A-C error	A-C A-C error		A-C A-C error
N satellites G Ground Stations	A-D A-D error	A-D A-D error		. .
	B-C B-C error	B-C B-C error		. .
	B-D B-D error	B-D B-D error		. .
	C-D C-D error	C-D C-D error		(G-1)-G (G-1)-G error

Notes:

- timeDiff can run with any size GND and SAT structures
- The coordinate conversion assumes a spherical earth
- Satellite position is absolute Lat and Long, not azimuth and elevation
- The elevations values are in meters above sea level

Changes

- Geo2rect can take any type of spheroid (really geo2ecef)
- measureInTopocentricFrame takes geodetic data turns into topocentric frame.
- getStruct records position in ECEF and Topocentric frame

Estimating the Uncertainty

By Anthony Iannuzzi

The uncertainty of some output is equal to input error times the partial derivative.

For a One-at-a-Time analysis:

Given

$$g = f(x, y, z, \dots)$$

Then,

$$\Delta g = \sqrt{\left(\frac{\delta f}{\delta x}\Delta x\right)^2 + \left(\frac{\delta f}{\delta y}\Delta y\right)^2 + \left(\frac{\delta f}{\delta z}\Delta z\right)^2 + \cdots}$$

The partial derivatives are called the ***sensitivities***. These identify which variables the output is most affected in. Larger errors in these mean larger error in the final answer.

For complicated functions, the partial derivatives can be estimated.

For a One-at-a-Time analysis:

Given

$$g = f(x, y, z, \dots)$$

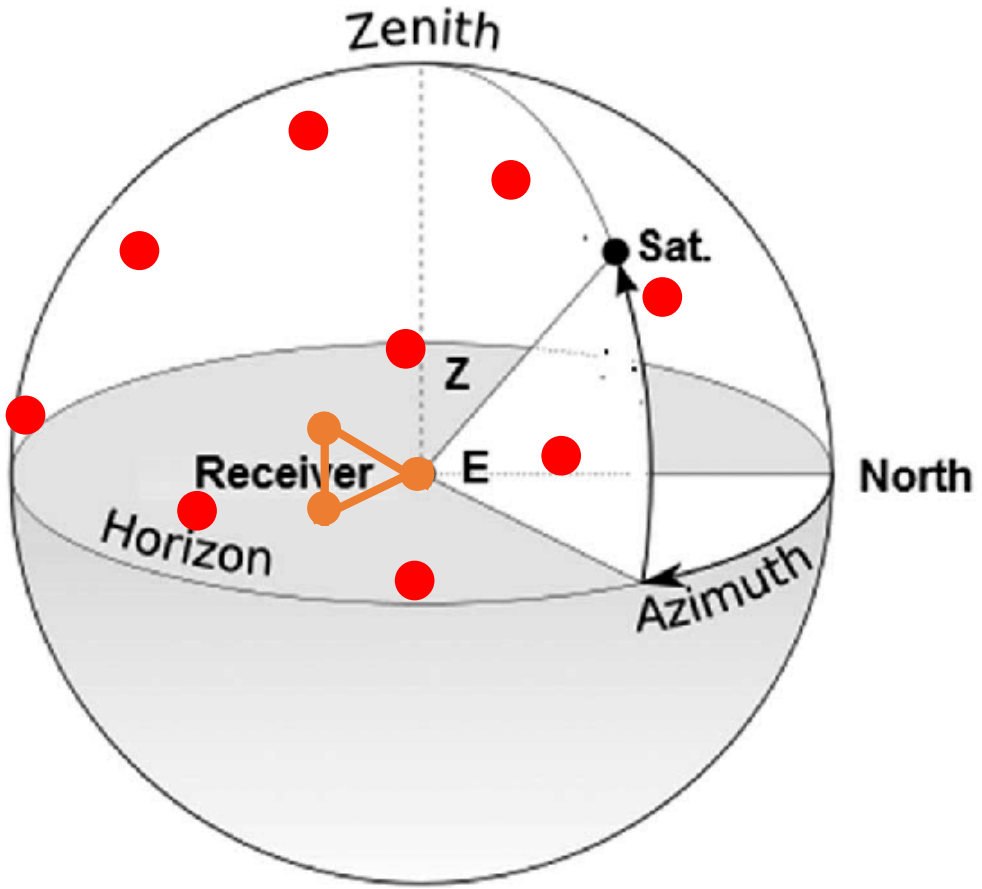
Then,

$$\frac{\delta f}{\delta x} = \frac{f(x + h, y, z, \dots) - f(x, y, z, \dots)}{h}$$

- By perturbating around the nominal inputs, we can find each partial derivative term.
- **Limitation:** the estimated partial derivatives only apply AT those nominal conditions.

For the satellite problem, the nominal input that changes is the time differences.

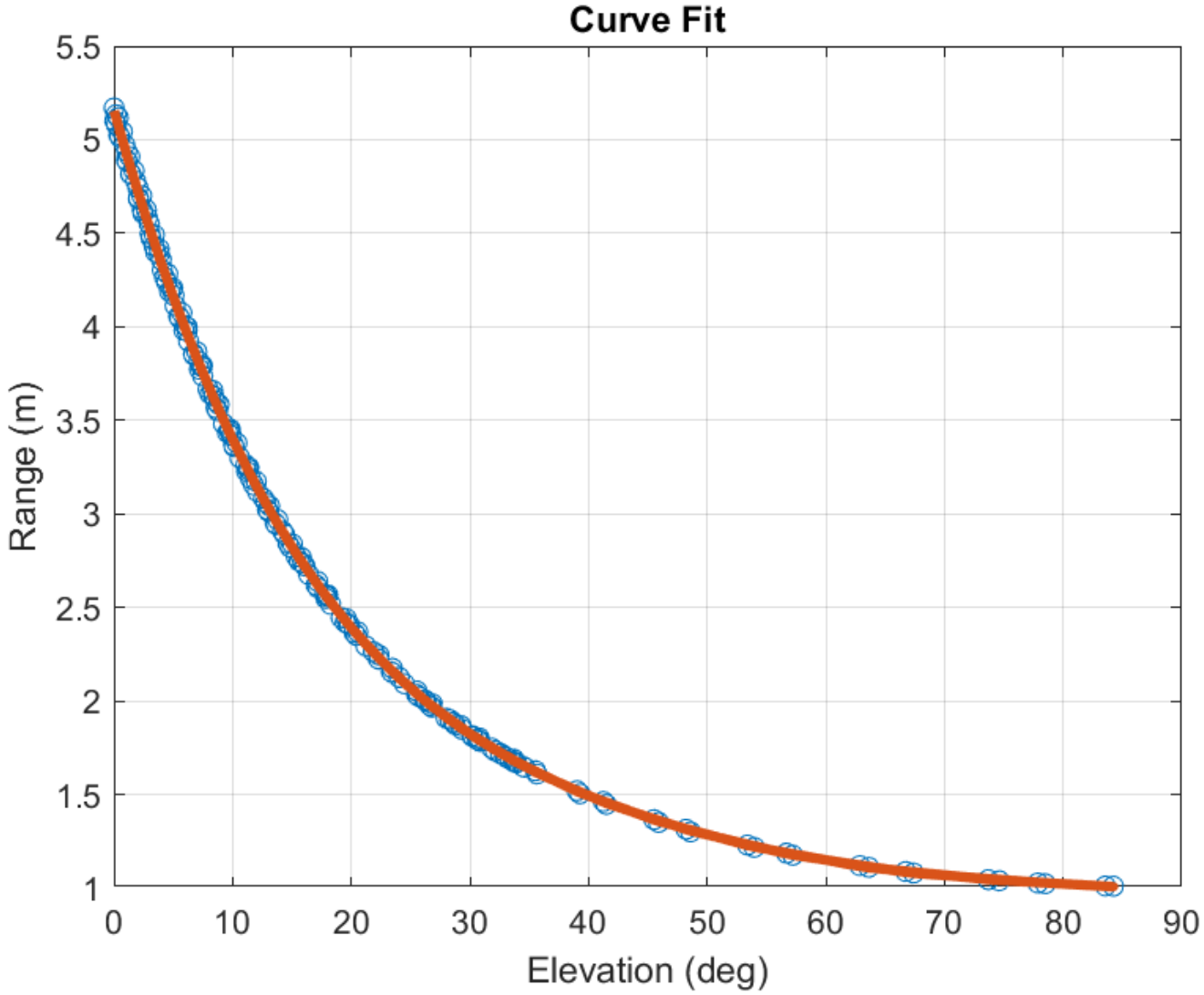
- The driver of the time differences is the satellite location in the sky.
  $Direction = f(TD, Receivers)$
- Receivers have error associated with them, but there nominal values never change.
- The Time Differences change depending on the satellite actual azimuth, elevation, and range.
  - We will estimate sensitivities for a "net" of the horizon.



https://www.researchgate.net/figure/Satellite-Azimuth-and-elevation-angle_fig1_334197029

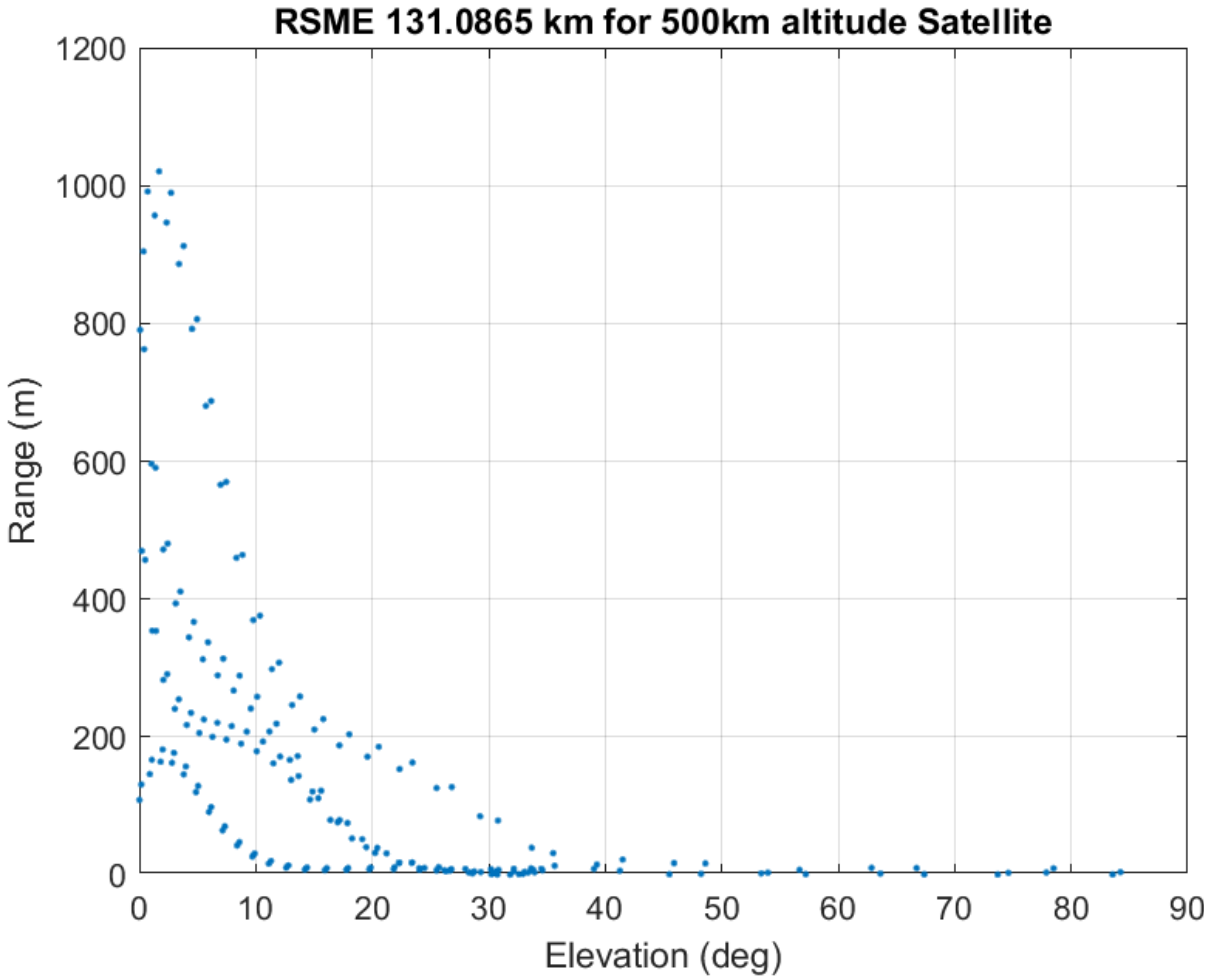
The net uses a curve fit for range as a function of elevation.

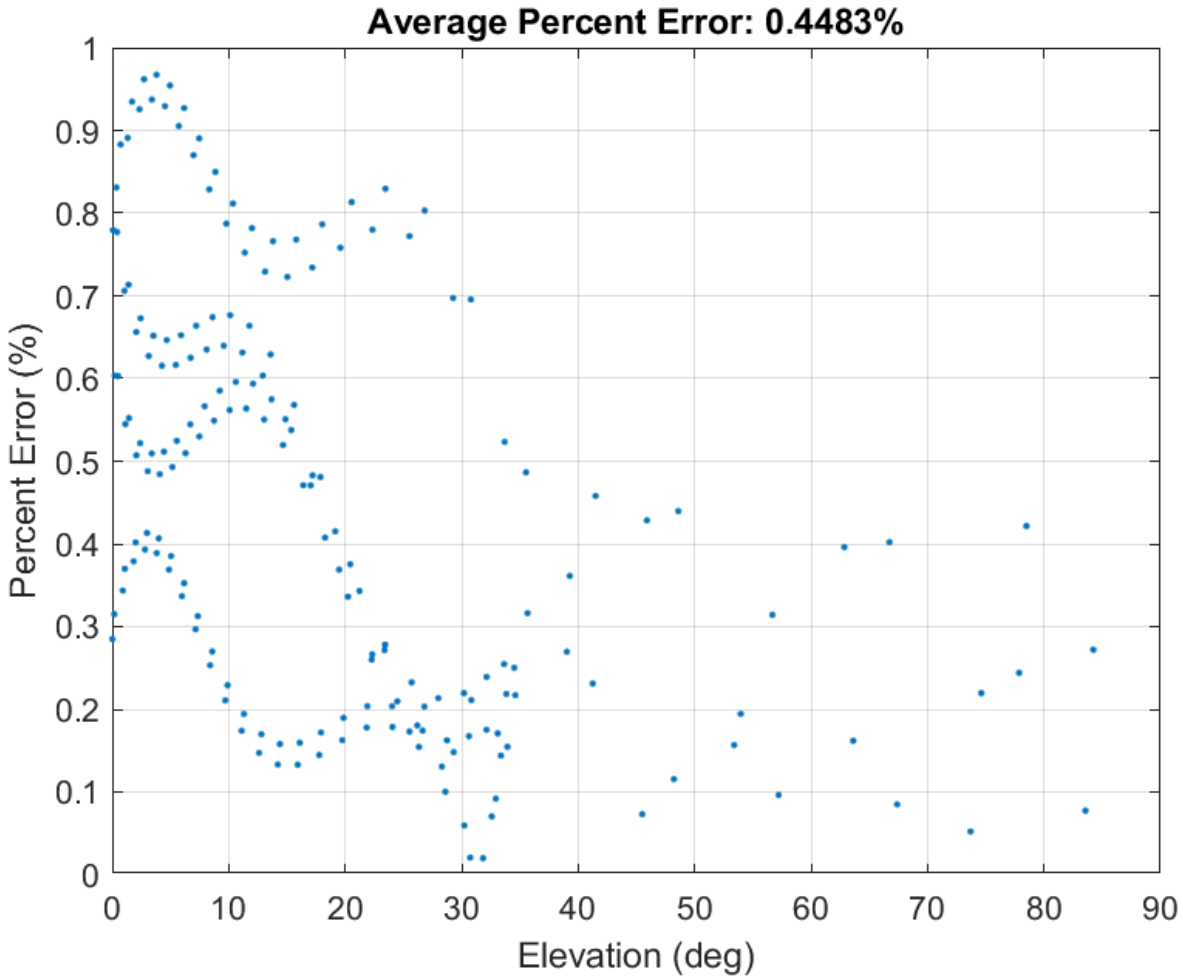
We fit a 6th order polynomial to data from 4 different ground tracks.



