**MAE 593**

Homework 1

Least Squares GPS Solution

Due: 9/11/2014

By: Ryan Watson

* 1. Develop a MATLAB® function that implements the Linear Least Square positioning based on the model we derived in class.

To derive a Least Squares solution to this problem some educated, initial assumptions needed to be made. Primarily, we need to make an assumption about the uses initial position (nominal position ( , )) and the initial clock bias. These nominal values are of great importance because they are the values around which we linearize the first-order Taylor series expansion of Equation 1.

= + c (1)

Once the system has been linearized we can solve for the change from the nominal position. This is mathematically represented by the linear relationship shown in Equation 2.

=+ (2)

From this model we can calculated the pseudorange. The pseudorange calculation, written in a similar form as Equation 2, can be seen below as Equation 3. In Equation 3, it can be seen that the difference between the computed pseudorange and the nominal pseudo range is linearly proportional to the unit vector between the user and the satellite and the delta nominal values. The matrix mapping the delta nominal values to the delta pseudorange values is known as the geometry matrix because its elements are the unit vector between the user and the satellite.

= (3)

The above linear relationship describe by Equation 3 can be written more compactly as Equation 4. From Equation 4, it can be seen that the delta nominal values can be calculated by taking the inverse of the geometry matrix and applying left addition. This solution can be seen in Equation 5.

(4)

(5)

Now that I have briefly described the Linear Least Square (LLS) solution to a GPS application, I will discuss the results obtained by applying the algorithm. For part one of problem one, we were asked to calculate and plot the position error for two data sets in both Earth Centered Earth Fixed (ECEF) and East, North, Up (ENU). We were also asked to compare the difference between an iterative and non-iterative nominal update value.

In Figure 1 through Figure 3, the non-iterative solution to data set one can be seen. The ECEF solution can be seen in Figure 1 and the ENU solution can be seen in Figure 2. One point of interest in both Figure1 and Figure 2 is the sudden position error at increase. This phenomenon will be addressed later in the paper. Figure 3 shows the difference between the ECEF solution and the ENU solution. As was to be expected, the two solutions are nearly identical. The difference between the two solutions is on the order of .

