LAB 2: ESTIMATION

ENGO 585: WIRELESS LOCATION

Author: Juan Carlos Terrazas Borbon

Student ID: 10130921

Due Date: February 1, 2018

Submitted on: February 1, 2018

University of Calgary

Department of Geomatics Engineering

Table of Contents

Table of Figures

Tables list

Introduction

The following report contains the methodology and analysis perform in this lab in order to achieve the requirements from the lab handout. The purpose of this lab is to “review non-linear least-squares estimation and related concepts” according to [1]. This lab involves performing Parametric Least Squares, Sequential Least Squares and Kalman Filtering. Some of the formulas for this lab were extracted from the lecture notes of the course ENGO 585 and other ones were extracted from the course ENGO 361. The necessary coding of this lab was written using Matlab language and software.

Methodology

Task 1: Parametric Least Squares

2-D Solution for epoch

To perform a parametric least squares adjustment on each epoch, it was needed to first obtain the weight matrix(P) and design matrix(A) of each epoch using the following formulas:

$$P = \sigma_0^2 * C_l^{-1}$$

$$where: C_l = diag(\sigma_1^2 + \ \sigma_2^2 \dots \sigma_n^2)$$

$$A_{4,2} = \begin{bmatrix} \frac{\dot{x} - x_{t_1}}{r_i} & \frac{\dot{y} - y_{t_1}}{r_i} \\ \frac{\dot{x} - x_{t_2}}{r_i} & \frac{\dot{y} - y_{t_2}}{r_i} \\ \frac{\dot{x} - x_{t_3}}{r_i} & \frac{\dot{y} - y_{t_3}}{r_i} \\ \frac{\dot{x} - x_{t_4}}{r_i} & \frac{\dot{y} - y_{t_4}}{r_i} \end{bmatrix}$$

$$where: r_i = range\ observation \quad and \quad t_i = target\ number$$

Since the lab handout does not contains any information with respect to the standard deviation of the observations and an apriori value, it was then assumed that for this lab the covariance matrix of the observations is equal to identity and the apriori value is equal to 1, as shown in the equation below.

$$Assumption: C_l = I \ \ and \ \ \sigma_0^2 = 1 \ \ so\ P_{4,4} = I_{4,4}$$

Once the weight matrix and the design matrix were calculated, it was needed to iterate the following formulas to obtain a correct disclosure ($\dot{w}$), misclosure ($\hat{\delta}$) and the unknown coordinates of each epoch ($\hat{x}$). The threshold for this iteration was that any value of $\hat{\delta}$ matrix should not exceed 0.0001 meters.

$$\dot{w_{4,1}} = f(\dot{x}) - l$$

$$\hat{\delta}_{1,2} = \ -(A^T * P * A)^{-1} * A^T * P * \dot{w}$$

$$\hat{x}_{1,2} = \dot{x} + \hat{\delta}$$

Batch Parametric Least Squares

Performing a Batch Parametric Least Squares, involves using the same formulas as in the first part in Task 1, but performing it for several epochs at once. This means that instead of performing the least square solution for each epoch, you perform least square procedure once taking multiple epochs as observations. For this lab performing a batch works out, since the data shows that there are some static periods on it.

After looking at the data and the results of the first part of Task 1 it was found that there was only 1 static period in the data and it was from the beginning all the way to epoch 50. Then to perform a Least Squares solution a Design, Weight and disclosure matrices need to be created. Since the first 50 epochs are taken as observations, it means that there are 200 observations now (50 epochs x 4 observation per epoch).

The weight matrix is equal to Identity so for this matrix it means that it only grows but keeps it values. For the design and disclosure matrices it is needed to push back every single design matrix and disclosure matrix for every single epoch into 1 matrix of each. The final sizes of the matrices are:

$$A_{200,2}\ P_{200,200}\ \ \dot{w_{200,1}}$$

Having the Design, Weight and disclosure matrices, then the unknow coordinates can be obtained using the same formulas as above:

$$\hat{\delta}_{1,2} = -(A^T * P * A)^{-1} * A^T * P * \dot{w}$$

$$\hat{x}_{1,2} = \dot{x} + \hat{\delta}$$

Residuals

To obtain the residuals of all the range measurements, the following formula is needed:

$$\hat{r} = A * \hat{\delta} + \dot{w}$$

Once the residuals of all the observations are obtained, then the residual for certain measurements (ranges measurements to Target 1) can be extracted and then plotted as requested in [1].

2D Error ellipse

To obtain the error ellipse for either a single observation or the batch observation, it is needed to first obtain the aposteriori value of your observations. To do so the following formula was used in this lab:

$$\hat{\sigma}_0^2 = \frac{\hat{r}^T * P * \hat{r}}{n - u}$$

Once the aposteriori value is obtained, then it is needed to compute the covariance matrix of the unknowns. To compute the covariance matrix the following formula is needed:

$$\hat{C}_x = \hat{\sigma}_0^2 * (A^T * P * A)^{-1} = \hat{\sigma}_0^2 * \mathrm{N}^{-1}$$

The covariance matrix of the unknowns for this lab should look like the equation below, where it shows that the top left and bottom right values are the variance of the east and north obtained values (respectively).

$$\hat{C}_x = \begin{pmatrix} \sigma_e^2 & \sigma_{en} \\ \sigma_{en} & \sigma_n^2 \end{pmatrix}$$

Using the variance and standard deviation values obtained from the unknown's covariance matrix, then the Semi-Major (a) axis, Semi-Minor axis(b) and the ellipse orientation can be computed using the formulas that can be found on Figure 1.

$$a = \left[\frac{1}{2}\left(\sigma_e^2 + \sigma_n^2\right) + \left[\frac{1}{4}\left(\sigma_e^2 - \sigma_n^2\right)^2 + \sigma_{en}^2 \right]^{\frac{1}{2}} \right]^{\frac{1}{2}}$$

$$b = \left[\frac{1}{2}\left(\sigma_e^2 + \sigma_n^2\right) - \left[\frac{1}{4}\left(\sigma_e^2 - \sigma_n^2\right)^2 + \sigma_{en}^2 \right]^{\frac{1}{2}} \right]^{\frac{1}{2}}$$

$$\theta = \frac{1}{2} \arctan \left[\frac{2\sigma_{en}}{\sigma_e^2 - \sigma_n^2} \right]$$

Figure 1: Semi-Major and Semi-minor axes, and orientation formulas extracted from [3]

Once having the Semi-Major (a) axis, Semi-Minor axis(b), these values needed to be multiply by a factor which is dependent to the confidence level of the error ellipse, which in this case the factor was 2.45 which means is equal to 95% confidence level. Using a, b and the orientation, then the error ellipse can be plotted. Using the procedure above the error ellipse was calculated for a single epoch and the batch solution. For the single epoch, it was selected to plot the error ellipse of the epoch 150 (this selection was randomly).