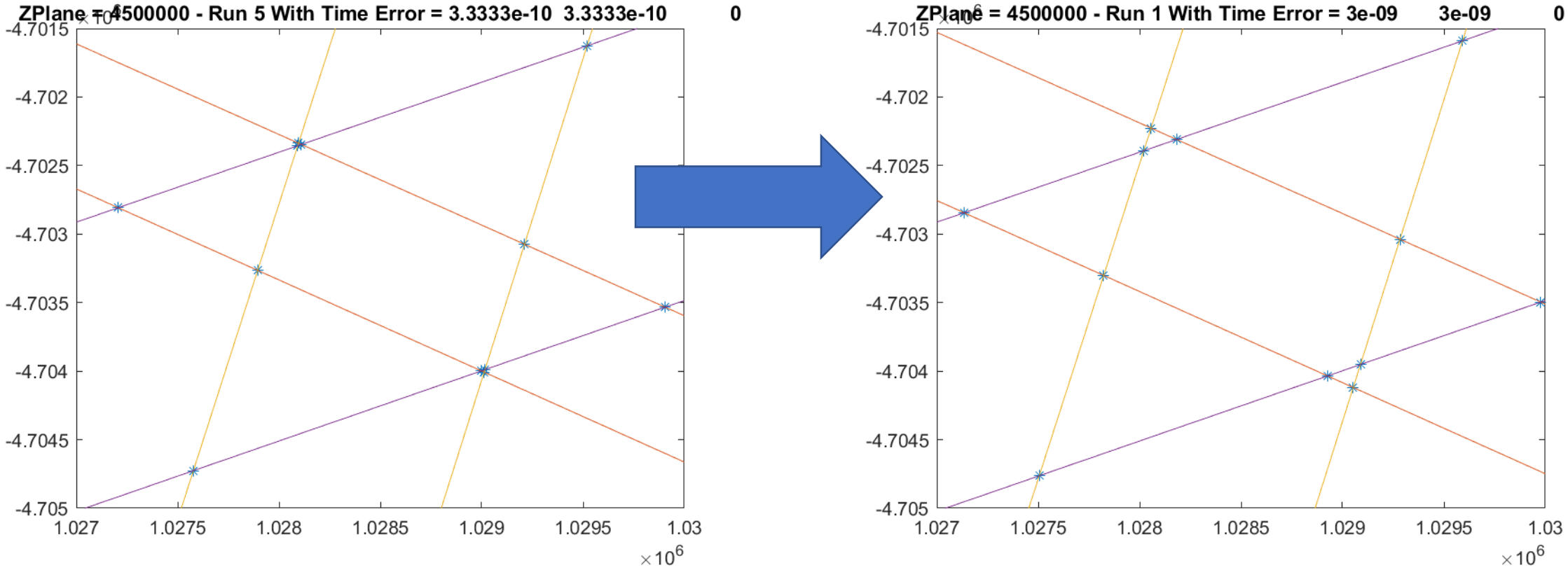
We can further improve these results by using more ground tracks. But this is unnecessary. TDoA is not very sensitive to satellite range.

The curve fit error is small, but could be validated with more ground tracks.



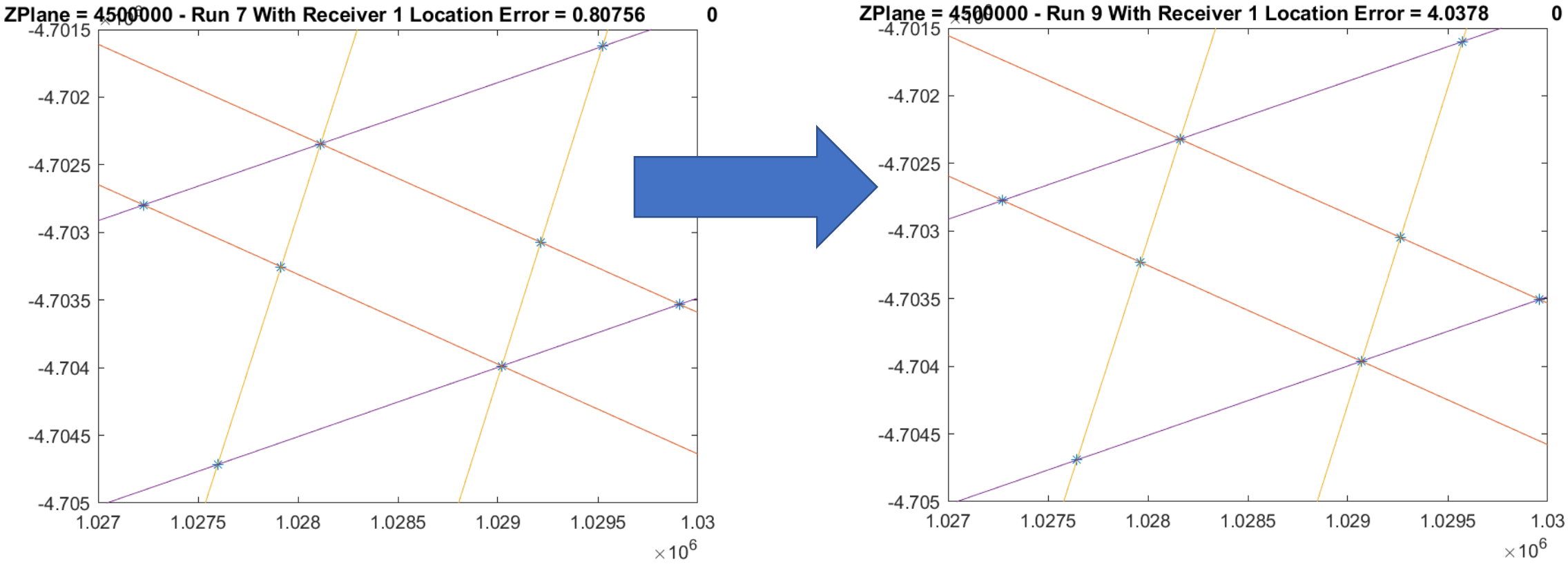


Errors in time tend to increase cluster spacing, increasing the area of potential locations. **Precision Loss**



But first… How do the hyperbolas change when there are errors in the system?

Errors in location shift 2 of the hyperbolas, but the cluster size is constant. **Accuracy Loss.**



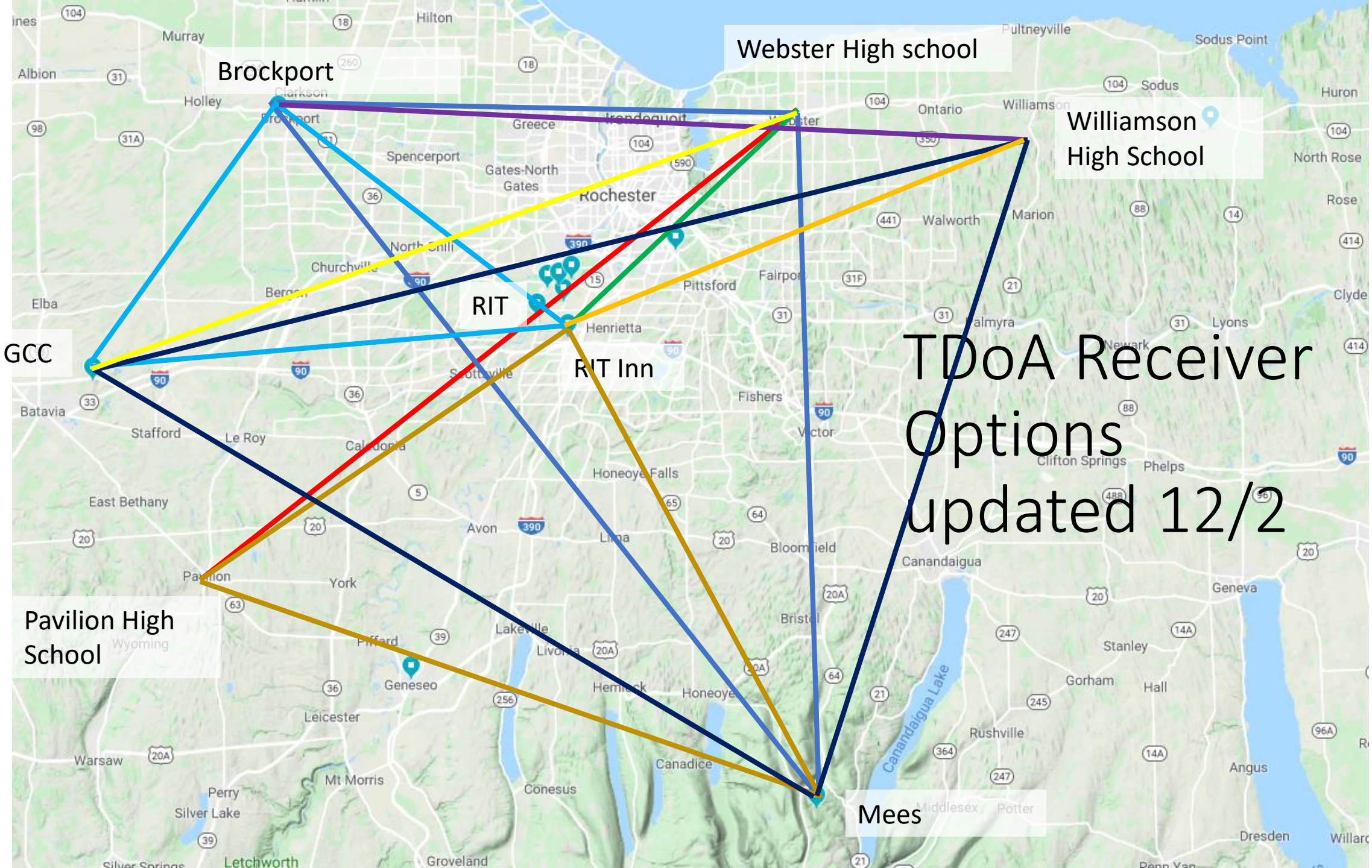
But first… How do the hyperbolas change when there are errors in the system?

We consider 10 triangles for the sensitivity analysis.

#	Location 1	Location 2	Location 3	Mean Distance (km)	Largest Angle (deg)
1	Mees	Brockport	Webster	56	96
2	Mees	Webster	Pavilion	56	64
3	RIT Inn	Institute Hall	Ellingson	3	94
4	Mees	Pavilion	RIT Inn	44	84
5	RIT Inn	Brockport	Webster	31	94
6	Mees	Williamson	Brockport	64	72
7	RIT Inn	GCC	Brockport	32	89
8	Mees	GCC	Williamson	70	77
9	Mees	Webster	GCC	61	74
10	Mees	RIT Inn	Williamson	49	88

Approach	Receiver Location Uncertainty	Time Difference Uncertainty
One-at-a-time	9m	100 ns
Monte Carlo	9m	100 ns (validation)
Monte Carlo	9m	5 us

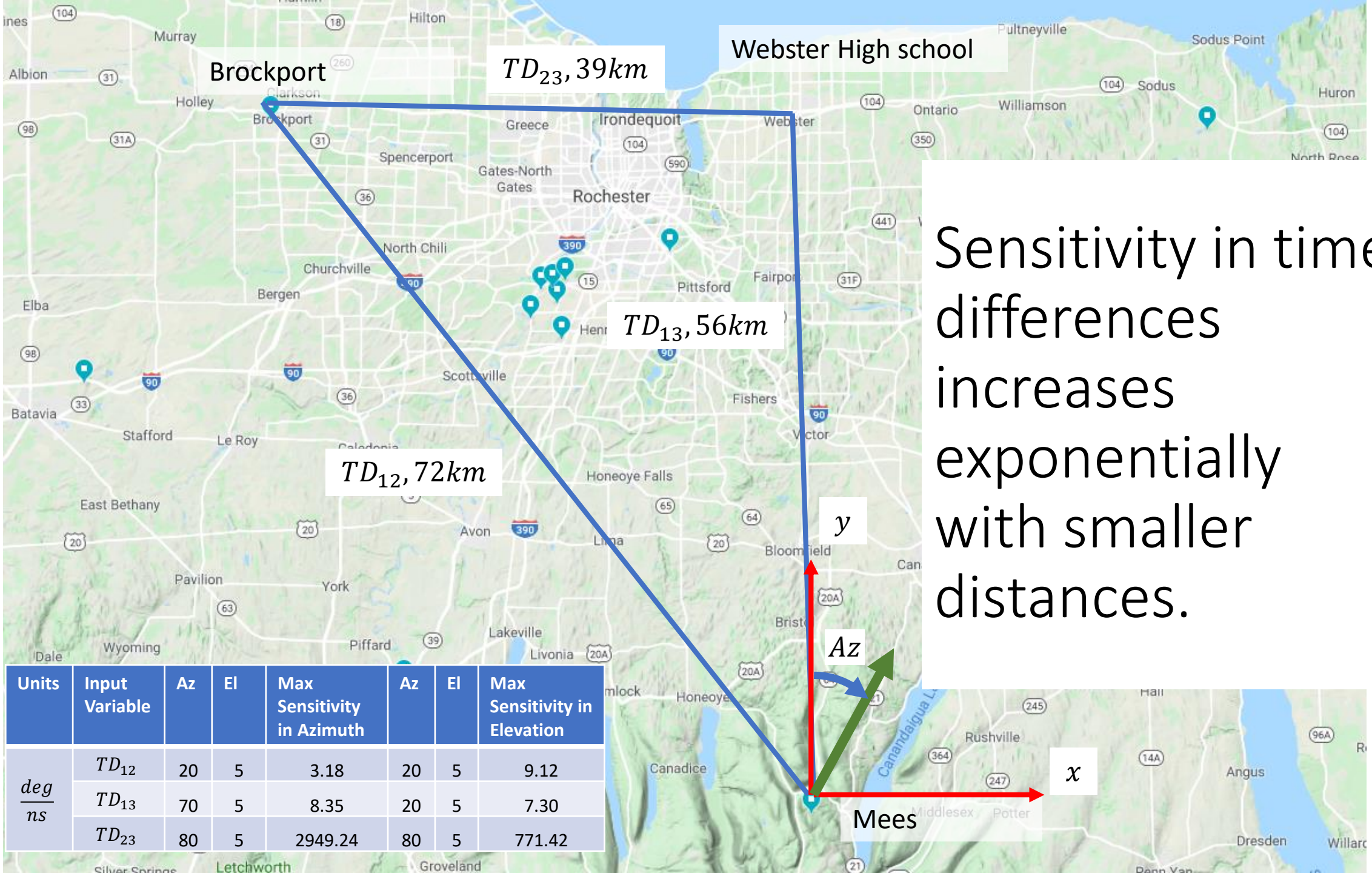
We run one-at-a-time (OAAT) for lower uncertainty. For higher uncertainty, we switch to Monte Carlo. We run one Monte Carlo for small uncertainty to compare its results to OAAT.