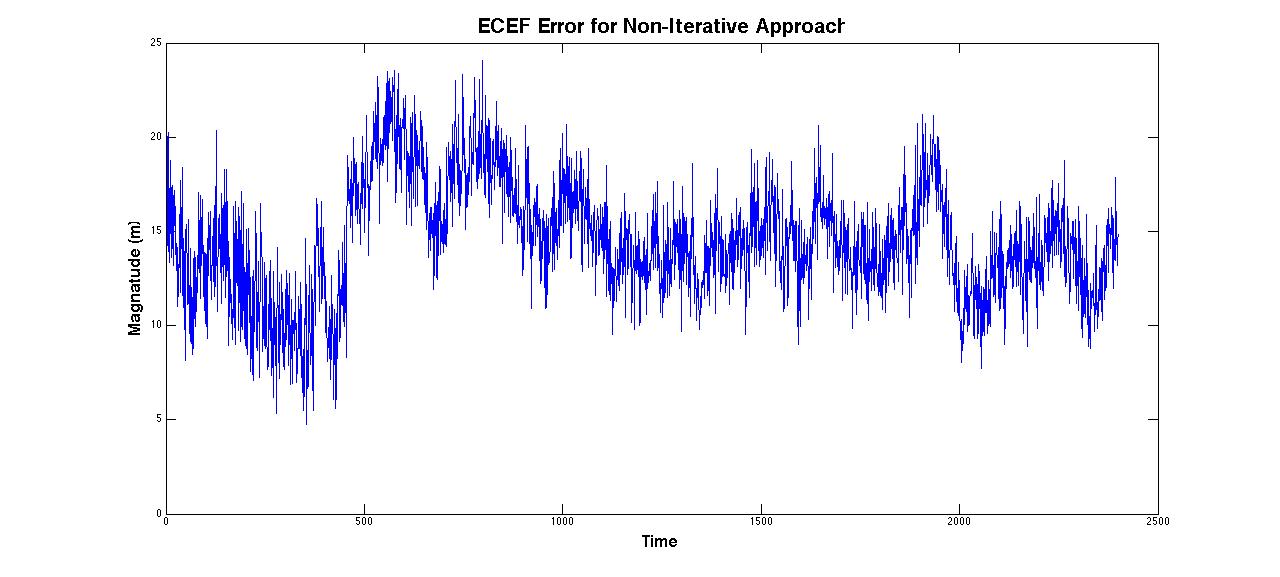


Figure : ECEF Solution for dataset 1, non-iterative solution

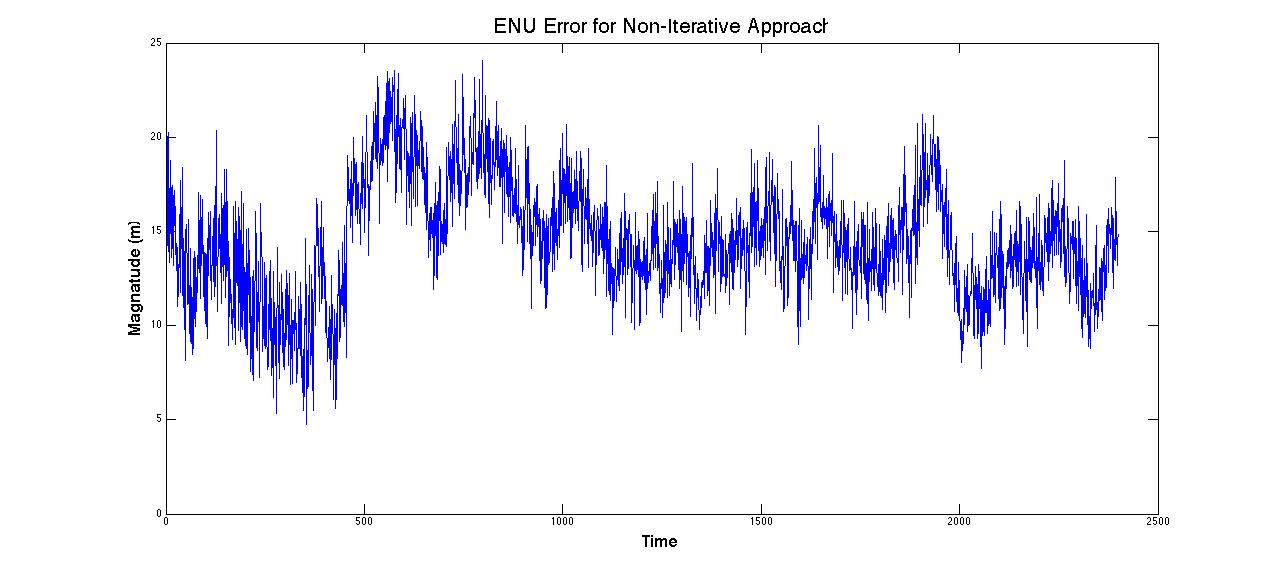


Figure : ENU solution to dataset 1, non-iterative solution

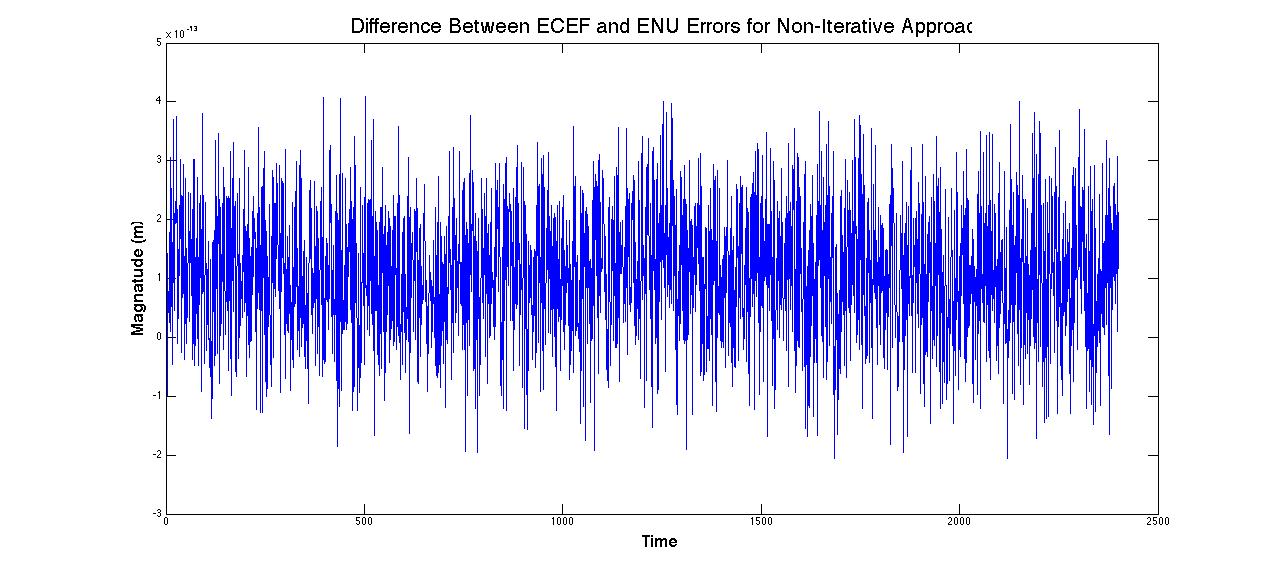


Figure : Difference between ECEF solution and ENU solution

The same procedure as above was conducted to calculate the position error for a non-iterative approach for dataset two. The difference between dataset one and dataset two is that dataset one is stationary while dataset two is a mobile platform. Comparing Figures 4 through 6 to Figures 1 through 3 little difference can be seen. To me this signifies that the positioning errors associated with this problem are more correlated to satellite geometry than the relatively small motion of the users platform.