Task 2: Summation of Normals and Sequential Least Squares

Summation of Normals of Static Data

To compute the Summation of Normals model for the static data (batch data in task 1), it was required to compute the Design and Disclosure matrix for each of the 50 epochs selected and then use the following formula in order to compute the unknown coordinates. The formulas below where obtained from [3].

$$\delta = -\sum_{i=1}^{n} (N_i)^{-1} (u_i)$$

$$where: N_i = A_i^T * P_i * A_i \;\; and \;\; u_i = A_i^T * P_i * w_i$$

Sequential Least Squares of Static Data

To compute the Sequential Least Squares, it is also needed to compute the Design and Disclosure matrices, like the Summation of Normals, but also it requires some more computations. The following procedure explains how the sequential least square was performed in this task (All formulas were obtained from [3]).

1. For the first epoch it is needed to compute the delta matrix the following way:

$$\delta(-) = \; -(A_1^T * P_1 * A_1)^{-1} * (A_1^T * P_1 * w_1)$$

2. Then the Design Matrix and the Disclosure matrix of the next epoch are computed
3. Compute the Gain Matrix:

$$K = N_1^{-1} * A_2^T \left(C_{l_2} + A_2 * N_1^{-1} * A_2^T\right)^{-1}$$

4. Compute the updated delta value:

$$\delta(+) = \delta(-) - K(A_2 * \delta(-) + w_2)$$

5. Update Covariance Matrix:

$$For\ epoch\ 1: C_{\delta(-)} = N^{-1} = (A_1^T * P_1 * A_1)^{-1}$$

$$C_{\delta(+)} = C_{\delta(-)} - K * A_2 * C_{\delta(-)}$$

6. Repeat steps 2 to 5 for all the remaining observations considering that:

$$\delta(-)\ will\ be\ become\ \delta(+)\ and\ C_{\delta(+)}\ will\ become\ C_{\delta(-)}\ in\ next\ epoch$$

Task 3: Kalman Filter

Sequential Least Squares of all data

For this section of the lab it is requested to perform sequential least squares of all the observations (static and kinematic data), using the same formulas and procedure as in the section "Sequential Least Squares of Static Data". To do so the only small modification, which involves adding the other 100 observations to the observations list (in other words increasing the for-loop number).

Kalman Filter

For this section of the lab it was required to create an easy way of Kalman Filter by grabing the Sequential Least Square and adding process noise (Q) to the covariance matrix $C_{\delta(+)}$. Which mean the same process must be done as the first part of Task 3 but in step 5 of the Sequential Least Squares procedure it is needed to ask the random variable Q. As requested from [1], it is only needed to try different values of Q and plot them to analyze.

Results and Analysis

Task 1: Parametric Least Squares

2-D Solution for epoch

Figure 2 contains a graphic representation of the obtained coordinates for all the epochs, and the location of the 4 given targets. The static area can be visualized since close to the coordinates (50, 50) the plotting of points seems to be denser.

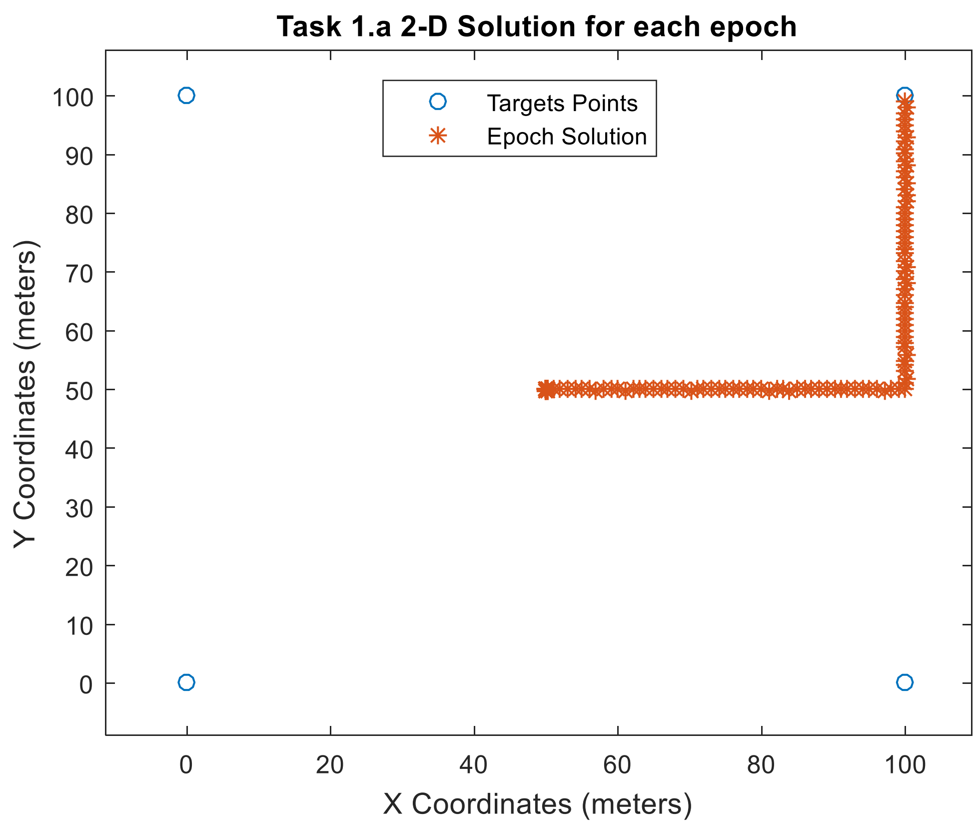

Figure 2: 2D solution plot of all epochs

Batch Parametric Least Squares

The obtained coordinates of the Batch Parametric Least Squares can be found on Table 1. These results are used in the error ellipse section below, as well as will be analyze with more detail in Task 2. A visual representation can be found in Figure 4.