Webster High school
Brockport
Williamson High School
GCC
RIT
RIT Inn
Pavilion High School
Mees
TDoA Receiver Options updated 12/2

OAAT enables us to see which inputs are most important where.

Azimuth and Elevation wrt to Location 1 where this input variable is most sensitive. Data taken from triangle 1.

Units	Input Variable	Azimuth	Elevation	Max Sensitivity in Azimuth	Azimuth	Elevation	Max Sensitivity in Elevation
$\frac{deg}{km}$	$R1\ (x)$	160	5	1.95	160	5	1.60
	$R1\ (y)$	10	5	1.25	310	85	0.00
	$R1\ (z)$	310	85	1.22	160	5	0.25
	$R2\ (x)$	150	5	0.00	80	5	0.00
	$R2\ (y)$	160	5	0.00	300	5	0.00
	$R2\ (z)$	290	85	0.00	80	5	0.00
	$R3\ (x)$	160	5	0.00	160	5	0.00
	$R3\ (y)$	160	5	0.00	10	5	0.00
	$R3\ (z)$	320	85	0.00	280	5	0.00
$\frac{deg}{ns}$	TD_{12}	20	5	3.18	20	5	9.12
	TD_{13}	70	5	8.35	20	5	7.30
	TD_{23}	80	5	2949.24	80	5	771.42



Sensitivity in time differences increases exponentially with smaller distances.

Units	Input Variable	Az	El	Max Sensitivity in Azimuth	Az	El	Max Sensitivity in Elevation
$\frac{deg}{ns}$	TD_{12}	20	5	3.18	20	5	9.12
	TD_{13}	70	5	8.35	20	5	7.30
	TD_{23}	80	5	2949.24	80	5	771.42

OAAT enables us to see which inputs are most important where.

Azimuth and Elevation wrt to Location 1 where this input variable is most sensitive. Data taken from triangle 2.

Units	Input Variable	Azimuth	Elevation	Max Sensitivity in Azimuth	Azimuth	Elevation	Max Sensitivity in Elevation
$\frac{deg}{km}$	$R1\ (x)$	120	5	6.28	120	5	9.63
	$R1\ (y)$	20	5	1.21	250	5	1.87
	$R1\ (z)$	310	85	0.57	120	5	0.48
	$R2\ (x)$	250	5	0.01	120	5	0.00
	$R2\ (y)$	170	5	0.00	20	5	0.00
	$R2\ (z)$	320	85	0.01	170	5	0.00
	$R3\ (x)$	120	5	0.00	250	5	0.01
	$R3\ (y)$	120	5	0.00	120	5	0.00
	$R3\ (z)$	0	90	0.00	120	5	0.00
$\frac{deg}{ns}$	TD_{12}	120	5	0.00	270	5	259.22
	TD_{13}	260	5	2062.36	160	5	1969.74
	TD_{23}	120	5	0.00	270	5	257.74

Units	Input Variable	Az	El	Max Sensitivity in Azimuth	Az	El	Max Sensitivity in Elevation
$\frac{deg}{ns}$	TD_{12}	120	5	0.00	270	5	259.22
	TD_{13}	260	5	2062.36	160	5	1969.74
	TD_{23}	120	5	0.00	270	5	257.74

Webster High school

$TD_{23}, 58km$

$TD_{12}, 56km$

$TD_{13}, 53km$

Pavilion High School

y

Az

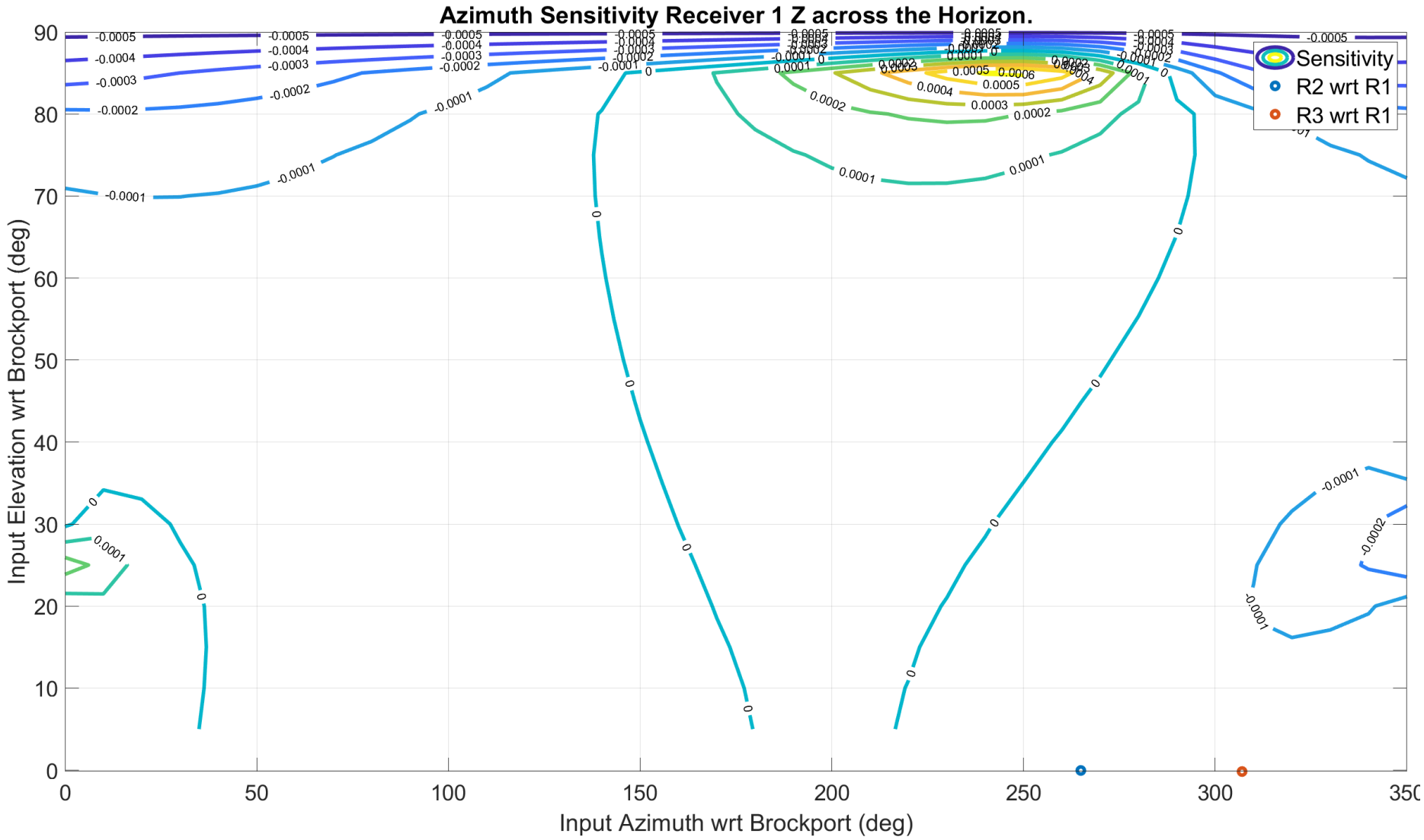
x

Mees

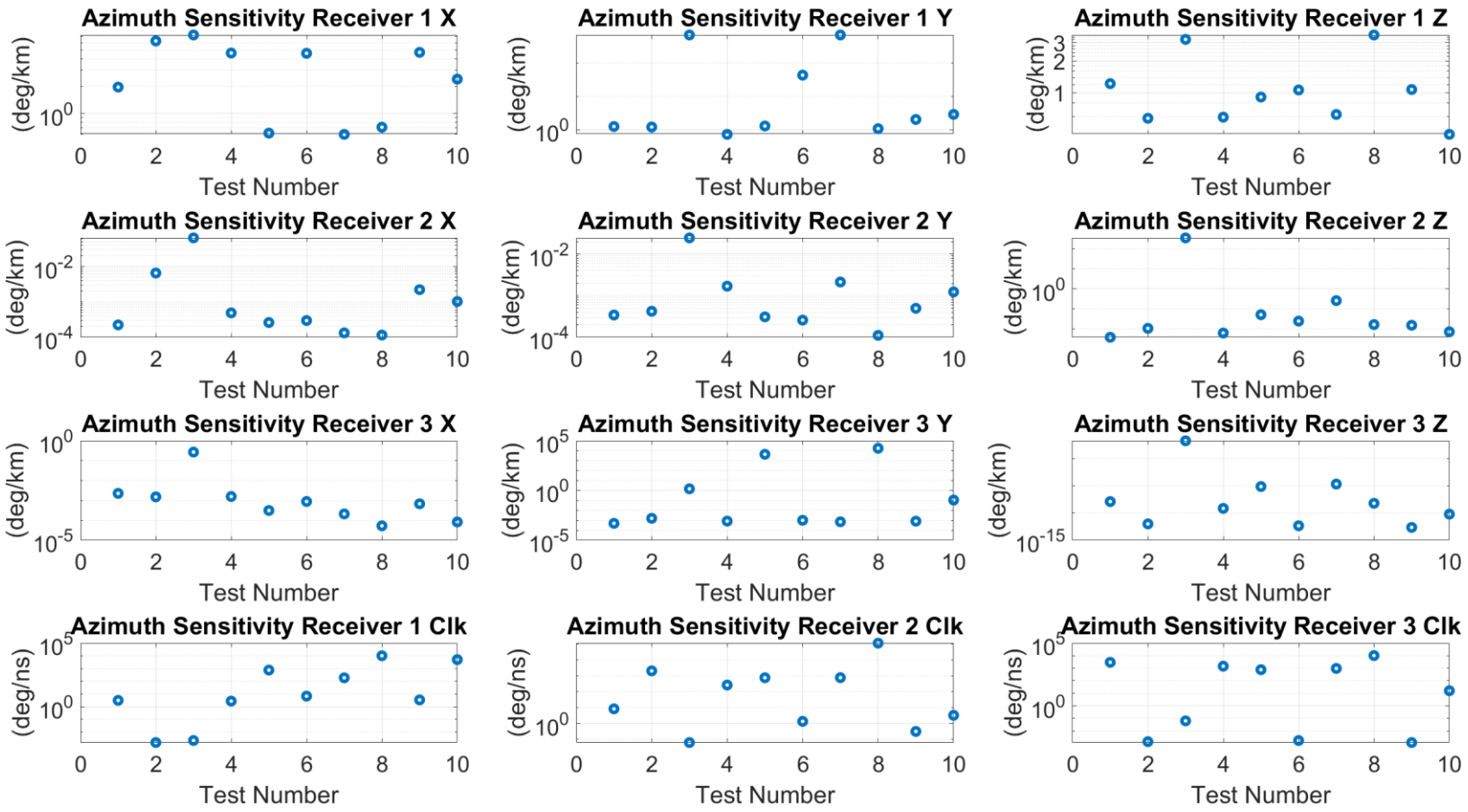
Sensitivity in time differences increases exponentially with smaller distances, <u>but is relative to the other time differences.</u>

At high elevations, uncertainty in the z axis of receiver 1 has high sensitivity.

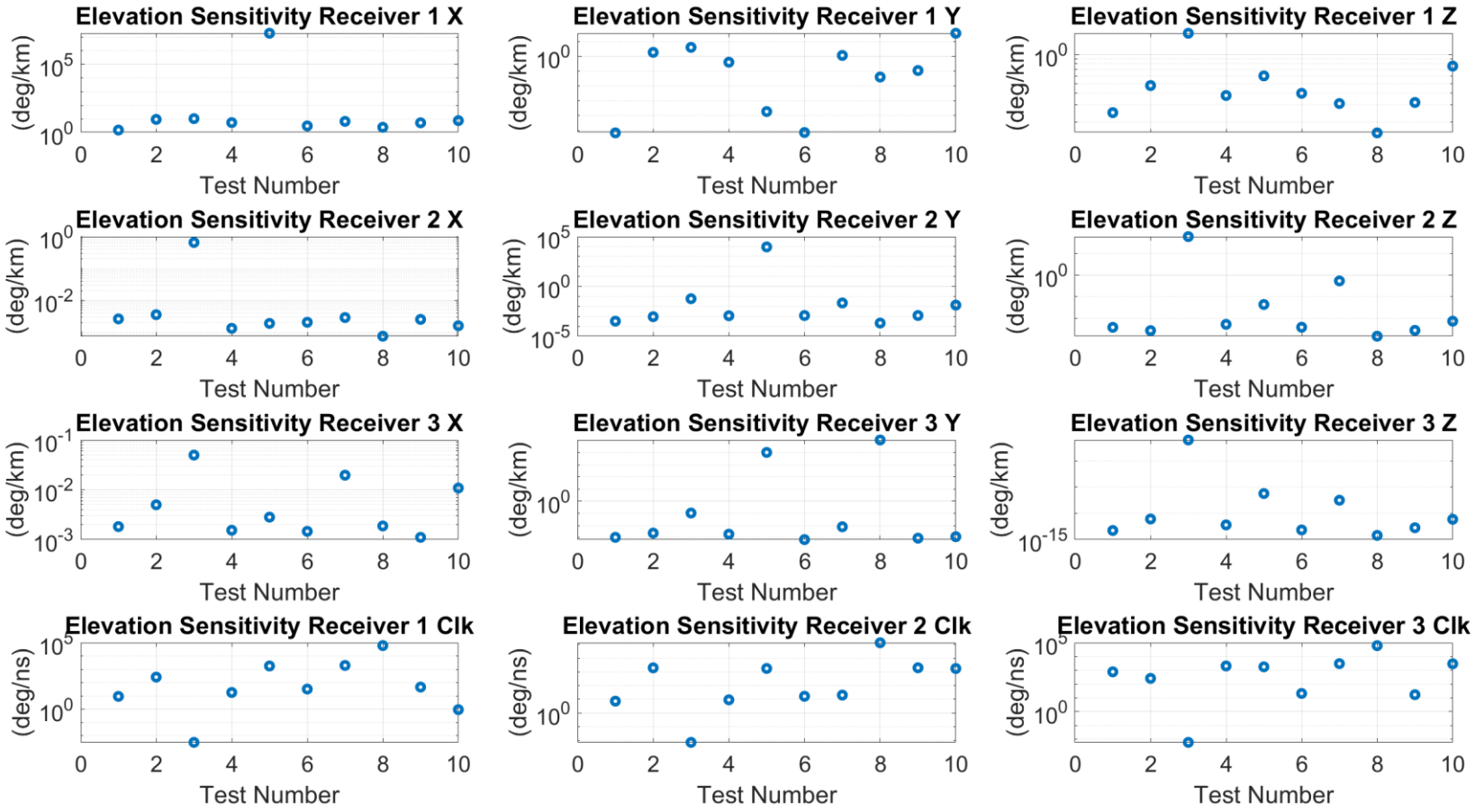
This is a contour plot, representing the sensitivity to RIT Inn for triangle 7.