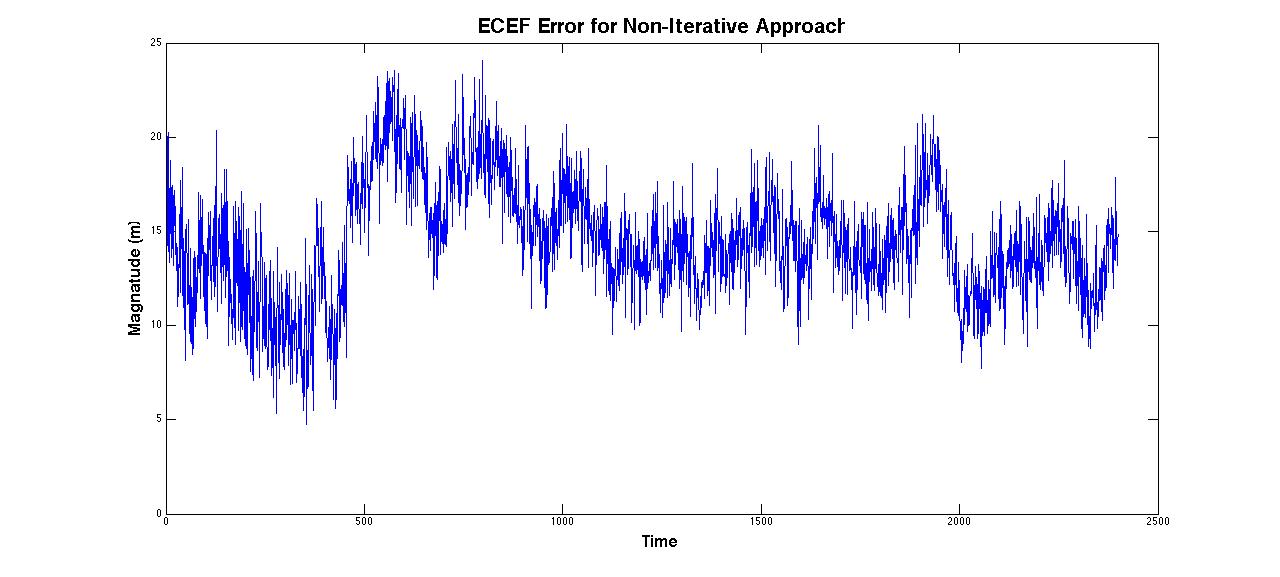


Figure : ECEF solution to dataset 2, non-iterative approach

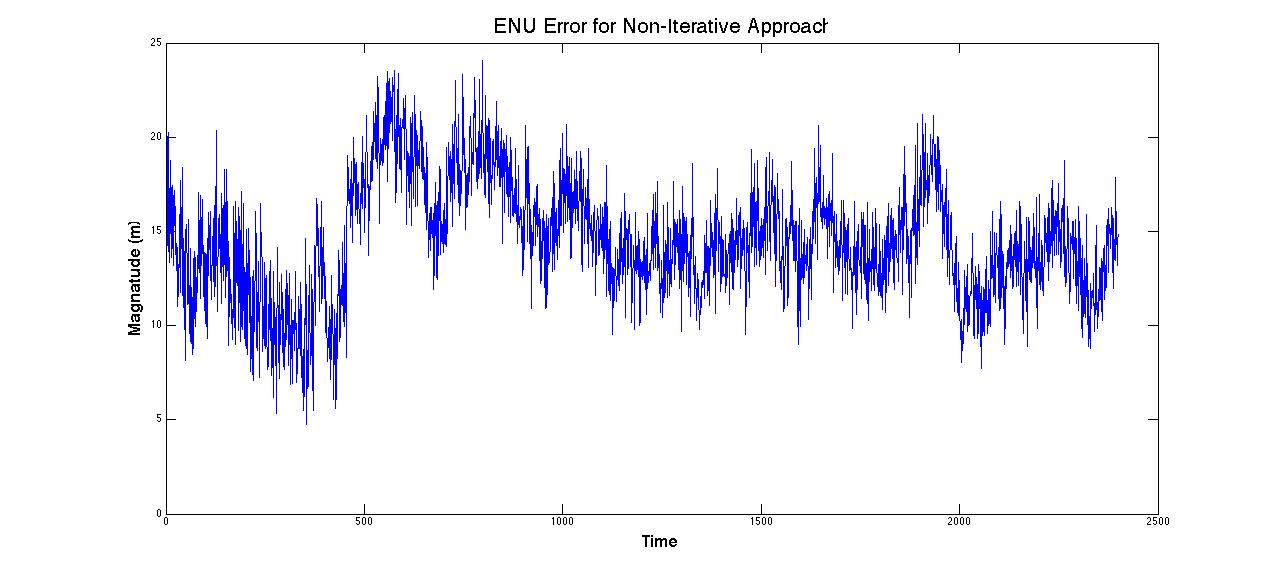


Figure : ENU solution to dataset 2, non-iterative approach

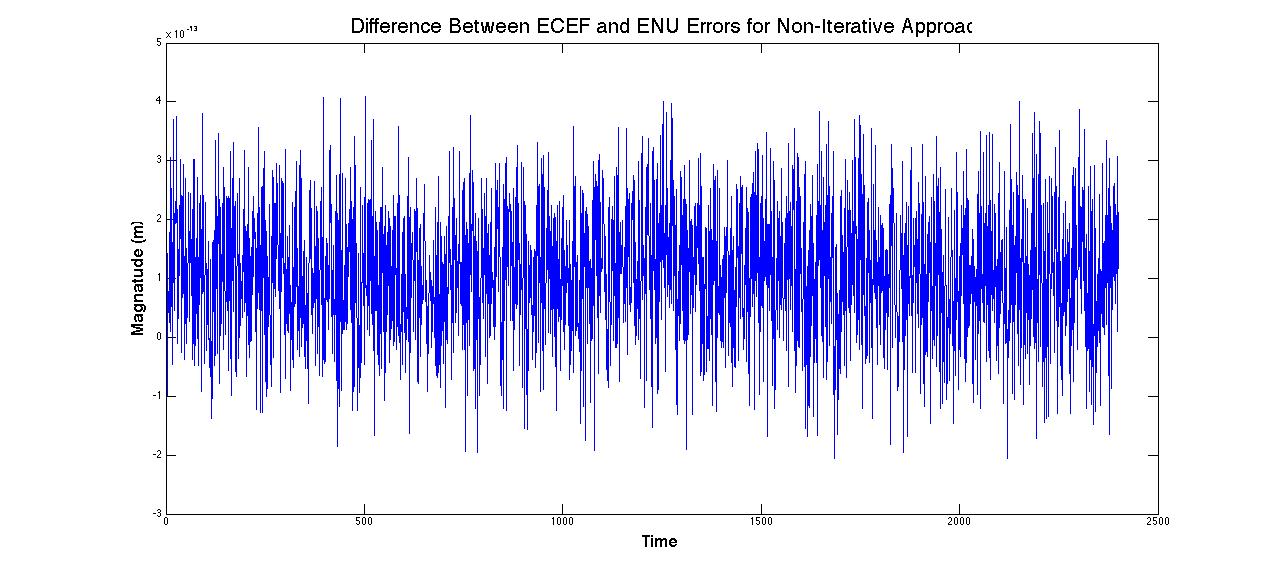


Figure : Difference between the ECEF solution and the ENU solution

For all of the previous examples a constant nominal value was used through out the analysis of the dataset. It would be beneficial to know if correcting the nominal value at each time step would decrease the error in any significant way. To do this the estimated value at each time step was saved as the nominal value for the next time step. The results of this analysis can be seen in Figure 7 through Figure 9.