Table 1: Batch Parametric Least Squares Results

	Northing	Easting
$\hat{x}$	50.0099	49.9994

Residuals

Figure 3 has a graphical representation of the residuals obtained from the observations to Target 1 from the user. The plot shows how the variation of residuals go from almost -0.2 to 0.3. Since the obtained A posteriori value obtained from the batch results is: 0.0102, which is small and no large so the measurements does not contain big blunders which also means that this residuals are proper residuals.

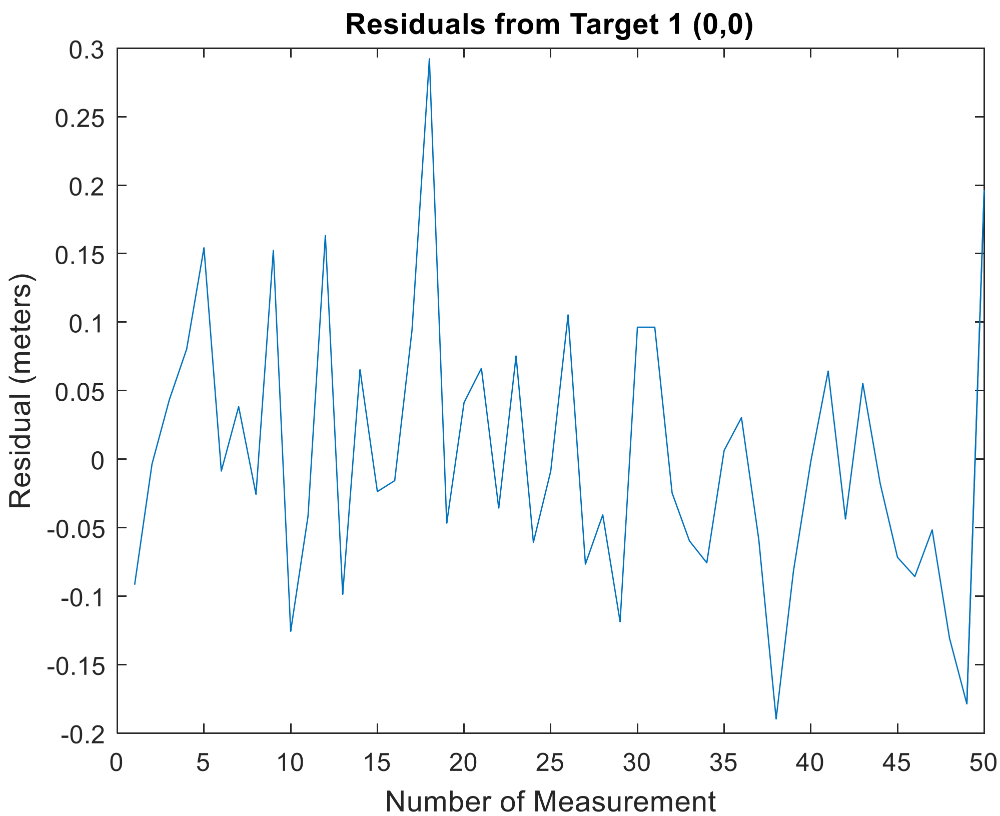

Figure 3: Residuals graph of observation to Target 1

2D Error Ellipse of Batch Solution

Figure 4 show the error ellipse of the batch solution at 95% confidence level, the solution of each epochs 1 to 50 and the solution of the batch parametric least squares. It is found that the error ellipse does contains some of the other points but most of the points are outside this error ellipse. Table 2 contains some of the variables obtained to calculate the error ellipse. Also, the Covariance matrix of the unknows can be found below.

Table 2: Batch Error Ellipse Detailed values

Variables	Values
Aposteriori	0.0102
Semi-Major(a)	0.024769
Semi-Minor(b)	0.024768
Orientation(Azimuth)	-0.7848

$$\hat{C}_x = \begin{bmatrix} 0.000102 & -1.69E-09 \\ -1.69E-09 & 0.000102 \end{bmatrix}$$