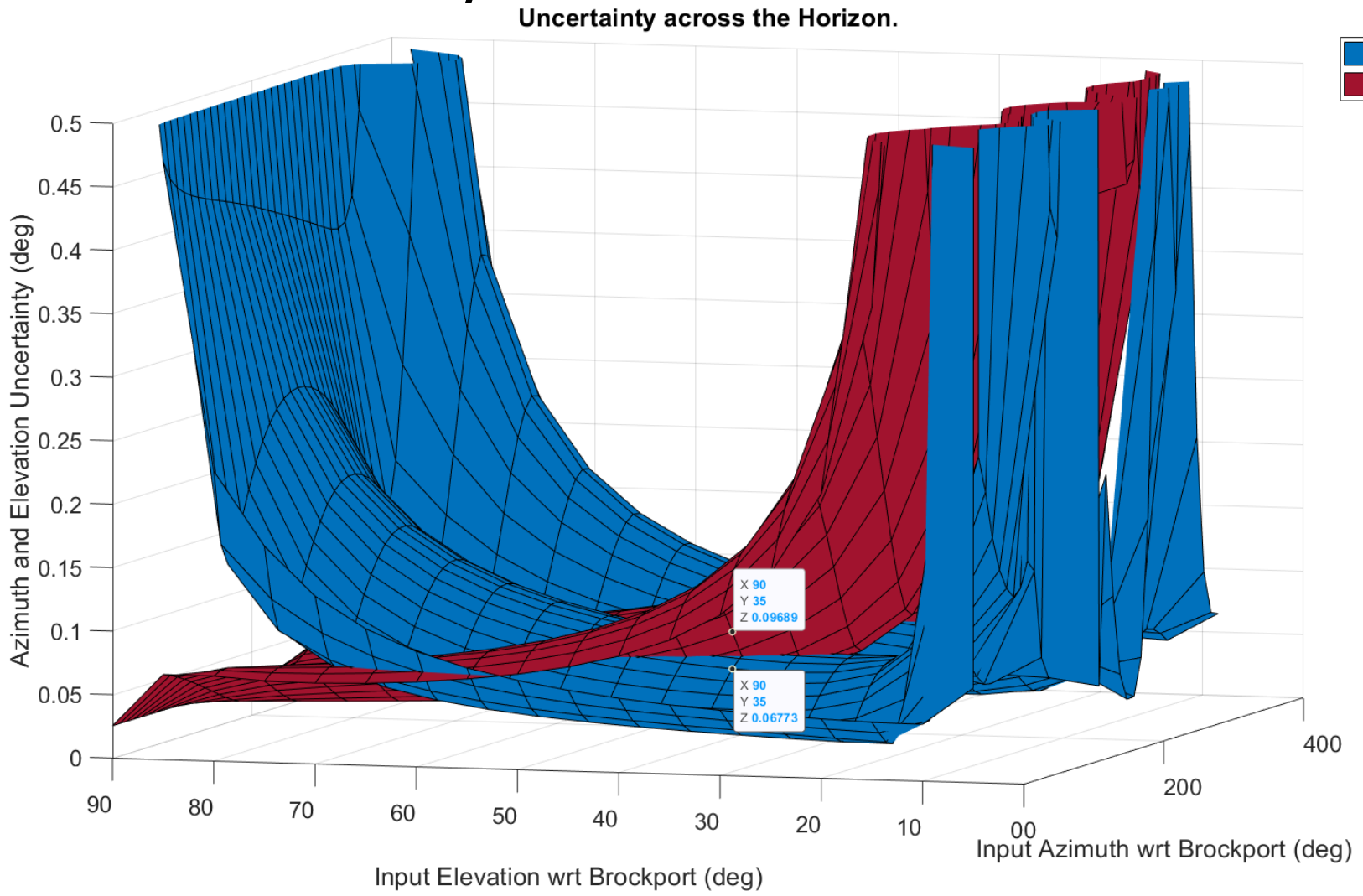
These max sensitivities change magnitude significantly with different triangles. Azimuth.



These max sensitivities change magnitude significantly with different triangles. Elevation



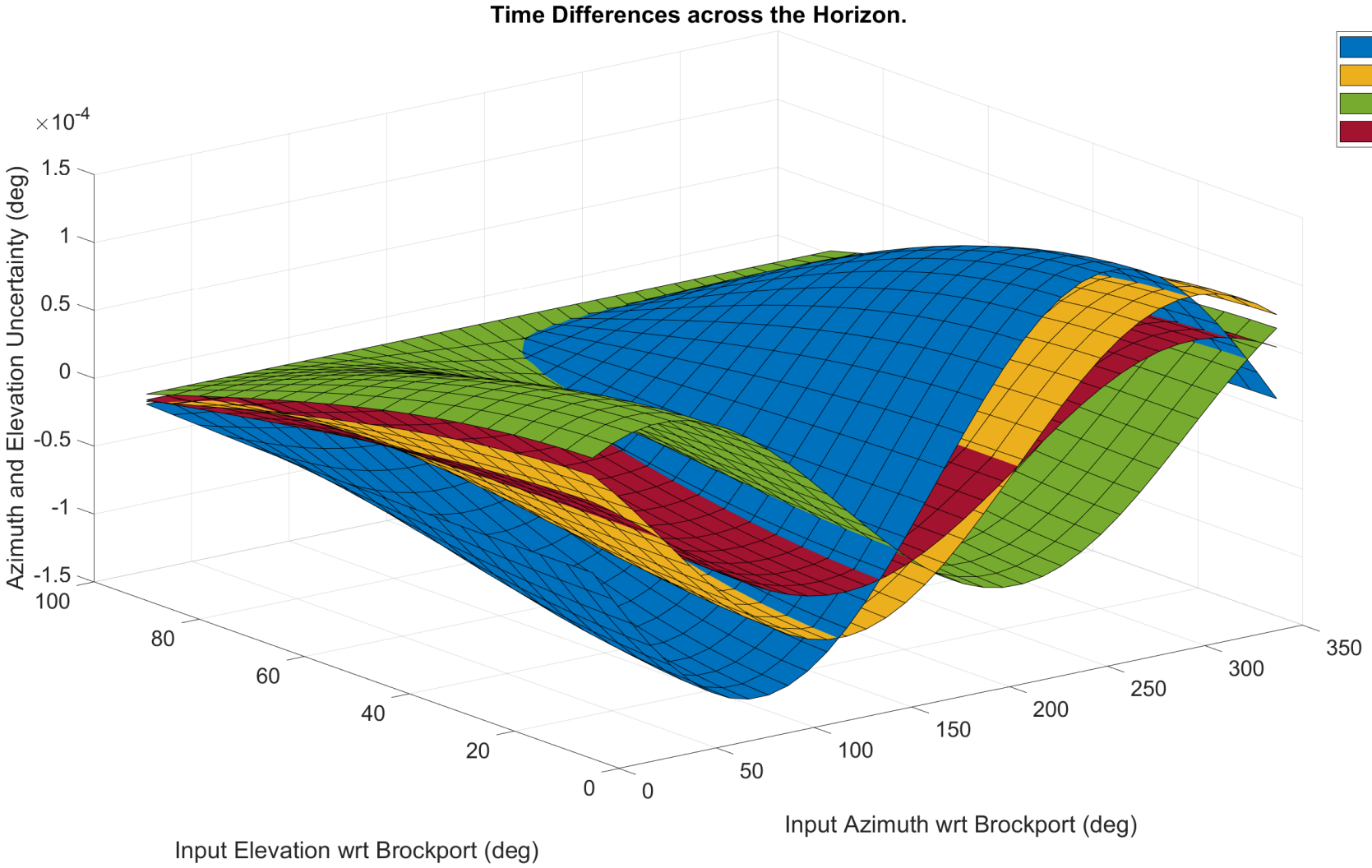
When we apply RSME, we can estimate the uncertainty based on OAAT.



While the exact location varies, the overall shape seen in this plot remains constant for all triangles.

- Elevation is highly uncertain in 2 low azimuth locations.
- In general all elevations grow less certain with lower elevations..
- Azimuth is highly uncertain in 2 high elevation places.
- In general all azimuths grow less certain with higher elevations.

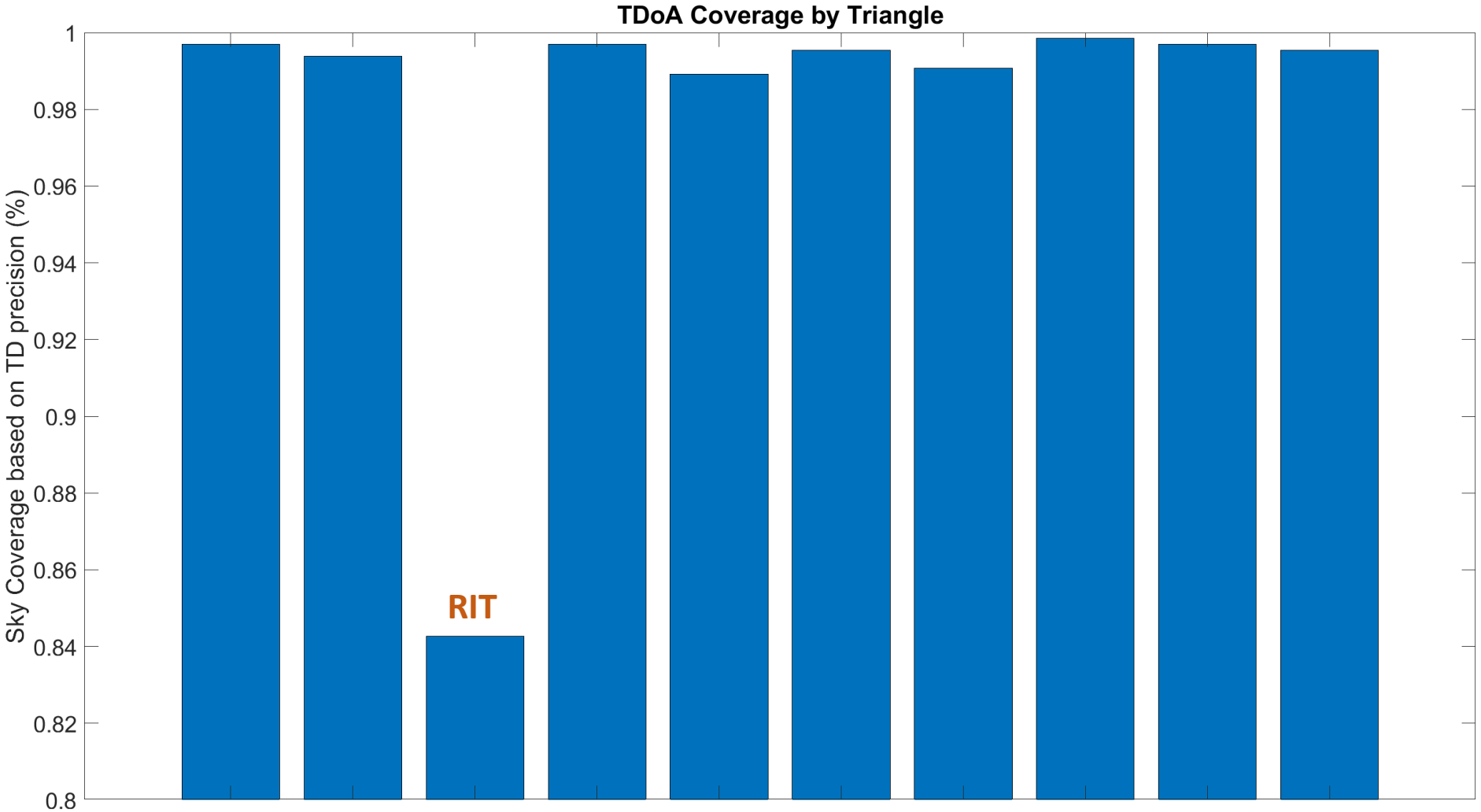
The Time Differences tend to have a dampened sinusoid appearance.



By collecting time differences during simulation, we can determine locations where we have blind spots due a discrete sampling.

i.e. we can't accuractly collect 0 time difference with 100 ns sampling rate.