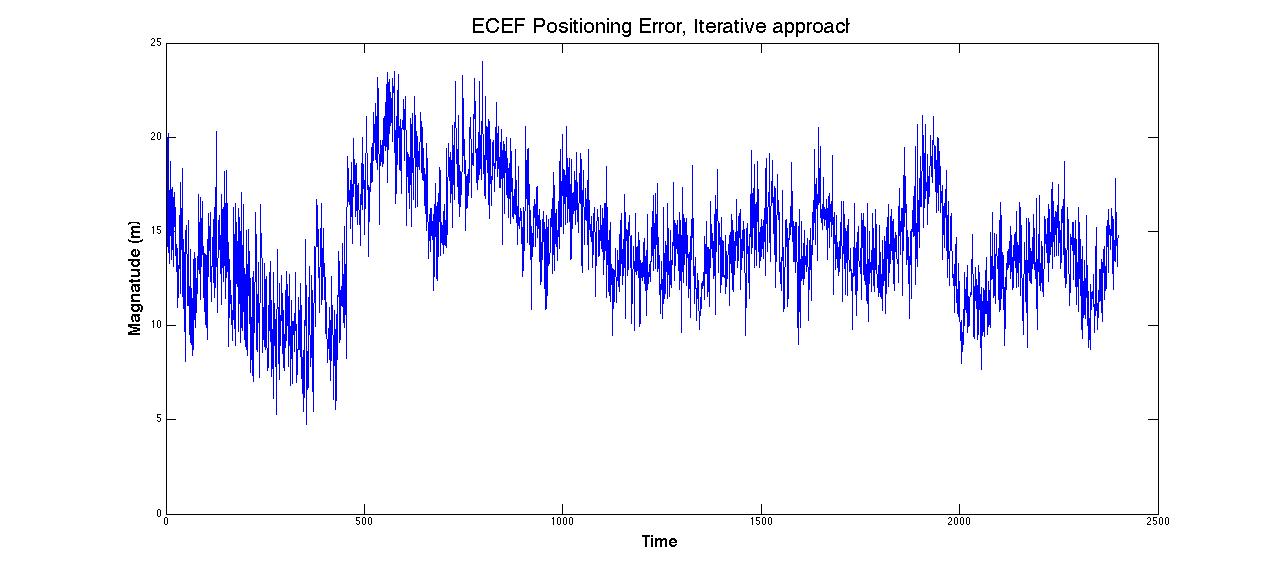


Figure : ECEF Positioning Error, Iterative Approach

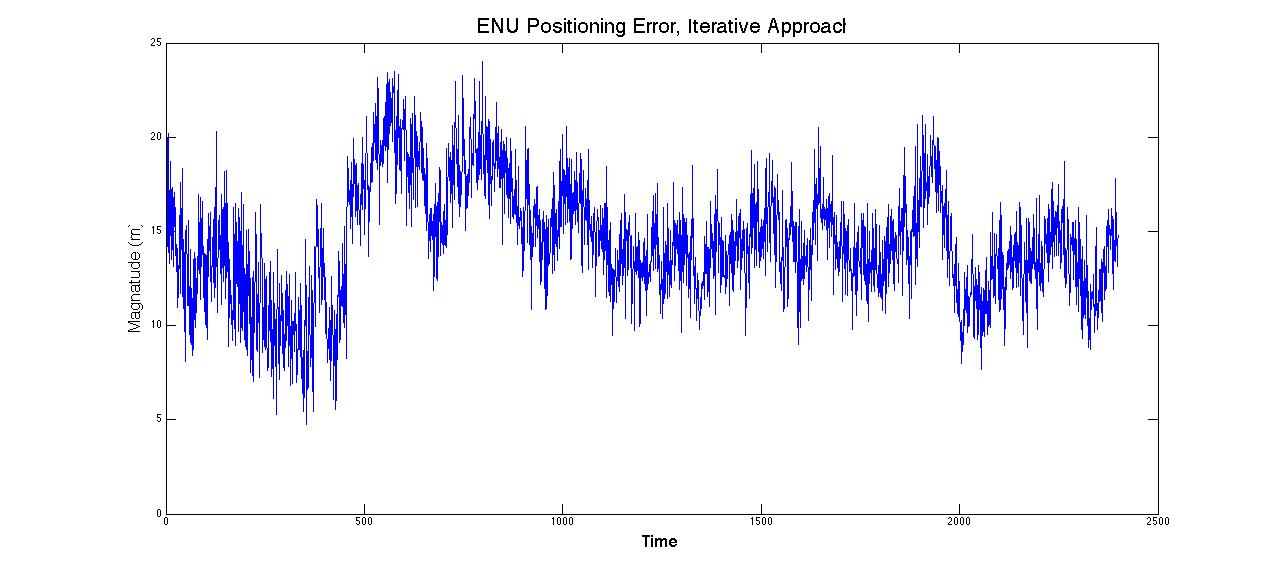


Figure : ENU Positioning Error, Iterative Approach

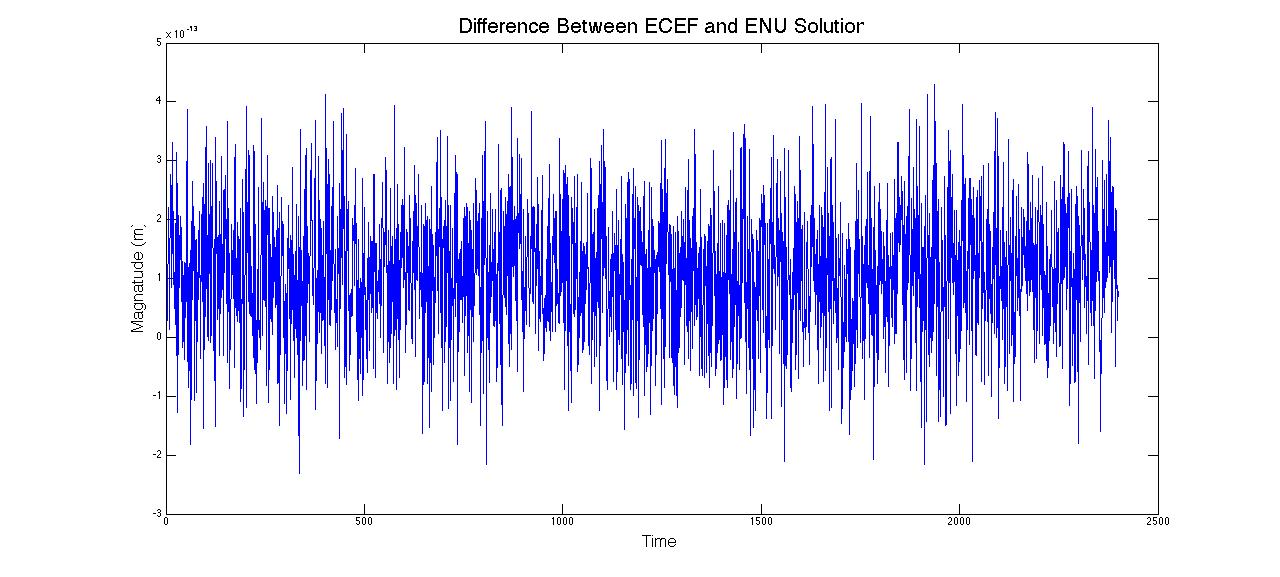


Figure : Difference between ECEF solution and ENU Solution

A visual comparison of Figure 1 with Figure 7 shows that the iterative approach does not significantly decrease the positioning error. This is because the non-iterative solution was able to sufficiently model the nonlinear equation. So really an iterative approach is only useful at a threshold between linear and nonlinear model. I am assuming that you would need an iterative approach to model a platform that is undergoing highly dynamic motion. Because of this, for the rest of this analysis a non-iterative approach for every solution will be used.