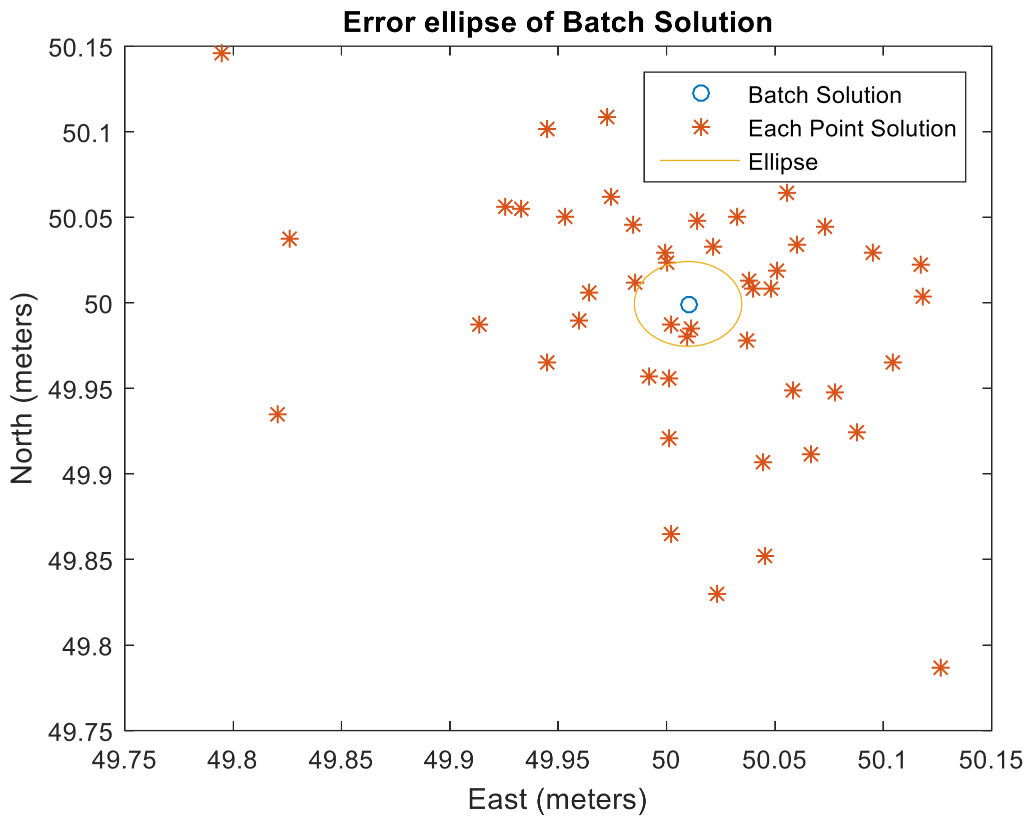

Figure 4: Error Ellipse of Batch Solution

2D Error Ellipse of a Single Epoch

Figure 5 show the error ellipse of the epoch 150 from the least squares parametric solution at 95% confidence level. It can be visualized that the error ellipse is small and that it only covers the center point. This is because the error is small and since this section of the data is dynamic, the distance between points are higher than when data is static. Table 3 contains some of the variables obtained to calculate the error ellipse. Also, the Covariance matrix of the unknows can be found below.

Table 3: Epoch Error Ellipse Detailed values

Variables	Values
Aposteriori	0.0103
Semi-Major(a)	0.21567
Semi-Minor(b)	0.15010
Orientation(Azimuth)	-0.3551

$$\hat{C}_x \begin{bmatrix} 0.007265958 & -1.30E-03 \\ -1.30E-03 & 0.004236763 \end{bmatrix}$$

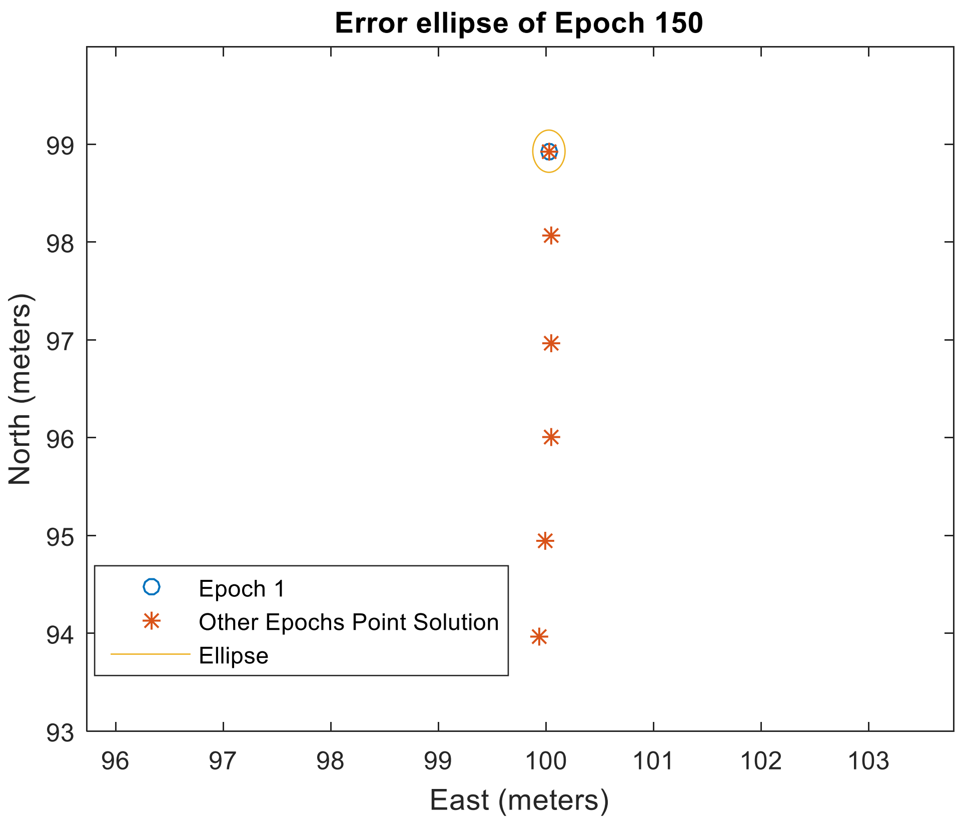

Figure 5: Error Ellipse of Epoch 150

Task 2: Summation of Normals and Sequential Least Squares

As shown in Table 4 and Table 5 below, both methods obtained the same results for the unknown coordinates of the static data. These coordinates are equal to the ones obtained in the batch parametric least squares. The reasoning for this is due to the fact that the three methods have a relation(one is derives from the other) and since they all are using the same stationary data and they are using all the same information out of it.

Table 4: Summation of Normals of Static Data Results

	Northing	Easting
$\hat{x}$	50.0099	49.9994

Table 5: Sequential Least Squares of Static Data Results

	Northing	Easting

$\hat{x}$	50.0099	49.9994

Task 3: Kalman Filter

Sequential Least Squares of all data

From Figure 6, it can be visualized that the same results were obtained for the static data, but the kinematic section of the data had some problems. These problems happen since they are all dependent to the estimated first value which in this case that value was 50 E, 50 N.