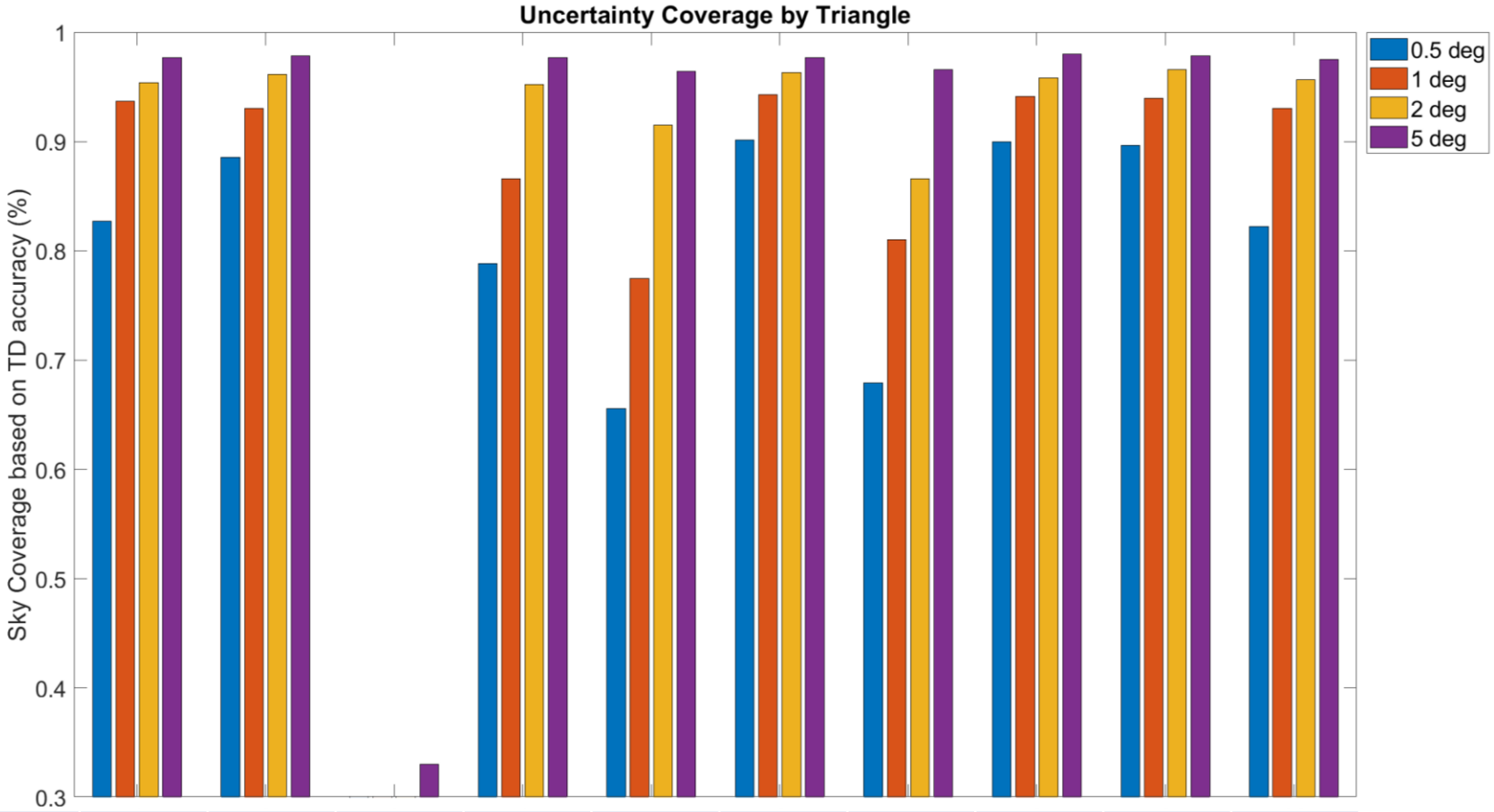
All triangles tend to have high coverage based on the minimum time difference we can measure.



Notice we start at 80%

Test	1	2	3	4	5	6	7	8	9	10
Distance (km)	56	56	3	44	31	64	32	70	61	49
Max Angle (deg)	96	64	94	84	94	72	89	77	74	88

Triangles with larger mean distances have better coverage based on uncertainty.

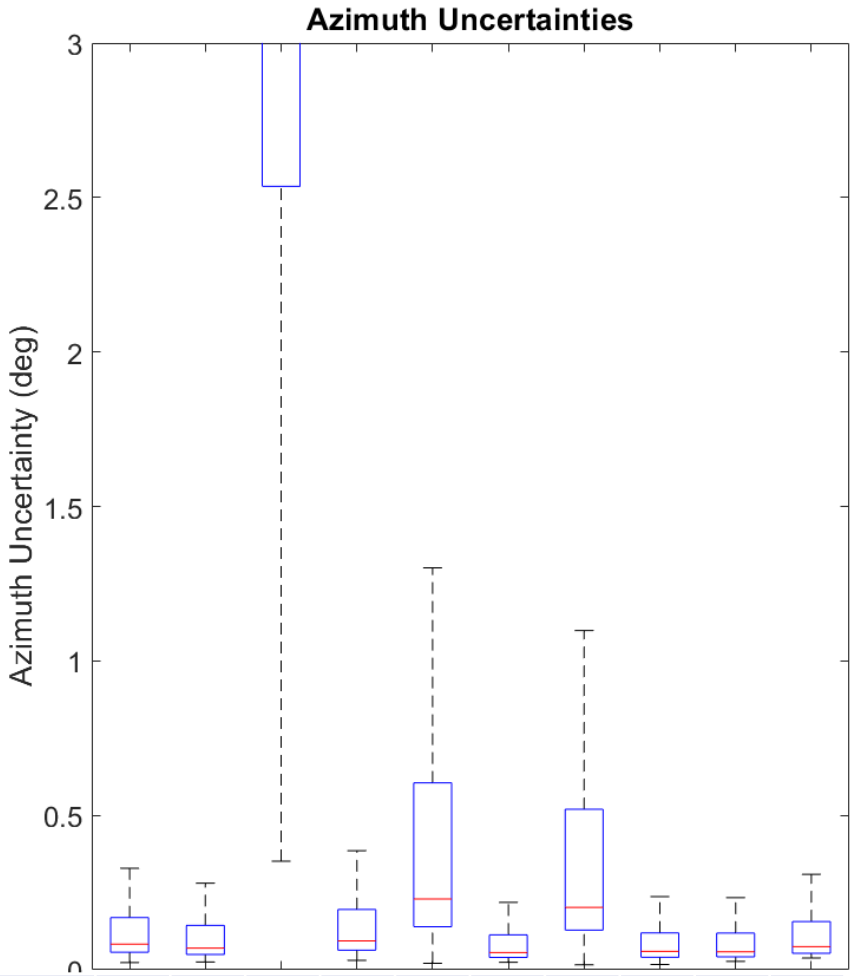


Notice we start at 30%

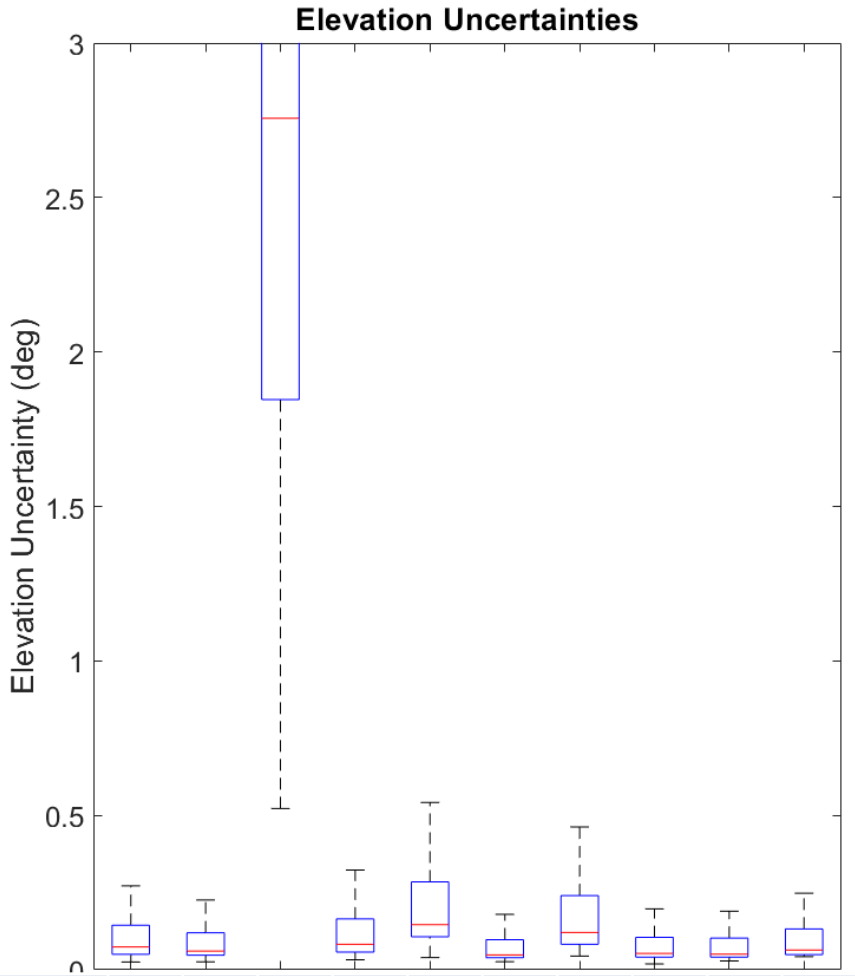
Test	1	2	3	4	5	6	7	8	9	10
Distance (km)	56	56	3	44	31	64	32	70	61	49
Max Angle (deg)	96	64	94	84	94	72	89	77	74	88

We count number of points in the net with uncertainty lower than 0.5, 1, 2, and 5 degrees for BOTH azimuth and elevation.

We can compare the spread of uncertainties for each triangle.

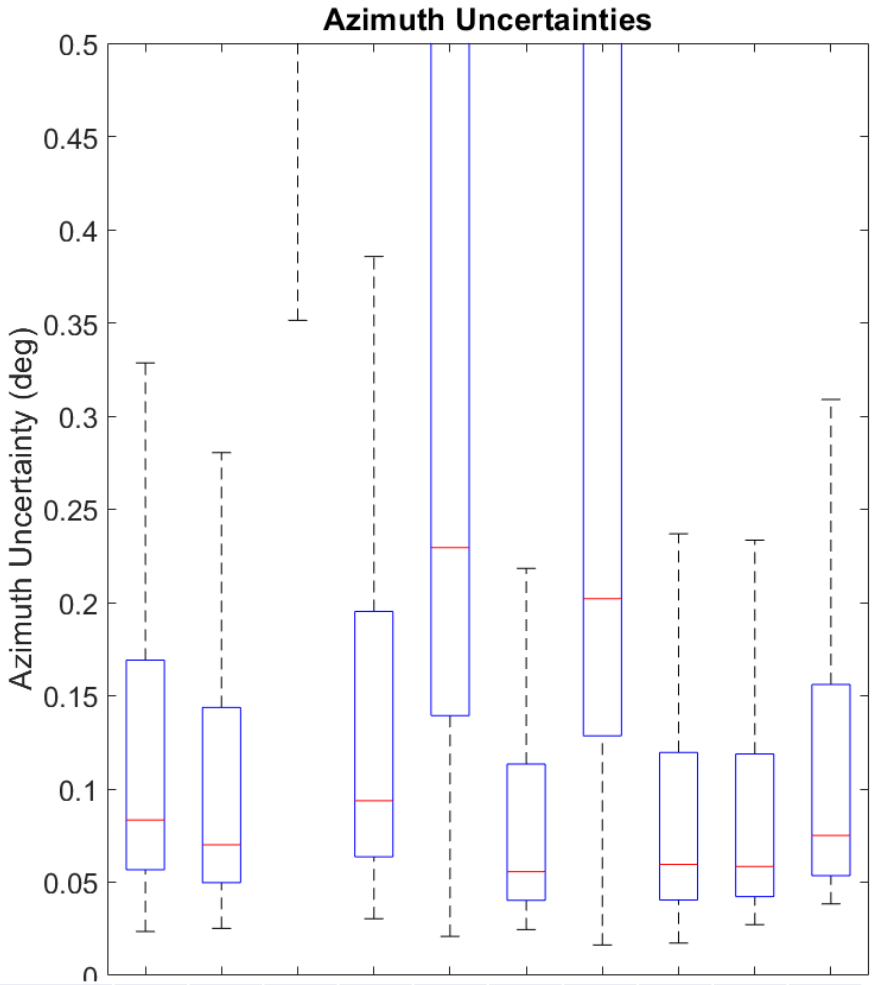


Test	1	2	3	4	5	6	7	8	9	10
Distance (km)	56	56	3	44	31	64	32	70	61	49
Max Angle (deg)	96	64	94	84	94	72	89	77	74	88

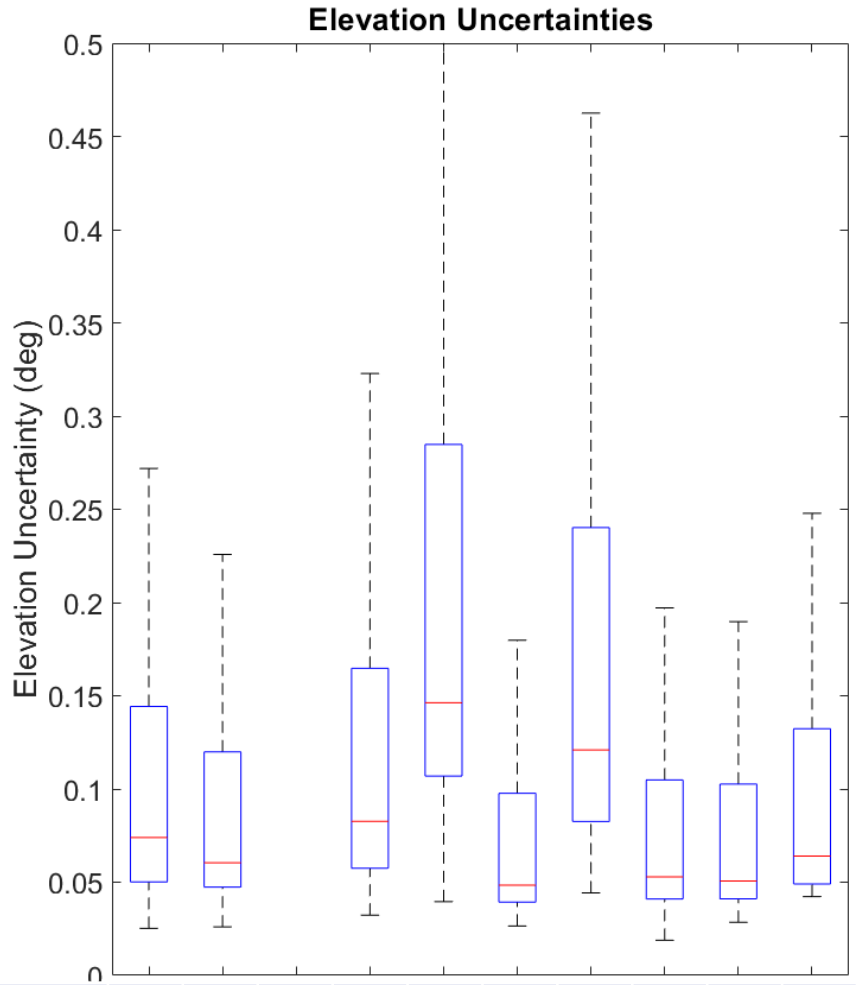


Test	1	2	3	4	5	6	7	8	9	10
Distance (km)	56	56	3	44	31	64	32	70	61	49
Max Angle (deg)	96	64	94	84	94	72	89	77	74	88

Triangles 6, 8, and 9 have lowest spread and median uncertainty.