* 1. Update the solution to also output PDOP, GDOP, and TDOP.

One of the most important things in engineering or science is the ability to tell someone how confident you are in a measurement or set of data. With GPS one method for determining the certainty in your measurement is the dilution of precision (DOP) associated with that epoch.

For this section of the problem set we were asked to calculated several o the common DOPs. The Geometric Dilution Of Precision (GDOP) represents the amplification of the standard deviation of the measurement errors onto the positioning errors. The Precision Dilution Of Precision (PDOP) is almost identical to the GDOP with the exception that it does not take time into consideration. The Time Dilution Of Precisions on the other hand is only a function of the variance associated with time. A good description of the numerical value associated with the DOP can be seen in Table 1 (http://wiki.gis.com/wiki/index.php/Dilution\_of\_precision\_(GPS)).

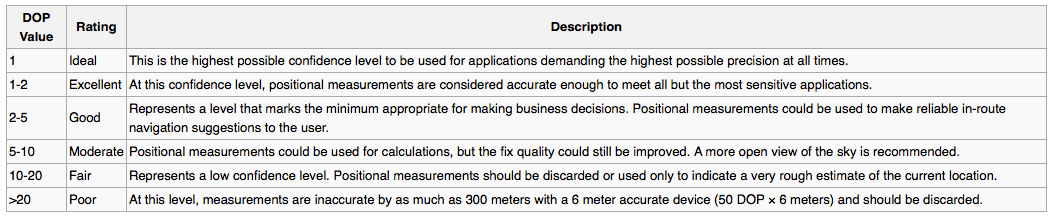
 The Dilution of Precision calculations for data set one can be seen in Figure 10 through Figure 12. As can be seen from the graphs, the greatest DOP value for data set one is just under five. By looking at Table 1, you can see that this means that all of the data that was collected is “good” data.