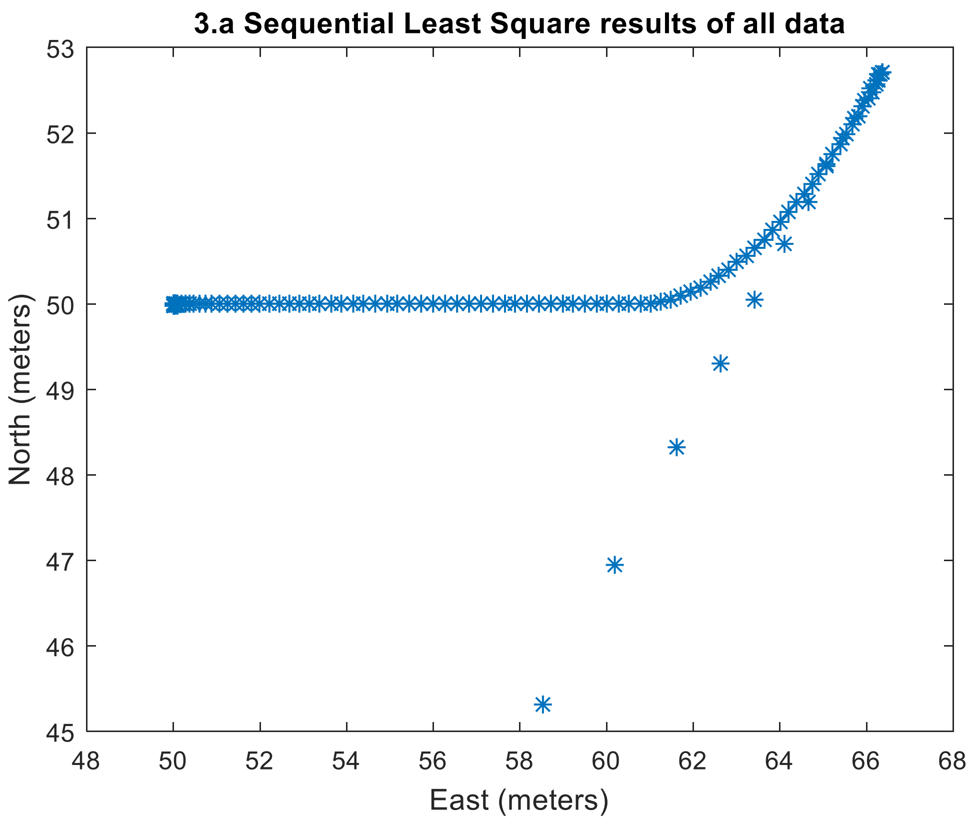

Figure 6: Sequential Least Squares of all data without modifications

The following lines of code were added after the static period passes so that the estimated value is updated before computing the disclosure and design matrices of the next epoch.

```
if i >50
    est_coords = [est_coords(1) + delta(1),est_coords(2) + delta(2)];
end
```

As shown in Figure 7, this occasioned a different result that is closer to the one in task 1 but still had it issues since in some section the values were dragged more to the right than they should had been.

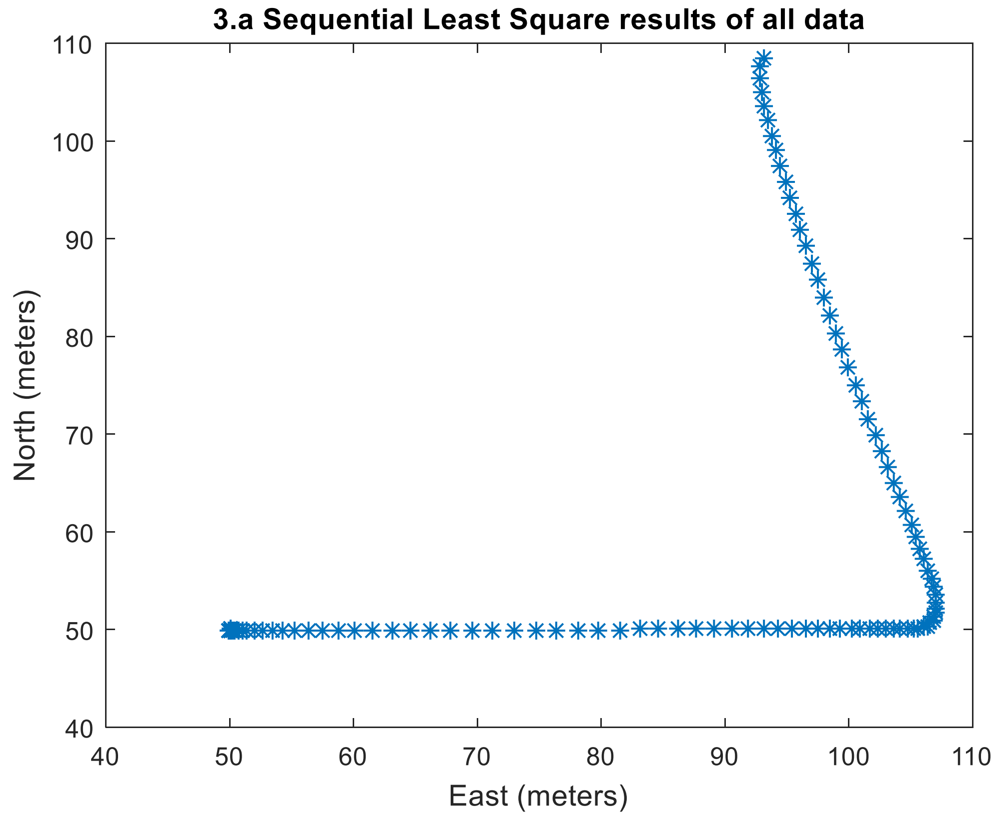

Figure 7: Sequential Least Squares of all data with modifications

Kalman Filter

After trying multiple types of Q values and performing the Kalman Filter with the modified Sequential Least Squares, it can be seen that the higher the Q value is the more effect it has on the value which makes that the trust on the predictions is pretty low so for the same reason this noise as it increases it makes the results smaller in this situation. Please refer to Figure 8, Figure 9 and Figure 10 for three results of modified Q values.