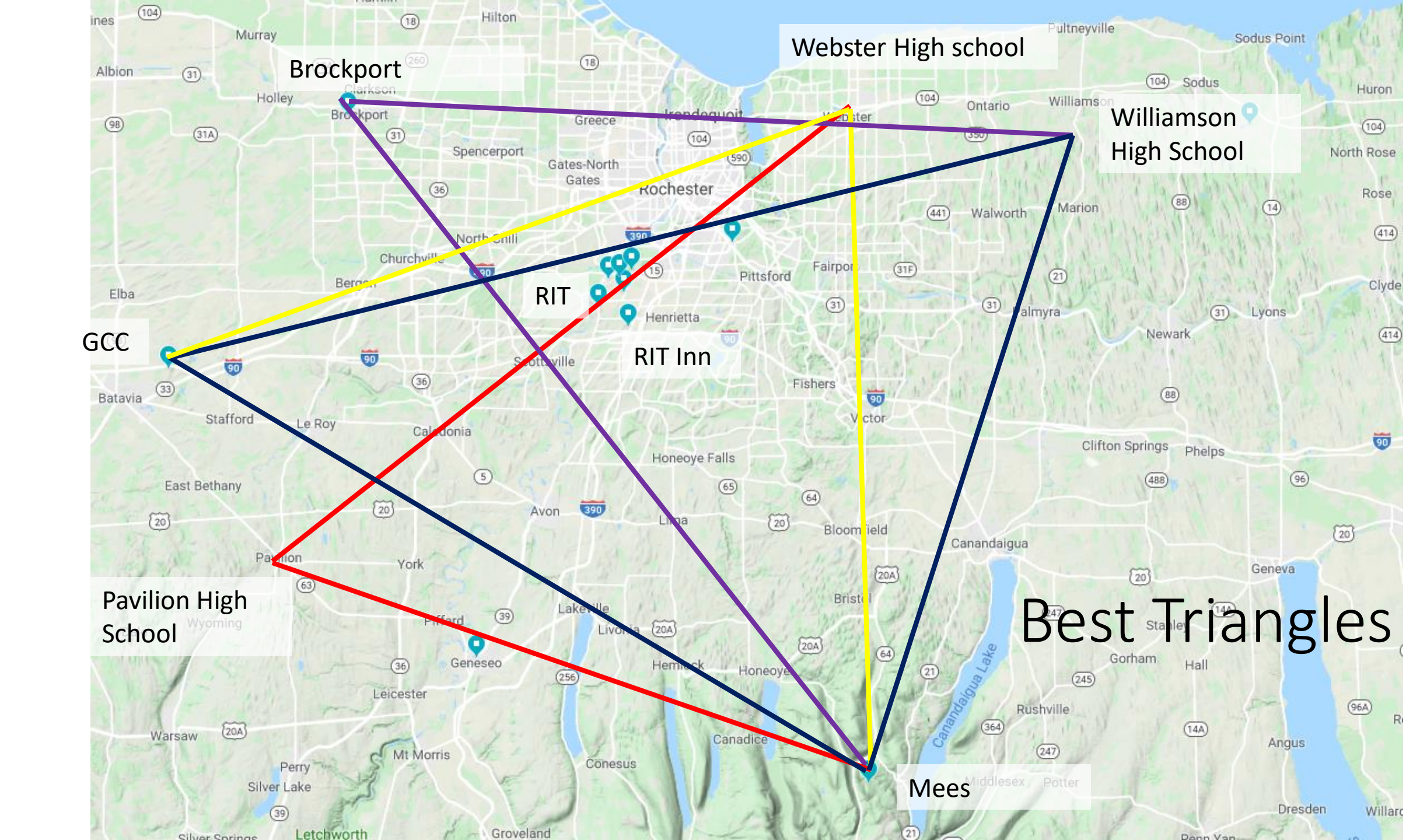
Test	1	2	3	4	5	6	7	8	9	10
Distance (km)	56	56	3	44	31	64	32	70	61	49
Max Angle (deg)	96	64	94	84	94	72	89	77	74	88



Test	1	2	3	4	5	6	7	8	9	10
Distance (km)	56	56	3	44	31	64	32	70	61	49
Max Angle (deg)	96	64	94	84	94	72	89	77	74	88

Both Mean Distance and having the smallest max angle makes the best triangle.

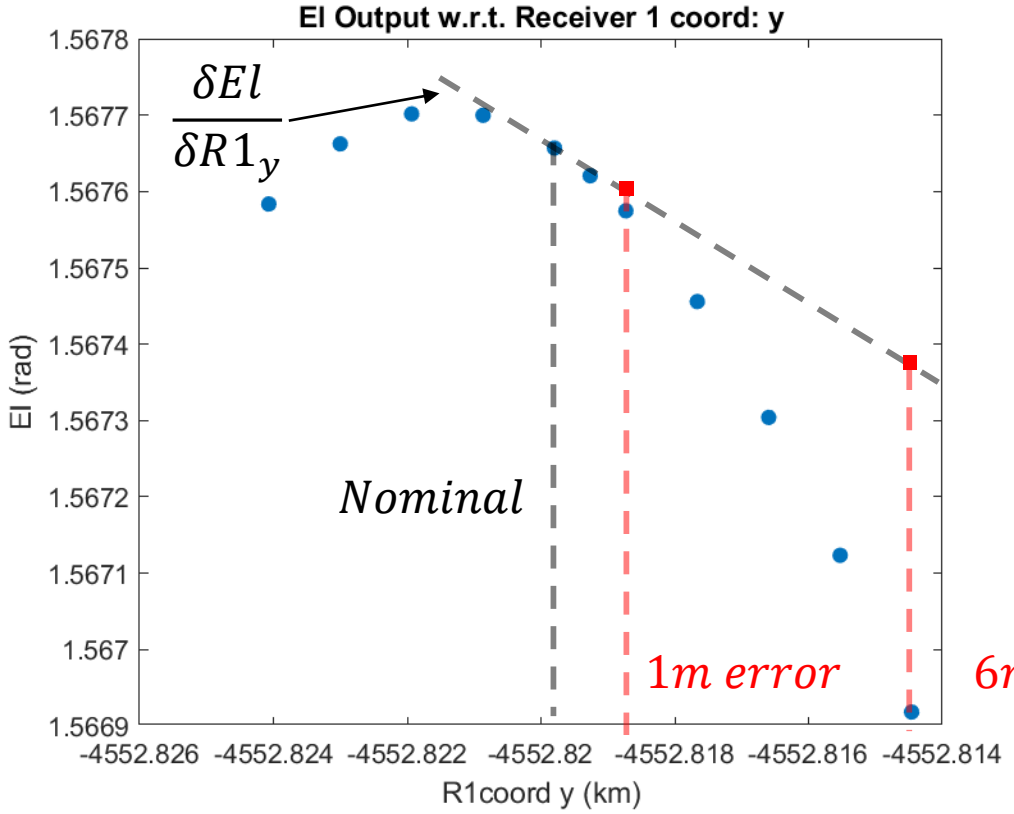
Triangle	Mean Distance (km)	Largest Angle (deg)	TDoA Coverage	Uncertainty Coverage (1 deg)	Median Uncertainty Az	IQR Az	Median Uncertainty El	IQR El	Minimum Elevation (1 deg)
3. RIE	3	94	84%	0%	5.91	7.95	2.76	1.86	NaN
5. RBW	31	94	99%	66%	0.23	0.47	0.15	0.18	35
7. RGB	32	89	99%	68%	0.2	0.39	0.12	0.16	40
4. MPR	44	84	100%	79%	0.09	0.13	0.08	0.11	20
10. MRWi	49	88	100%	82%	0.08	0.1	0.06	0.08	15
1. MBW	56	96	100%	83%	0.08	0.11	0.07	0.09	20
2. MWP	56	64	100%	89%	0.07	0.09	0.06	0.07	15
9. MWG	61	74	100%	90%	0.06	0.08	0.05	0.06	15
6. MWiB	64	72	100%	90%	0.06	0.07	0.05	0.06	15
8. MGWi	70	77	100%	90%	0.06	0.08	0.05	0.06	20


Webster High school
Brockport
Williamson High School
RIT
GCC
RIT Inn
Pavilion High School
Mees
Best Triangles

When we increase our uncertainty, OAAT assumption of linear sensitivity breaks down.

OAAT assumes linear sensitivity:

- Derive the partial derivative of elevation wrt to R1 (y).
- Multiply that partial derivative by the uncertainty in R1 (y) to estimate uncertainty in Elevation. (Red Line)



TDoA is a highly nonlinear problem. Very quickly from the nominal point, we deviate from a linear approximation.

OAAT overestimates R1 (y) contribution to uncertainty for uncertainty in R1 (y) beyond 1m.

We can estimate uncertainty for nonlinear problems with a Monte Carlo Approach.

For a Monte Carlo analysis:

Given

$$g = f(x, y, z, \dots)$$

Then,

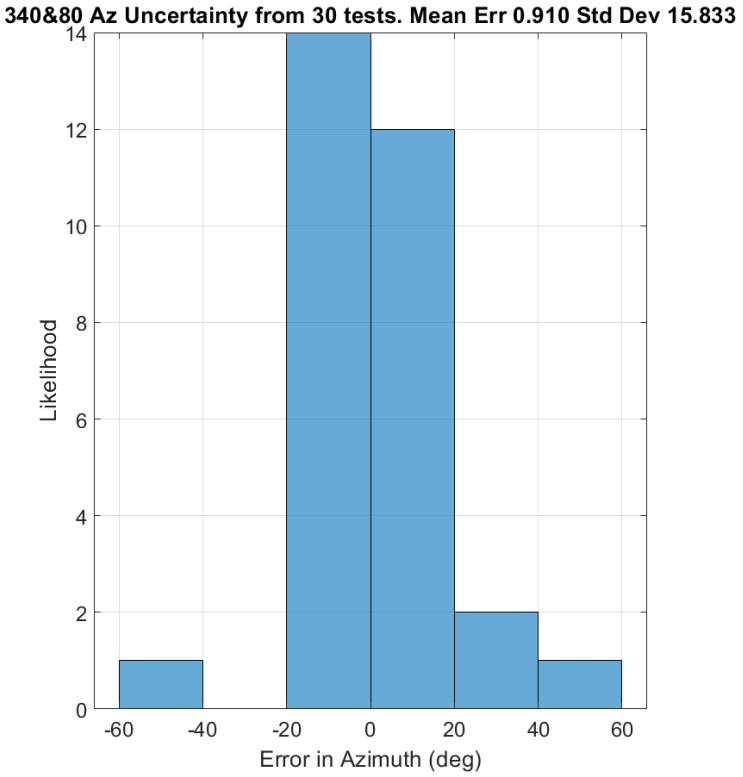
$$g_1 = f(x + \Delta x_1, y + \Delta y_1, z + \Delta z_1, \dots)$$

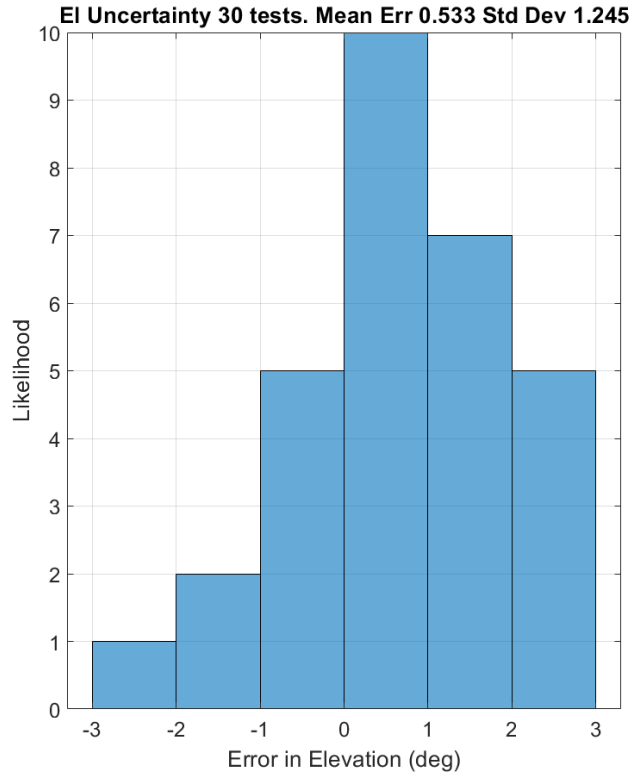
$$g_n = f(x + \Delta x_n, y + \Delta y_n, z + \Delta z_n, \dots)$$

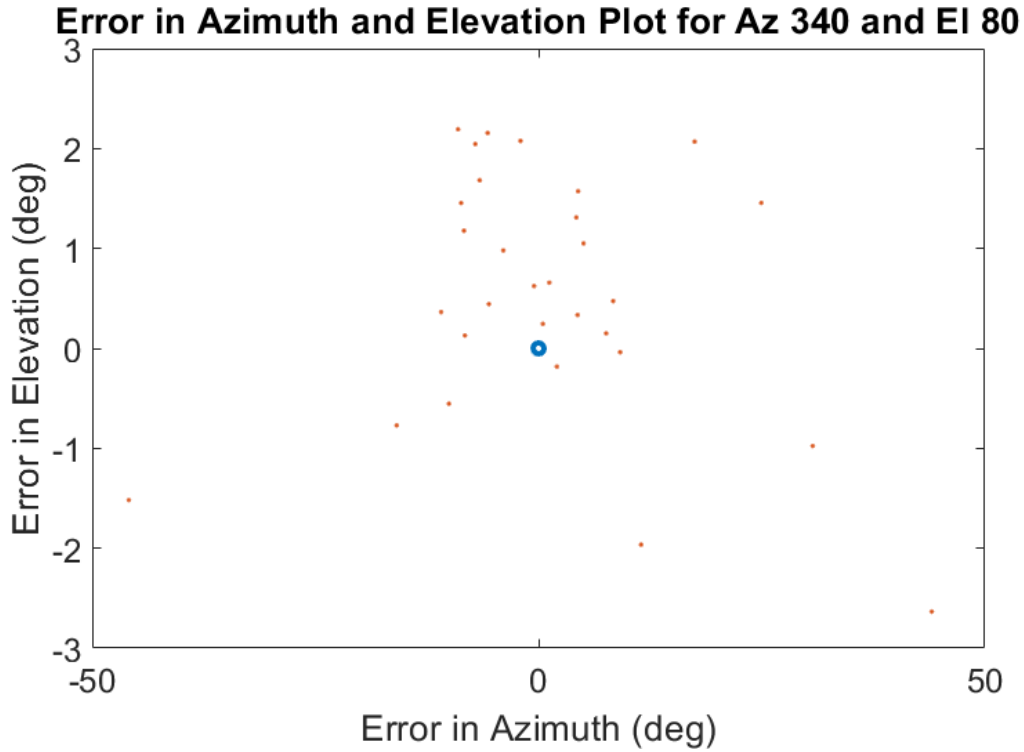
$$\Delta g = \overline{g_n} + 2\sigma_{g_n}$$

We perturbate all inputs by a Gaussian scaled by that input's associated uncertainty. We solve the problem multiple times then take run statistics on the results. We take 2 times the Std Dev. to account for 95% of the spread.

Monte Carlo outputs a histogram showing the spread of solutions based on the input uncertainties.