Table : DOP Value Description

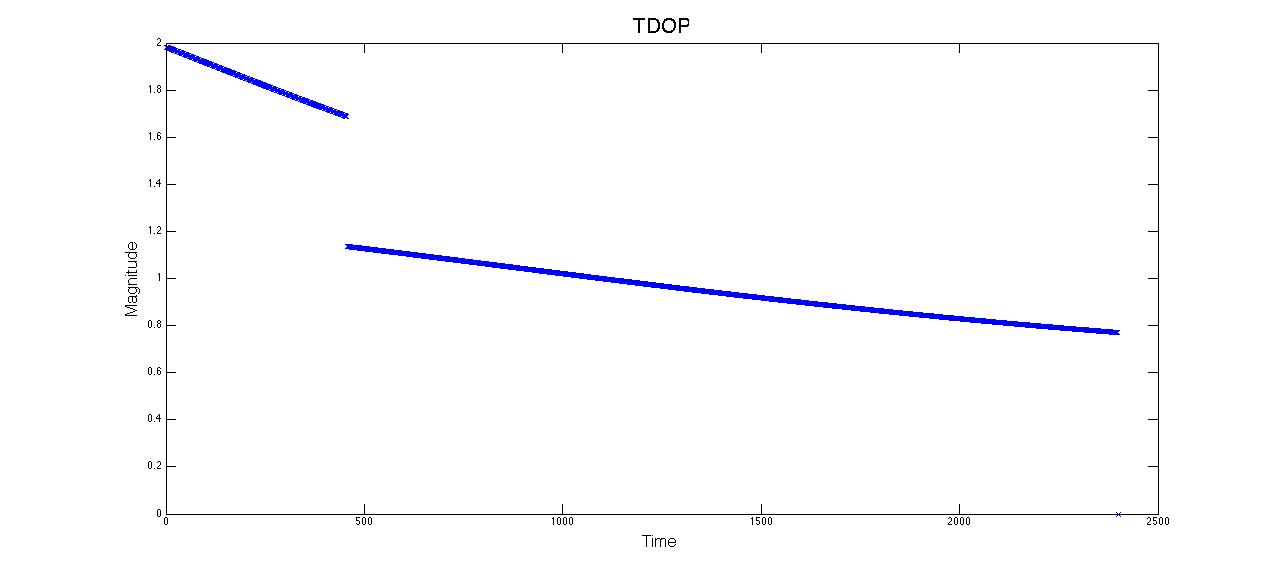


Figure : Time Dilution of Precision for data set 1

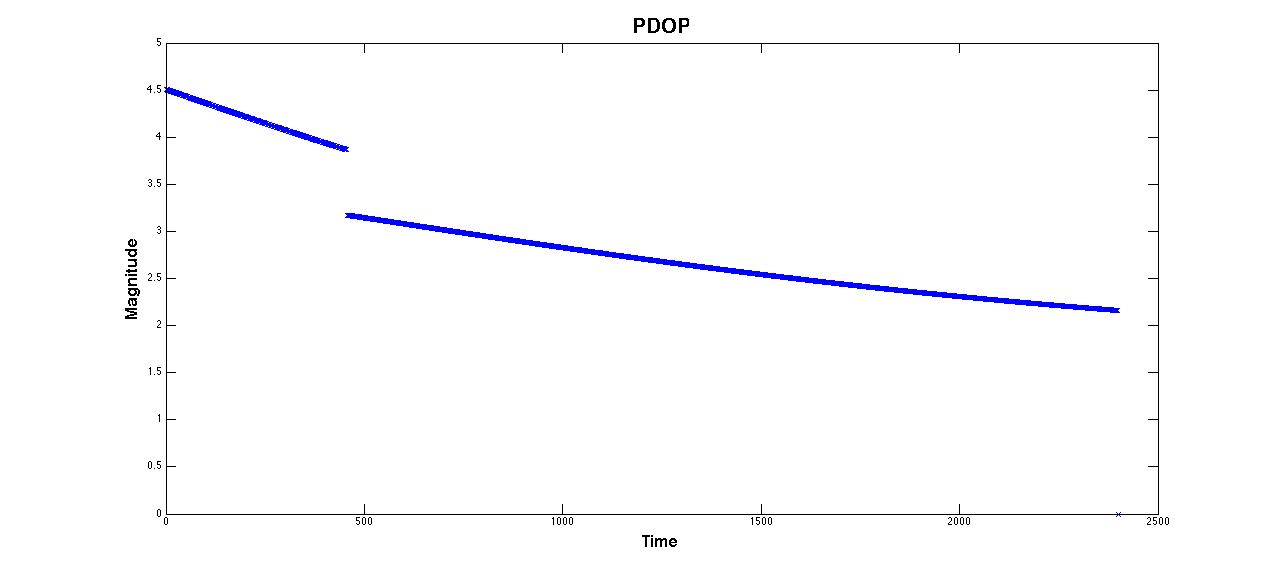


Figure : Precision Dilution of Precision for data set 1

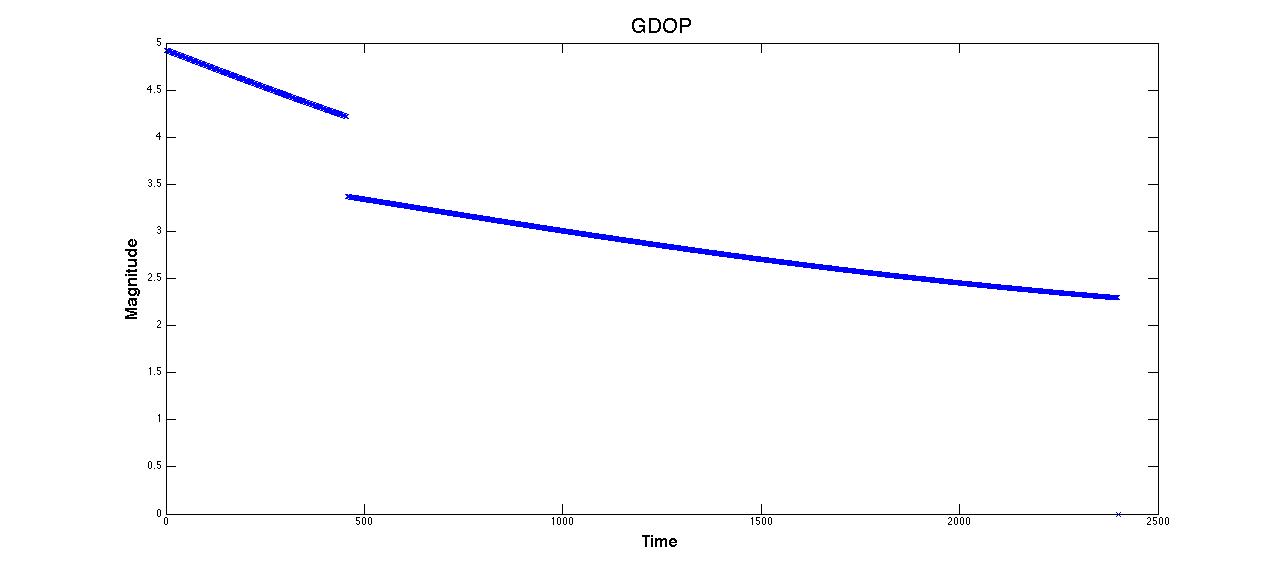


Figure : Geometric Dilution of Precision for data set 1

* 1. Modify the function to accept a weighting matrix, W

One of the largest issues facing GPS today is the modeling of the errors associated with it. One of the easiest ways to reduce the error associated with GPS is to apply a weighting matrix that is a function of the elevation angle of the satellites in view.

The reason that the weighting matrix is a function of the elevation angle is because of the effects that the atmosphere has on the GPS signals. So a lower elevation angle means that that the GPS signal has to travel through more of the atmosphere and therefore will be nosier measurement. This proportionality of GPS error to elevation angle can be seen in Equation 6.

(6)

The weight matrix was calculated by first finding the unit vector between the users position and the satellite. Next the east, north, and up local unit vectors were calculated. Once those values were found the elevation angle can be calculated by taking the dot product between the unit vector and the up vector. The equation to calculate elevation angle can be seen in Equation 7.

el=arcsin( ) (7)

Figure 13 shows the positioning error for data set one with the weighting matrix in the algorithm. At first inspection it does not seem like the weighting matrix really made significant improvement on the solution. However, when the weighted solution is plotted with the non-weighted solution you can see that the positioning error of the weighted solution is smaller when the seventh satellite comes into view. This is to be expected because the weight matrix is a function of the elevation angle, and when a new satellite comes into view it will have a very low elevation angle. If this low elevation angle is not accounted for (i.e. non-weighted solution) the error associated with he low elevation angle of the new satellite will increase the overall error of the solution. The comparison of the weighted solution to the non-weighted solution can be seen in Figure 14.