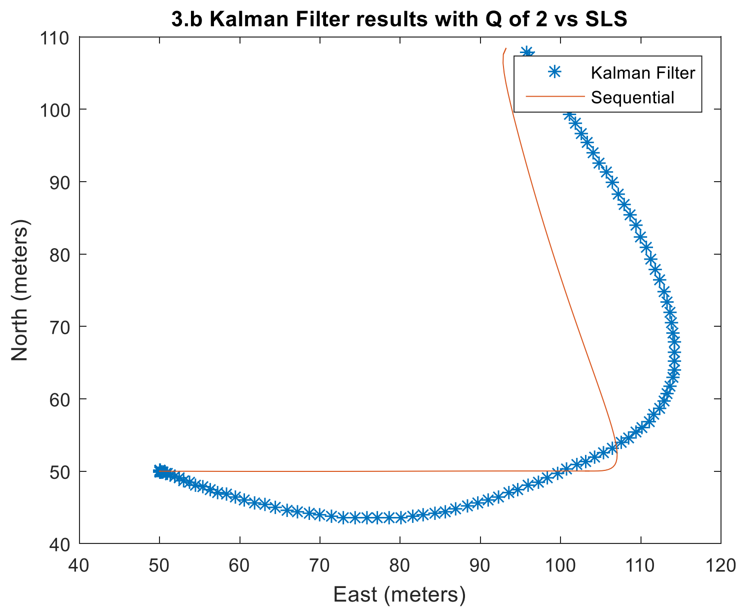

Figure 8: Kalman Filter with Q =2

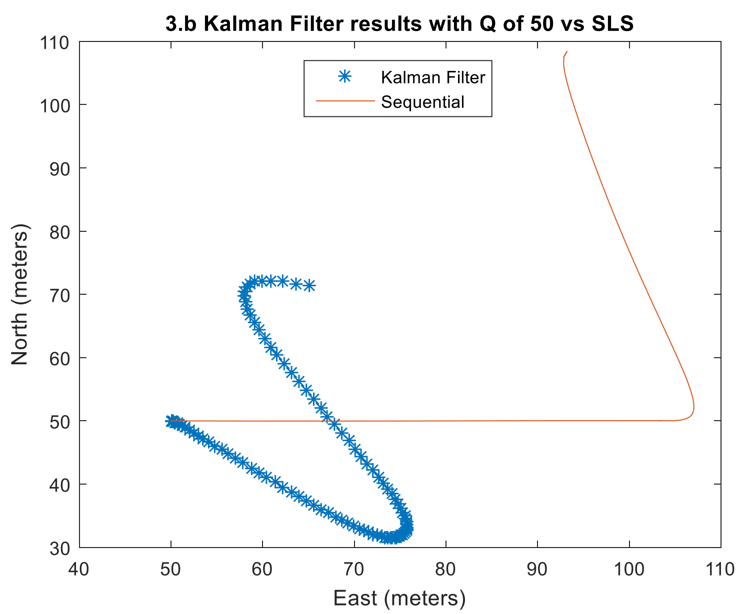

Figure 9: Kalman Filter with Q =50

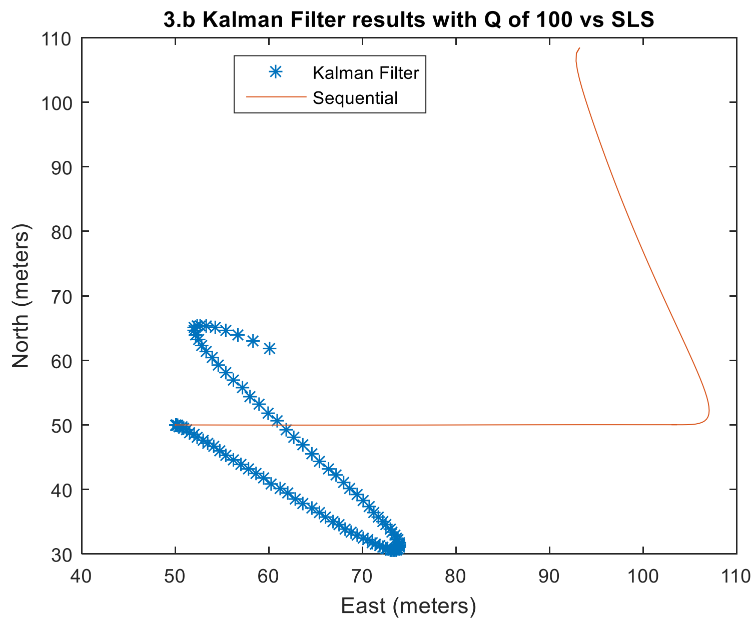

Figure 10:: Kalman Filter with Q =100

References

[1] O'Keefe, Kyle. (2018) *Lab #2: Estimation* [PDF] University of Calgary University of Calgary. Retrieved from: https://d2l.ucalgary.ca

[2] O'Keefe, Kyle. (2018) *ENGO 585 Notes: Chapter 2 Estimation for Navigation* University of Calgary. Retrieved from: https://d2l.ucalgary.ca

[3] Sheimy, El. (2014) *Chapter 8: Combination of Models*, [PDF] University of Calgary. Retrieved from: https://d2l.ucalgary.ca