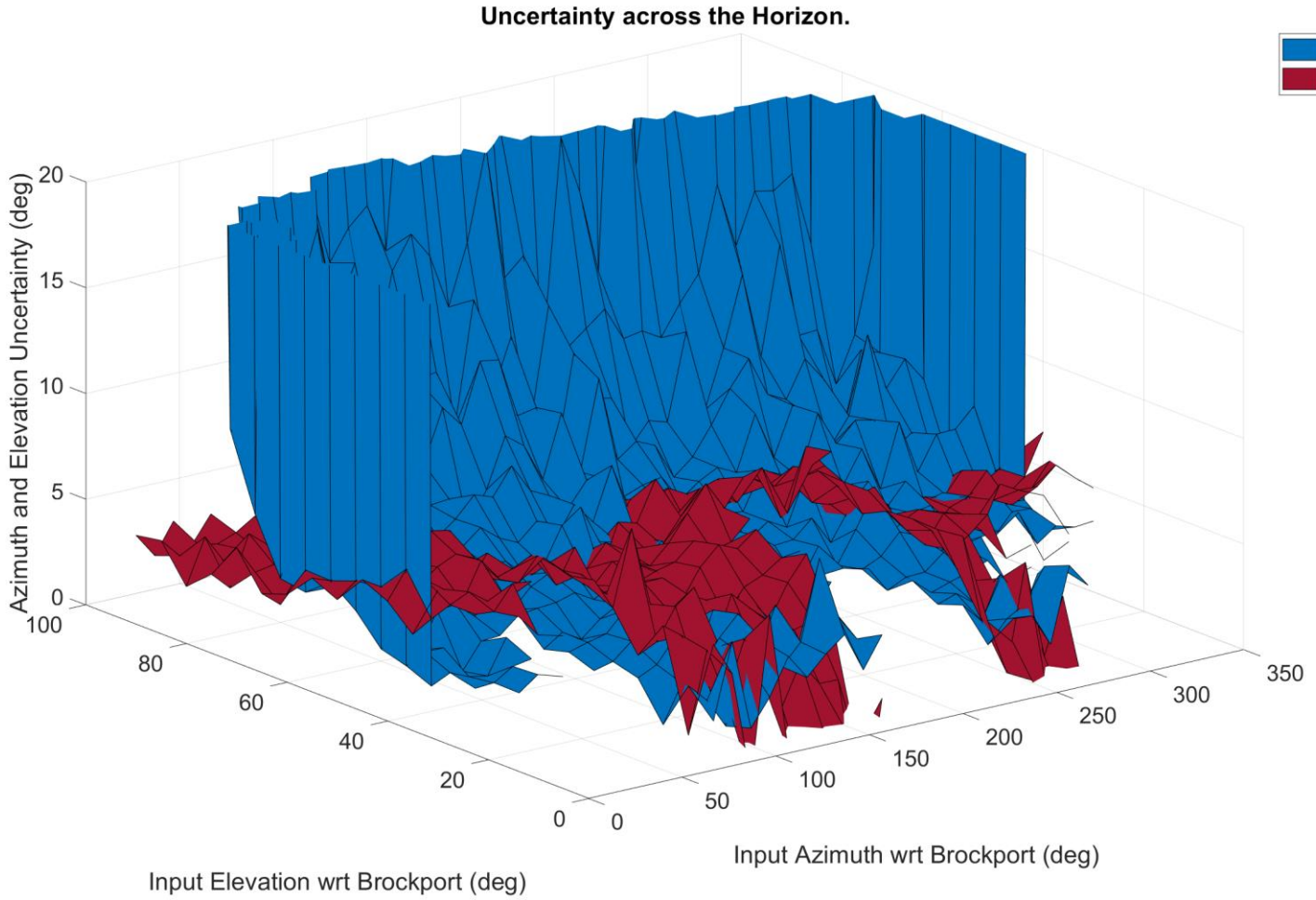




We sample the distribution 30 times. We chose 30 based on the *Central Limit Theorem* in statistics, which says we can approximate the data as a Gaussian with 30 samples. We see the mean error is not zero. Recall that the TDoA code has inherent error in it. We would predict the following uncertainties.

- $\Delta Az = 0.91 + 2 * 15.83 = 32.5\ degrees$
- $\Delta El = 0.53 + 2 * 1.25 = 3\ degrees$

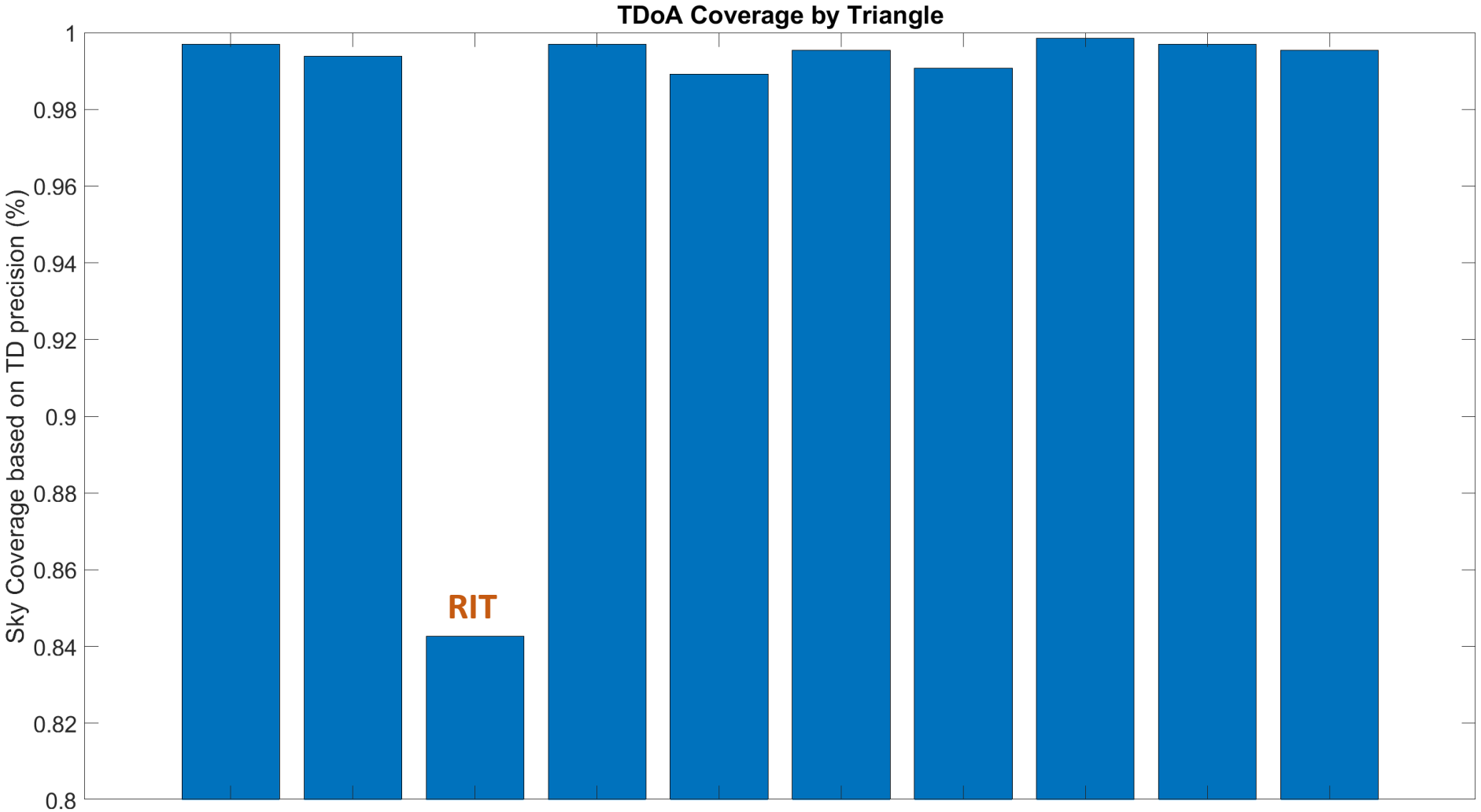
When we apply RSME, we can estimate the uncertainty based on OAAT.



While the exact location varies, the overall shape seen in this plot remains constant for all triangles.

- Elevation is highly uncertain in 2 low azimuth locations.
- In general all elevations grow less certain with lower elevations..
- Azimuth is highly uncertain in 2 high elevation places.
- In general all azimuths grow less certain with higher elevations.

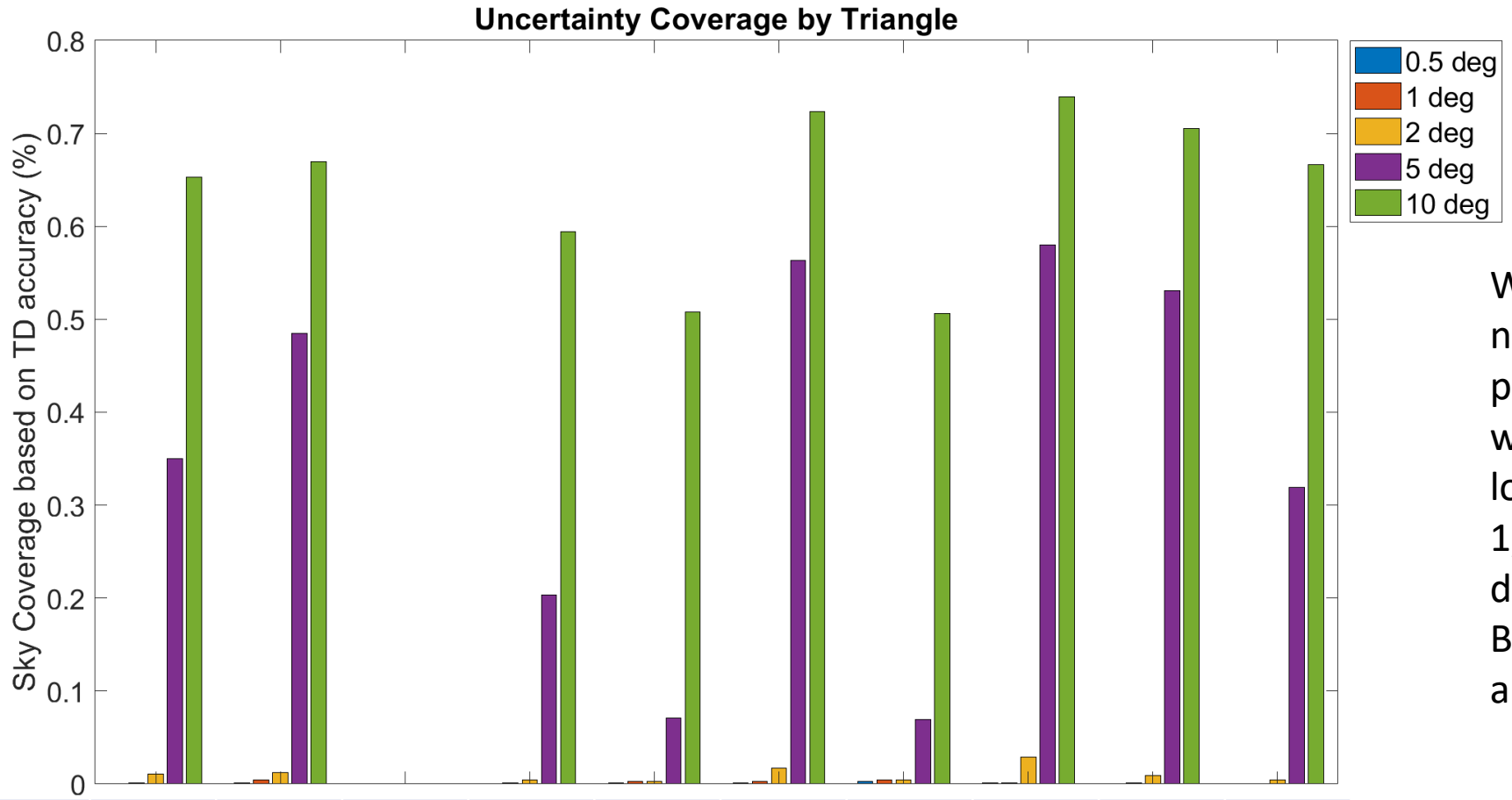
All triangles tend to have high coverage based on the minimum time difference we can measure.



Notice we start at 80%

Test	1	2	3	4	5	6	7	8	9	10
Distance (km)	56	56	3	44	31	64	32	70	61	49
Max Angle (deg)	96	64	94	84	94	72	89	77	74	88

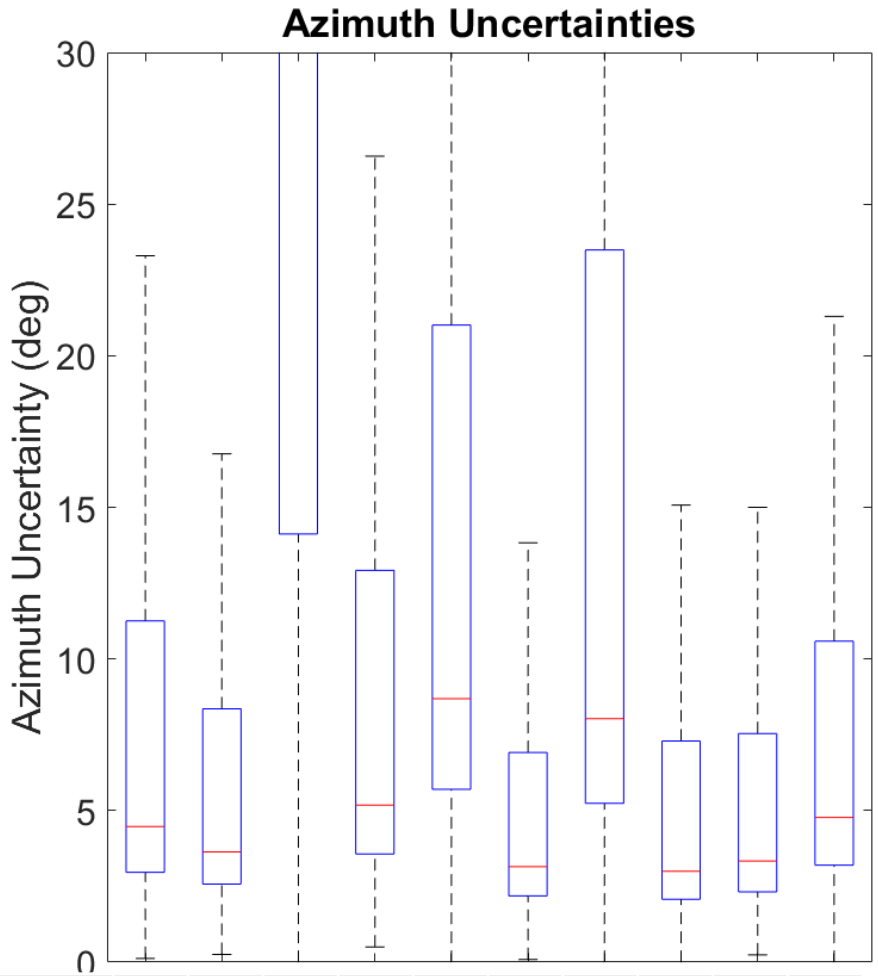
Triangles with larger mean distances have better coverage based on uncertainty.