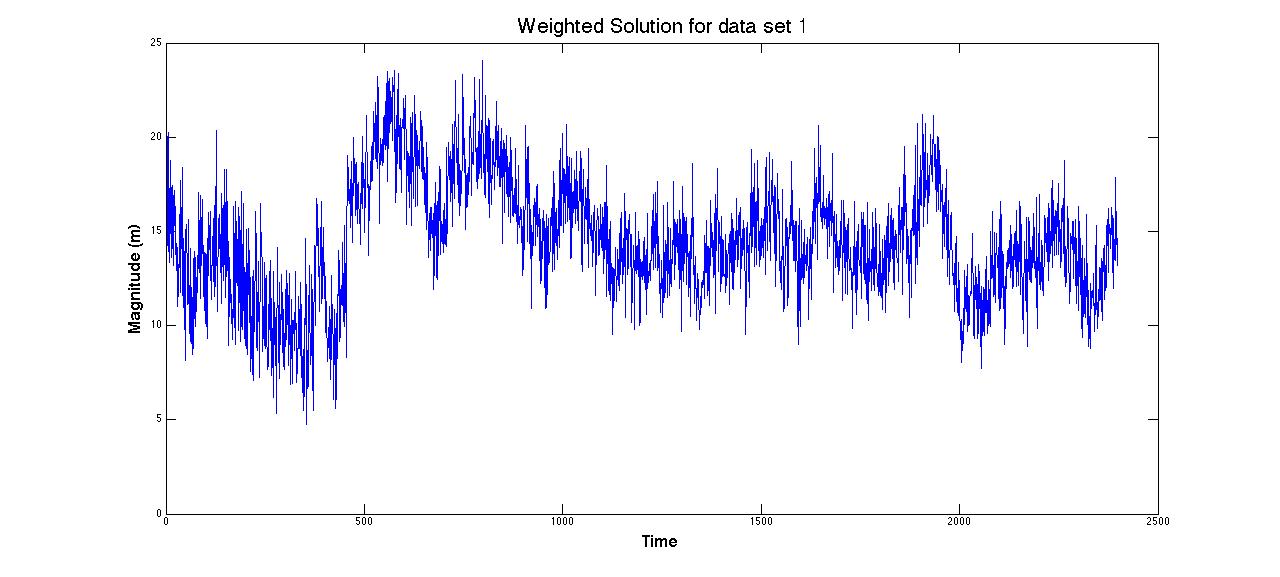


Figure : Weighted solution to data set 1

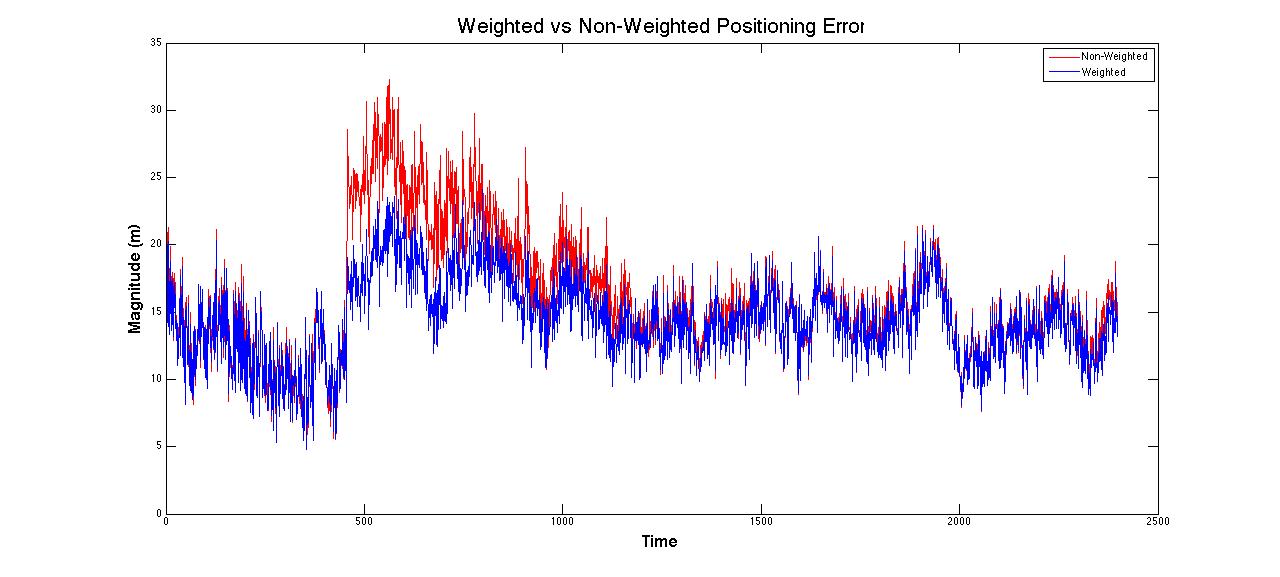


Figure : Weighted Vs Non-Weighted solution for data set 1

# Appendix

## Appendix A

### Main Function

[] = GPS('data\_name.mat')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%% Inital Inputs %%%%%%%%%%%%%%%%%%%

A='data\_name.mat';

clear all %% Clear data

clc %% Clear workspace

load('A') % Load Data

z=1; j=1; y=1; %% Set counters

Orgin = nomXYZ; %% Orgin for converting to ENU

sigma\_URE = sigma\_URE^2\*eye(4);

Length = length(nSat);

Speed\_of\_Light = 299792458;

[Sat\_XYZ,Pseudorange,Computed\_Pseudorange,Unit\_Vector,XYZ\_Estimate,PDOP,GDOP,TDOP,W,delta\_x,delta\_x\_Weight] = Memory(nSat,Length);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%% END INITIAL INPUTS %%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%% MAIN LOOP %%%%%%%%%%%%%%%%%%%%%%

for i = 1:Length-1;

z=(i-1)+1;

[Sat\_XYZ,Pseudorange] = Seperate\_Data(nSat,satsXYZ,prData,z);

[Computed\_Pseudorange,Unit\_Vector] = Computed(Sat\_XYZ,nomXYZ,clockBiasNom,nSat,Speed\_of\_Light,z);

[P,R,EL,W] = ELandAV(z,nSat,truthXYZ,Unit\_Vector,W);

[G,deltaRho,delta\_x,delta\_x\_Weight,XYZ\_Estimate,XYZ\_Estimate\_Weighted] = ECEF\_Estimate(nSat,Unit\_Vector, Computed\_Pseudorange, prData,nomXYZ,z,W);

H(:,:,z) = sigma\_URE\*(inv(G(:,:,z)'\*G(:,:,z)));

PDOP(:,z) = (sqrt(((H(1,1,z)^2)+(H(2,2,z)^2)+(H(3,3,z)^2))))/(sigma\_URE(1,1));

GDOP(:,z) = (sqrt(((H(1,1,z)^2)+(H(2,2,z)^2)+(H(3,3,z)^2) +(H(4,4,z)^2))))/((sigma\_URE(1,1)));

TDOP(:,z) = (sqrt((H(4,4,z)^2)))/(sigma\_URE(1,1));

Error\_ECEF(:,z) = norm(XYZ\_Estimate(:,z)-truthXYZ(:,z));

Error\_ECEF\_Weighted(:,z) = norm(XYZ\_Estimate\_Weighted(:,z)-truthXYZ(:,z));

[Estimated\_ENU(:,:,z),R\_Estimate(:,:,z)] = xyz2enu(XYZ\_Estimate(:,z)',Orgin);

[True\_ENU(:,:,z),R\_True(:,:,z)] = xyz2enu(truthXYZ(:,z)',Orgin);