Appendix A: Matlab Code

```
% Author: Juan Carlos Terrazas Borbon
% Last Update: 2018-02-01
% Course: ENGO 585
% Lab: 2

% ---------------------Purpose of Code-----------------------------------
% The purpose of this code is to able to perform the 3 task required to
% perfom in the lab handout which involves Parametric Least Squares
% adjustment, Summation of Normal, Sequential LS and Kalman Filter

% Clear variables, close figure and command
clc
clear all
close all

% Read the file with data and extract its data
ranges = load('Lab2data.txt');

% Given Target Coordinates in meters
 targets = [0,0; 100,0; 100,100; 0,100];

%% Task 1: Batch Parametric Least Squares
est_coords = [50, 50];
P = diag(ones(4,1));

% 1.a Compute 2-D Solution for each epoch----------------------------------
x_hat_1_a = zeros(150,2);
for i = 1 : length(ranges)
    thres = 0;
    while thres == 0
        % Obtain the A matrix
        A = zeros(4,2);
        for j = 1 : 4
            A(j, 1) = (est_coords(1) - targets(j, 1)) / ranges(i, j + 1);
            A(j, 2) = (est_coords(2) - targets(j, 2)) / ranges(i, j + 1);
        end

        % Compute w Matrix
        w= zeros(4,1);
        for j = 1 : 4
            w(j, 1) = sqrt((targets(j, 1) - est_coords(1))^2 + ...
                (targets(j, 2) - est_coords(2))^2) - ranges(i, j + 1);
        end

        % Compute N Matrix and obtain the delta values
        N = A' * P * A;
        delta = -1 * inv(N) * A' * P * w;

        % check
        if abs(delta(1)) < 0.0001 && abs(delta(2)) < 0.0001
           thres = 1;
        else
           est_coords = [est_coords(1) + delta(1),est_coords(2) + delta(2)];
```

```
        end
    end
    x_hat_1_a(i,:) = [est_coords(1) + delta(1),est_coords(2) + delta(2)];
    est_coords = [est_coords(1) + delta(1),est_coords(2) + delta(2)];
end

figure
plot(targets(:,1), targets(:,2),'o')
hold on
plot(x_hat_1_a(:,1), x_hat_1_a(:,2),'*');
hold off
title('Task 1.a 2-D Solution for each epoch')
xlabel('X Coordinates (meters)')
ylabel('Y Coordinates (meters)')
legend('Targets Points', 'Epoch Solution')

% 1.b Batch Solution-------------------------------------------------------
est_coords = [50, 50];
A_b = zeros(50*4,2);
w_b= zeros(50*4,1);
P_b = diag(ones(50*4,1));
thres = 0;
while thres == 0
    for i = 1: 50
        % Obtain the A matrix
        for j = 1 : 4
            A_b((i-1)*4+j, 1) = (est_coords(1) - targets(j, 1)) / ranges(i, j
+ 1);
            A_b((i-1)*4+j, 2) = (est_coords(2) - targets(j, 2)) / ranges(i, j
+ 1);
        end

        % Compute w Matrix
        for j = 1 : 4
            w_b((i-1)*4+j, 1) = sqrt((targets(j, 1) - est_coords(1))^2 + ...
                (targets(j, 2) - est_coords(2))^2) - ranges(i, j + 1);
        end
    end
    % Compute N Matrix and obtain the delta values
    N = A_b' * P_b * A_b;
    delta = -1 * inv(N) * A_b' * P_b * w_b;

    % Check for delta and if threshold passes obtain coordinates
    if abs(delta(1)) < 0.0001 && abs(delta(2)) < 0.0001
        thres = 1;
        x_hat_1_b = [est_coords(1) + delta(1),est_coords(2) + delta(2)];
    else
        est_coords = [est_coords(1) + delta(1),est_coords(2) + delta(2)];
    end
end

% 1.c Plot Residuals-------------------------------------------------------
res = A_b * delta + w_b;

figure
plot(1:1:50, res(1:4:200,1))
```

```
title('Residuals from Target 1 (0,0)')
xlabel('Number of Measurement')
ylabel('Residual (meters)')

Apos =(res' * P_b * res)/(200-2);%aposteriori

C_x = Apos * inv(A_b' * P_b * A_b);%covariance matrix

% 1.d Error Ellipse--------------------------------------------------------
a = sqrt(0.5 * (C_x(1,1) + C_x(2,2)) + sqrt((1/4) * (C_x(1,1) - C_x(2,2))^2
...
    + C_x(1,2)^2))* 2.45;

b = sqrt(0.5 * (C_x(1,1) + C_x(2,2)) - sqrt((1/4) * (C_x(1,1) - C_x(2,2))^2
...
    + C_x(1,2)^2)) * 2.45;

azimuth = 0.5* atan((2 * C_x(1,2))/(C_x(1,1) - C_x(2,2)));

alpha = (azimuth) * pi/180;
t = [-0.01:0.01:2*pi];

%Obtain ellipse coordinates

ellipse_x =  (sin(alpha)*(a*cos(t)) - cos(alpha)*(b*sin(t))) + x_hat_1_b(1);
ellipse_y =  (cos(alpha)*(a*cos(t)) + sin(alpha)*(b*sin(t))) + x_hat_1_b(2);

figure
plot(x_hat_1_b(1),x_hat_1_b(2), 'o')
hold on
plot(x_hat_1_a(1:50,1),x_hat_1_a(1:50,2), '*')
plot(ellipse_x, ellipse_y)
title('Error ellipse of Batch Solution')
legend('Batch Solution', 'Each Point Solution', 'Ellipse')
xlabel('East (meters)')
ylabel('North (meters)')

% 1.d Error Ellipse of a single point--------------------------------------
res = A * delta + w;%residuals
Apos =(res' * P * res)/(4-2);%aposteriori
C_x = Apos * inv(A' * P * A);%covariance matrix

a = sqrt(0.5 * (C_x(1,1) + C_x(2,2)) + sqrt((1/4) * (C_x(1,1) - C_x(2,2))^2
...
    + C_x(1,2)^2))* 2.45;

b = sqrt(0.5 * (C_x(1,1) + C_x(2,2)) - sqrt((1/4) * (C_x(1,1) - C_x(2,2))^2
...
    + C_x(1,2)^2)) * 2.45;

azimuth = 0.5* atan((2 * C_x(1,2))/(C_x(1,1) - C_x(2,2)));

alpha = (azimuth) * pi/180;