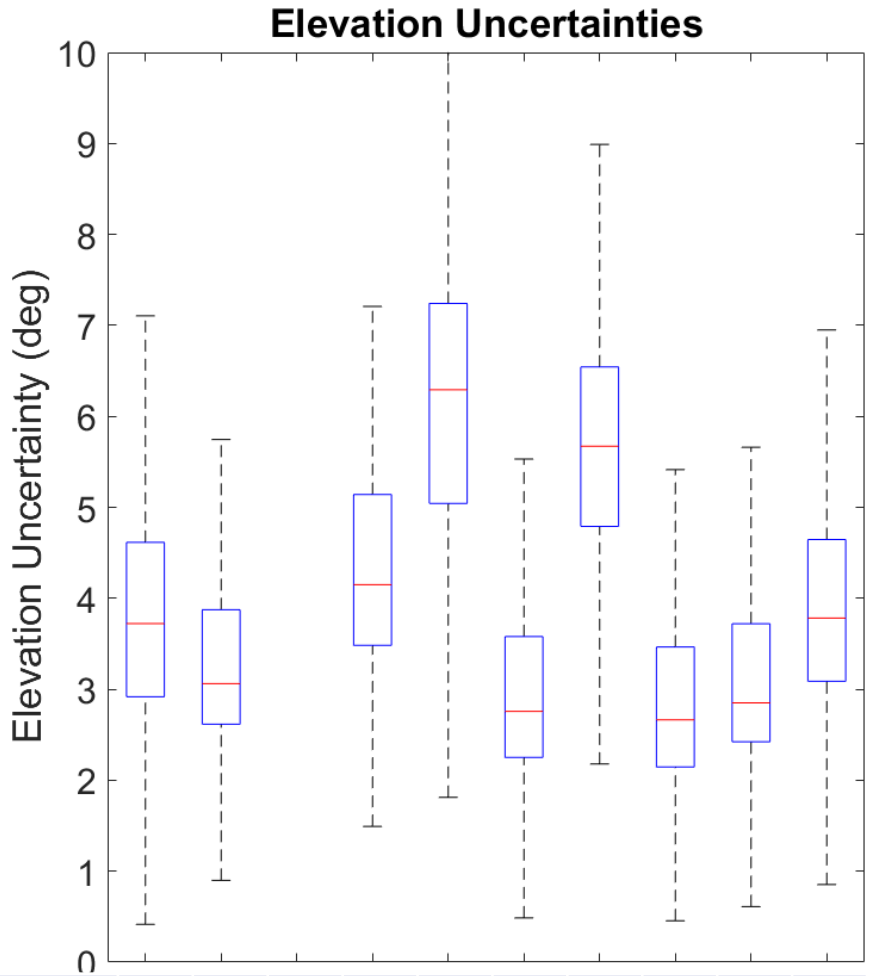


Test	1	2	3	4	5	6	7	8	9	10
Distance (km)	56	56	3	44	31	64	32	70	61	49
Max Angle (deg)	96	64	94	84	94	72	89	77	74	88

We count number of points in the net with uncertainty lower than 0.5, 1, 2, and 5 degrees for BOTH azimuth and elevation.

Triangles 6, 8, and 9 have lowest spread and median uncertainty.





Test	1	2	3	4	5	6	7	8	9	10
Distance (km)	56	56	3	44	31	64	32	70	61	49
Max Angle (deg)	96	64	94	84	94	72	89	77	74	88

Test	1	2	3	4	5	6	7	8	9	10
Distance (km)	56	56	3	44	31	64	32	70	61	49
Max Angle (deg)	96	64	94	84	94	72	89	77	74	88

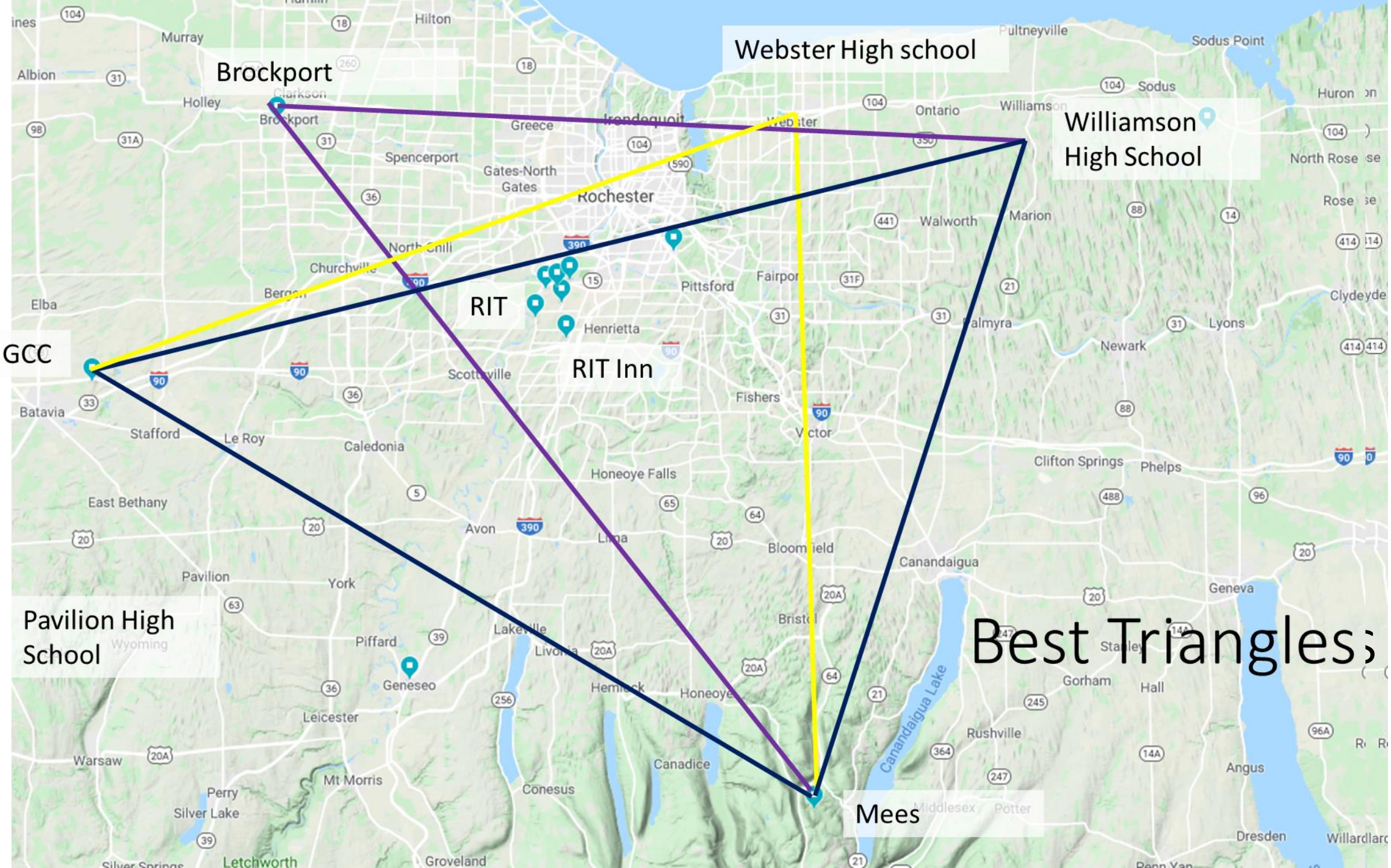
Both Mean Distance and having the smallest max angle makes the best triangle.

Triangle	Mean Distance (km)	Largest Angle (deg)	TDoA Coverage	Uncertainty Coverage (10 deg)	Median Uncertainty Az	IQR Az	Median Uncertainty El	IQR El	Minimum Elevation (5 deg, 40%)	Maximum Elevation (5 deg, 40%)
3. RIE	3	94	84%	0%	51.3	186	20.7	15.8	NaN	0
5. RBW	31	94	99%	51%	8.7	15.3	6.3	2.2	NaN	0
7. RGB	32	89	99%	51%	8	18.2	5.7	1.8	30	40
4. MPR	44	84	100%	59%	5.2	9.4	4.1	1.7	10	55
10. MRWi	49	88	100%	67%	4.8	7.4	3.8	1.6	5	55
1. MBW	56	96	100%	65%	4.5	8.3	3.7	1.7	5	60
2. MWP	56	64	100%	67%	3.6	5.8	3.1	1.3	15	65
9. MWG	61	74	100%	71%	3.3	5.2	2.9	1.3	5	65
6. MWiB	64	72	100%	72%	3.2	4.7	2.8	1.3	5	65
8. MGWi	70	77	100%	74%	3	5.2	2.7	1.3	5	70

Least Squares Results in higher coverage, but suffers with slightly higher uncertainty.

Triangle	Mean Distance (km)	Largest Angle (deg)	TDoA Coverage	Uncertainty Coverage (10 deg)	Median Uncertainty Az	IQR Az	Median Uncertainty El	IQR El	Minimum Elevation (5 deg, 40%)	Maximum Elevation (5 deg, 40%)
3. RIE	3	94	84%	1%	140.2	96.6	18.7	11.9	90	20
5. RBW	31	94	99%	51%	9.2	16.8	6.4	2.1	NaN	0
7. RGB	32	89	99%	56%	8.5	12.8	5.8	1.8	30	10
4. MPR	44	84	100%	70%	5.6	7.7	4.3	1.8	15	50
10. MRWi	49	88	100%	73%	5.0	6.9	3.9	1.7	5	60
1. MBW	56	96	100%	72%	4.8	7.4	3.9	1.8	5	60
2. MWP	56	64	100%	78%	3.8	4.9	3.3	1.6	5	65
9. MWG	61	74	100%	81%	3.6	4.2	3.1	1.6	5	65
6. MWiB	64	72	100%	81%	3.5	3.8	2.8	1.6	5	70
8. MGWi	70	77	100%	82%	3.2	3.9	2.8	1.6	5	70

Exact inputs as Montecarlo on previous slide, but the solver is least squares instead of symbolic. This simulation took 1 hour.


Webster High school
Brockport
Williamson High School
GCC
RIT
RIT Inn
Pavilion High School
Mees
Best Triangles;

Solving with more than 3 stations

By Anthony Iannuzzi

With 4 base stations, we find 4 possible directions. We can use an average direction from this, or use least squares.

1. Create m sets of 3 Hyperboloids.
2. Solve each of these individually
3. Intersect the resulting line fits via least squares.
4. Output: Single Point.

$$y = Mw$$

$$x = (M^T M)^{-1} * M^T * y$$

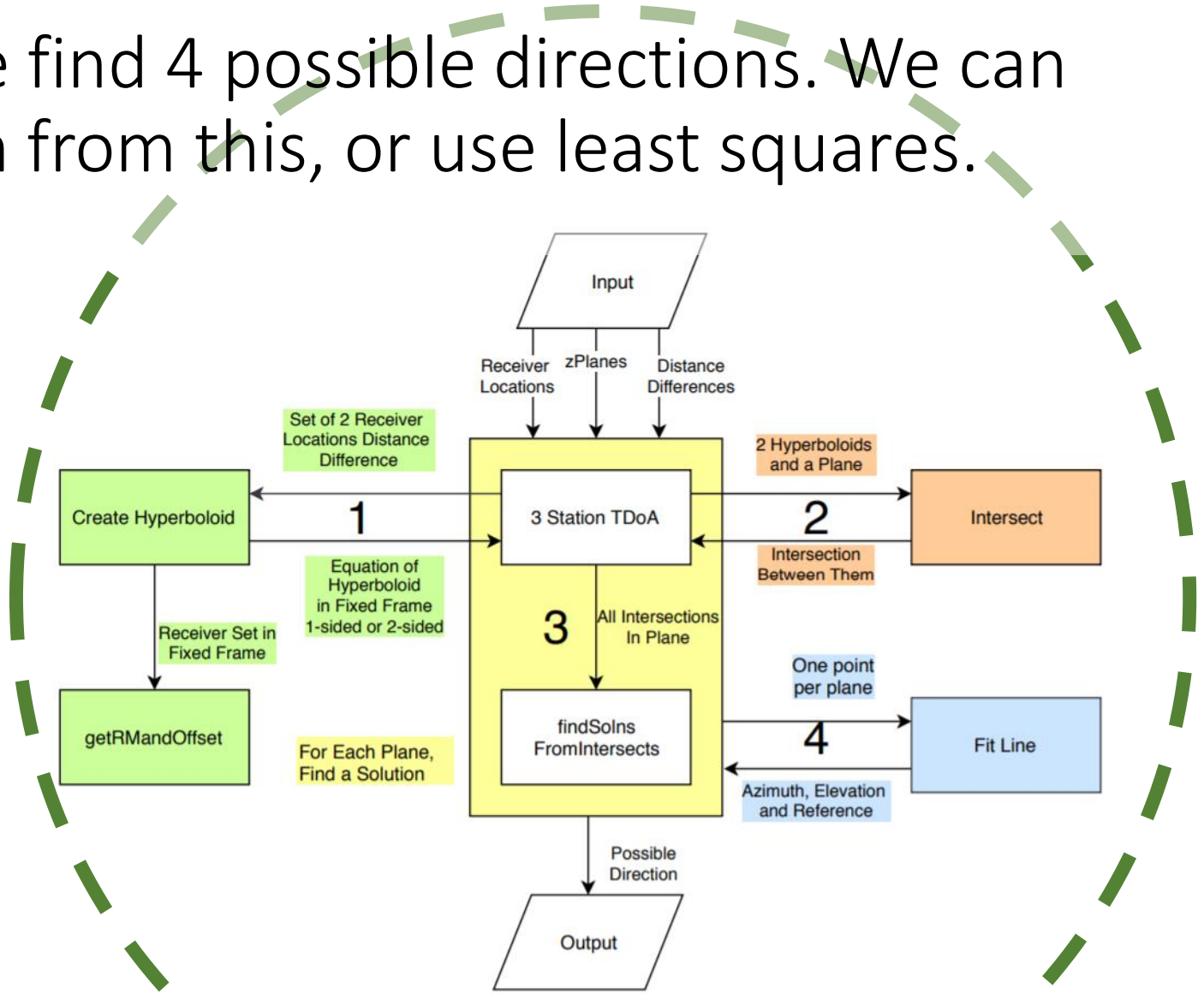
For 2 lines:

$$\begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ x_{02} \\ y_{02} \\ z_{02} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & x_d & 0 \\ 0 & 1 & 0 & y_d & 0 \\ 0 & 0 & 1 & z_d & 0 \\ 1 & 0 & 0 & 0 & x_{d2} \\ 0 & 1 & 0 & 0 & y_{d2} \\ 0 & 0 & 1 & 0 & z_{d2} \end{bmatrix} * \begin{bmatrix} x \\ y \\ z \\ t \\ s \end{bmatrix}$$

$3n\ x\ 1 \qquad 3n\ x\ (3+n) \qquad (3+n)\ x\ 1$

$r = r_0 + tr_d$

$r_2 = r_{02} + sr_{d2}$



$x4$

Average direction may be useful. The error for solving a point is large with no input noise.

- Sensitivity is due to 4 near parallel lines.
- Intersecting near parallel lines is error prone.

Rotating the green line by a few degrees significantly changes the intersection.