Error\_ENU(:,z) = norm(Estimated\_ENU(:,:,z)-True\_ENU(:,:,z));

Clock\_Bias\_Estimate(:,z) = clockBiasNom-1\*delta\_x(4);

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%% End Main LOOP%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%% PLOTS %%%%%%%%%%%%%%%%%%%%%%%%%%%

figure()

plot(Error\_ECEF)

title('ERROR\_ECEF')

figure()

plot(Error\_ENU)

title('Error\_ENU')

figure()

plot(Error\_ECEF\_Weighted)

title('ECEF WEIGHTED')

figure ()

plot(Error\_ECEF-Error\_ENU)

title('Error\_ECEF - Error\_ENU')

figure ()

plot(PDOP,'x')

title('PDOP')

figure()

plot(TDOP,'x')

title('TDOP')

figure()

plot(GDOP,'x')

title('GDOP')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% END PLOTS %%%%%%%%%%%%%%%%%%%%

## Appendix B

### Elevation and Azimuth Function

function [P,R,EL,W] = ELandAV(z,nSat,truthXYZ,Unit\_Vector,W)

P(1,z) = sqrt(((truthXYZ(1,z))^2)+((truthXYZ(2,z))^2));

R(1,z) = sqrt(((truthXYZ(1,z))^2)+((truthXYZ(2,z))^2)+((truthXYZ(3,z))^2));

East(z,:) = [(-truthXYZ(2,z)^2)/P(1,z) (truthXYZ(1,z)^2)/P(1,z) 0 ];

North(z,:) = [(((-truthXYZ(1,z))\*(-truthXYZ(3,z)))/(P(1,z)\*R(1,z))) (((-truthXYZ(2,z))\*(-truthXYZ(3,z)))/(P(1,z)\*R(1,z))) (P(1,z)/R(1,z))];

Up(z,:) = [((truthXYZ(1,z))/(R(1,z))) (truthXYZ(2,z)/(R(1,z))) (truthXYZ(3,z)/R(1,z))];

Up\_Ones = ones(nSat(z),3);

Up\_z(:,:,z)=[Up(z,1)\*Up\_Ones(:,1) Up(z,2)\*Up\_Ones(:,2) Up(z,3)\*Up\_Ones(:,3)];

k=1;

for k=1:nSat(z)

EL(:,:,z) = dot(Unit\_Vector(:,:,z),Up\_z(:,:,z),2);

W(k,k,z)=1/(diag(EL(k,:,z)));

k=k+1;

end

end

## Appendix C

### Computed Function

function [Computed\_Pseudorange,Unit\_Vector] = Computed(Sat\_XYZ,nomXYZ,clockBiasNom,nSat,c,z);

for j=1:nSat(z);

if j == 1;

Computed\_Pseudorange(1,1,z)=norm(Sat\_XYZ(1,:,z)-nomXYZ)+clockBiasNom\*c;

Unit\_Vector(1,:,z)=(Sat\_XYZ(1,:,z)-nomXYZ)/norm(Sat\_XYZ(1,:,z)-nomXYZ);

else

y=j+1;

Computed\_Pseudorange(y-1,1,z)=norm(Sat\_XYZ(y-1,:,z)-nomXYZ)+clockBiasNom\*c;

Unit\_Vector(y-1,:,z)=(Sat\_XYZ(y-1,:,z)-nomXYZ)/norm(Sat\_XYZ(y-1,:,z)-nomXYZ);

end

end

## Appendix D

### DOP Function

function [H,PDOP,GDOP,TDOP] = DOP(sigma\_URE,G,z)

H(:,:,z) = sigma\_URE\*(inv(G(:,:,z)'\*G(:,:,z)));

PDOP(:,z) = sqrt(((H(1,1,z)^2)+(H(2,2,z)^2)+(H(3,3,z)^2)));

GDOP(:,z) = sqrt(((H(1,1,z)^2)+(H(2,2,z)^2)+(H(3,3,z)^2) +(H(4,4,z)^2)));

TDOP(:,z) = sqrt((H(4,4,z)^2));

## Appendix E

### ECEF Estimate

function [G,deltaRho,delta\_x,delta\_x\_Weight,XYZ\_Estimate,XYZ\_Estimate\_Weighted] = ECEF\_Estimate(nSat,Unit\_Vector, Computed\_Pseudorange, prData,nomXYZ,z,W)

G(1:nSat(z),:,z) = horzcat(Unit\_Vector(1:nSat(z),:,z),ones(nSat(z),1));

deltaRho(1:nSat(z),:,z) = Computed\_Pseudorange(1:nSat(z),:,z)-prData(1:nSat(z),z);

delta\_x(:,:,z) = inv(G(:,:,z)'\*W(1:nSat(z),1:nSat(z),z)\*G(:,:,z))\*G(:,:,z)'\*W(1:nSat(z),1:nSat(z),z)\*deltaRho(:,:,z);

delta\_x\_Weight(:,:,z) = inv(G(:,:,z)'\*G(:,:,z))\*G(:,:,z)'\*deltaRho(:,:,z);

XYZ\_Estimate(:,z) = nomXYZ'+delta\_x(1:3,:,z);

XYZ\_Estimate\_Weighted(:,z) = nomXYZ'+delta\_x\_Weight(1:3,:,z);

## Appendix F

### Memory Function

function [Sat\_XYZ,Pseudorange,Computed\_Pseudorange,Unit\_Vector,XYZ\_Estimate,PDOP,GDOP,TDOP,W,delta\_x,delta\_x\_Weight] = Memory(nSat,Length)

Sat\_XYZ = zeros(max(nSat),3,Length); %% Allocate memory

Pseudorange = zeros(max(nSat),Length); %% Allocate memory

Computed\_Pseudorange = zeros(max(nSat),1,Length); %% Allocate memory

Unit\_Vector = zeros(max(nSat),3,Length); %% Allocate memory

XYZ\_Estimate = zeros(3,Length); %% Allocate memory

PDOP = zeros(1,Length); %% Allocate memory

GDOP = zeros(1,Length); %% Allocate memory

TDOP = zeros(1,Length); %% Allocate memory

W=zeros(max(nSat),max(nSat),Length); %% Allocate memory

delta\_x=zeros(4,1,Length);

delta\_x\_Weight=zeros(4,1,Length);