t = [-0.01:0.01:2*pi];
```

```
%Obtain ellipse coordinates
ellipse_x =  (sin(alpha)*(a*cos(t)) - cos(alpha)*(b*sin(t))) +
x_hat_1_a(150,1);
ellipse_y =  (cos(alpha)*(a*cos(t)) + sin(alpha)*(b*sin(t))) +
x_hat_1_a(150,2);

figure
plot(x_hat_1_a(150,1),x_hat_1_a(150,2), 'o')
hold on
plot(x_hat_1_a(1:150,1),x_hat_1_a(1:150,2), '*')
plot(ellipse_x, ellipse_y)
title('Error ellipse of Epoch 150')
legend('Epoch 1', 'Other Epochs Point Solution', 'Ellipse')
xlabel('East (meters)')
ylabel('North (meters)')

%% Task 2: Summation of Normals and Sequential LS
% 2.a Summation of Normal of 1.b
est_coords = [50, 50];
N = zeros(2,2);
u = zeros(2,1);

for i = 1: 50
    % Obtain the A matrix
    A = zeros(4,2);
    for j = 1 : 4
        A(j, 1) = (est_coords(1) - targets(j, 1)) / ranges(i, j + 1);
        A(j, 2) = (est_coords(2) - targets(j, 2)) / ranges(i, j + 1);
    end

    % Compute N Matrix
    N = N + (A' * P * A);

    % Compute w Matrix
    w= zeros(4,1);
    for j = 1 : 4
        w(j, 1) = sqrt((targets(j, 1) - est_coords(1))^2 + ...
            (targets(j, 2) - est_coords(2))^2) - ranges(i, j + 1);
    end

    % Compute U Matrix
    u = u + (A'*P*w);
end

delta = -1 * inv(N)* u;

x_hat_2_a = [est_coords(1) + delta(1), est_coords(2) + delta(2)];

% 2.b Sequential Least Squares of 1.b
est_coords = [50, 50];
for i = 1:50
    % Obtain the A matrix
    A = zeros(4,2);
    for j = 1 : 4
```

```
        A(j, 1) = (est_coords(1) - targets(j, 1)) / ranges(i, j + 1);
        A(j, 2) = (est_coords(2) - targets(j, 2)) / ranges(i, j + 1);
    end

    % Compute w Matrix
    w= zeros(4,1);
    for j = 1 : 4
        w(j, 1) = sqrt((targets(j, 1) - est_coords(1))^2 + ...
            (targets(j, 2) - est_coords(2))^2) - ranges(i, j + 1);
    end

    if i==1
        % Compute N Matrix and obtain the delta values for first observation
        N = A' * P * A;
        delta = -1 * inv(N) * A' * P * w;

    else
        % Sequential LS
        K = inv(N) * A' * inv(P + A*inv(N)*A');
        delta = delta - K*(A*delta + w);
        N = inv(inv(N) - K*A*inv(N));
    end
end
x_hat_2_b = [est_coords(1) + delta(1),est_coords(2) + delta(2)];

%% Task 3: Kalman Filtering
% 3.a Sequential Least Squares of whole data
est_coords = [50, 50];
for i = 1:150
    % Obtain the A matrix
    A = zeros(4,2);
    for j = 1 : 4
        A(j, 1) = (est_coords(1) - targets(j, 1)) / ranges(i, j + 1);
        A(j, 2) = (est_coords(2) - targets(j, 2)) / ranges(i, j + 1);
    end

    % Compute w Matrix
    w= zeros(4,1);
    for j = 1 : 4
        w(j, 1) = sqrt((targets(j, 1) - est_coords(1))^2 + ...
            (targets(j, 2) - est_coords(2))^2) - ranges(i, j + 1);
    end

    if i==1
        % Compute N Matrix and obtain the delta values for first observation
        N = A' * P * A;
        delta = -1 * inv(N) * A' * P * w;
        x_hat_3_a(i,:) = [est_coords(1) + delta(1),est_coords(2) + delta(2)];
    else
        % Sequential LS
        K = inv(N) * A' * inv(P + A*inv(N)*A');
        delta = delta - K*(A*delta + w);
        N = inv(inv(N) - K*A*inv(N));
        x_hat_3_a(i,:) = [est_coords(1) + delta(1),est_coords(2) + delta(2)];
```

```
        % Comment or Uncomment to see the kinematic results difference
%         if i >50
%             est_coords = [est_coords(1) + delta(1),est_coords(2) +
delta(2)];
%         end
    end
end
figure
plot(x_hat_3_a(:,1), x_hat_3_a(:,2), '*')
title('3.a Sequential Least Square results of all data')
xlabel('East (meters)')
ylabel('North (meters)')

% 3.b Kalman Filter of whole data
est_coords = [50, 50];
for i = 1:150
    % Obtain the A matrix
    A = zeros(4,2);
    for j = 1 : 4
        A(j, 1) = (est_coords(1) - targets(j, 1)) / ranges(i, j + 1);
        A(j, 2) = (est_coords(2) - targets(j, 2)) / ranges(i, j + 1);
    end
    % Compute w Matrix
    w= zeros(4,1);
    for j = 1 : 4
        w(j, 1) = sqrt((targets(j, 1) - est_coords(1))^2 + ...
            (targets(j, 2) - est_coords(2))^2) - ranges(i, j + 1);
    end

    if i==1
        % Compute N Matrix and obtain the delta values for first observation
        N = A' * P * A;
        delta = -1 * inv(N) * A' * P * w;
        x_hat_3_b(i,:) = [est_coords(1) + delta(1),est_coords(2) + delta(2)];
    else
        % Sequential LS
        K = inv(N) * A' * inv(P + A*inv(N)*A');
        delta = delta - K*(A*delta + w);
        Q = 50;
        N = inv(inv(N) - K*A*inv(N))+Q;
        x_hat_3_b(i,:) = [est_coords(1) + delta(1),est_coords(2) + delta(2)];

        % Comment or Uncomment to see the kinematic results difference
        if i >50
            est_coords = [est_coords(1) + delta(1),est_coords(2) + delta(2)];
        end
    end
end
figure
plot(x_hat_3_b(:,1), x_hat_3_b(:,2), '*')
hold on
plot(x_hat_3_a(:,1), x_hat_3_a(:,2))
title('3.b Kalman Filter results with Q of 50 vs SLS')
legend('Kalman Filter', 'Sequential')
xlabel('East (meters)')
ylabel('North (meters)')
```