

AE4878 - Mission Geometry and Orbit Design

Part 4

Monte Carlo and grid search optimisation on orbital observations

Ali Nawaz
3ME, LR, Delft University of Technology.

September 2, 2018

1 Introduction

The goal of this assignment is to run Monte Carlo and grid search optimisation on a set of orbital observations. The observations are presented with the aid of Figure 1. X axis values are shifted and do not represent true distance from the focal point. This will be corrected for in the estimations.

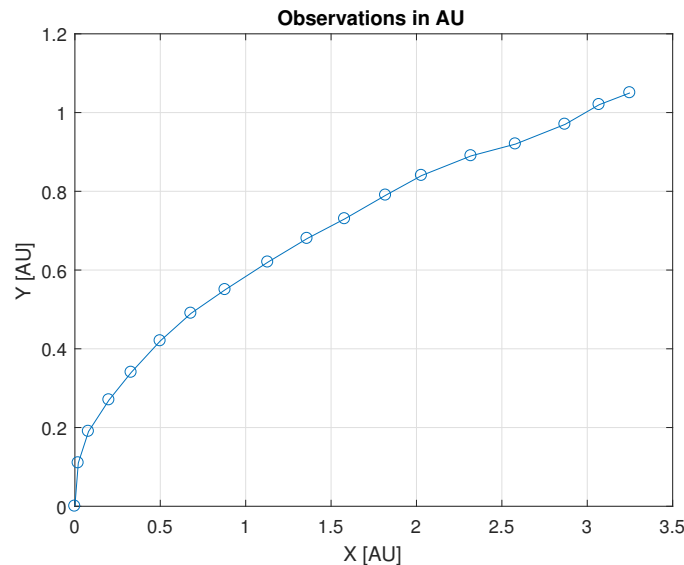


Figure 1: Observation x and y in AU.

Table 1 outlines the discrete X and Y parameters used to plot the observations in Figure 1[1][Lecture- Optimisation Slide- 28].

Table 1: X and Y observations in AU.

X [AU]	0.00	0.02	0.08	0.20	0.33	0.50	0.68	0.88	1.13	1.36	1.58	1.82	2.03	2.32	2.58	2.87	3.07	3.25
Y [AU]	0.00	0.11	0.19	0.27	0.34	0.42	0.49	0.55	0.62	0.68	0.73	0.79	0.84	0.89	0.92	0.97	1.02	1.05

Why Monte Carlo or grid search optimisation? These optimisation techniques rely on sampling or varying parameters to fit a function. The solutions obtained with these approximation are in most cases not optimal but rather starting point localisers. The initial point for optimisation is determined with these search techniques and later more advanced optimisation techniques are used around these sub-optimal solutions to find the optimal solution.

Monte Carlo offers a choice for sampling distribution. For this assignment a uniform sampling distribution is chosen. While grid search as the name suggests places a uniform/non-uniform grid on a topography to estimate parameters. For this assignment uniform spacing is used for the grid search criteria.

First the observation is fit through the assumption that the observations resemble to that of a parabolic orbit. Both Monte Carlo and grid search methods are implemented for parabolic orbit in Section 2. Section 3 focuses on application of Monte Carlo and grid search methods from the perspective of hyperbolic orbits. Section 4 presents the Matlab scripts used to facilitate the estimations.

2 Parabolic orbit

The objective of this Section is to assume that the orbital observations are parabolic. Monte Carlo and grid search sampling based optimization can be applied easily on the observations if an analytical expression of x and y positions of the orbital body wrt the focal point is known. Wertz [2][pg.51, 845] provides an analytic expression relating x and y parameters of a parabola. However, Wertz ignores the need to describe the process of deriving those parameters. As a result Wertz ends up deriving the parabolic expression for a regular parabolic function. However, for the purpose of orbital calculation sideways parabolic function is rather attractive.

Figure 2 orbit outlines a sideways parabolic orbit¹. Here q is the perifocal distance, which is the distance between the vertex of the parabola and the focal point. q is also the distance between directrix and the vertex point. The observations start from [0,0]. However, Figure 2 shows that from the origin along the directrix line, the vertex has an offset of q.

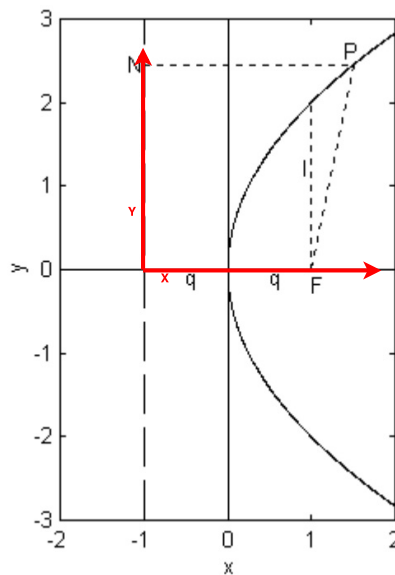


Figure 2: Observation x and y in AU.

Parametric expression for a parabolic orbit can be expressed with the aid of Equation 1².

$$4q(x-h) = (y-k)^2$$

Where h is the x directional offset of the vertex from the origin.

While k is the y directional offset of the vertex from the origin.

(1)

Origin lies on the intersection of directrix and a perpendicular line passing through the focal point.

For the observations shown in Figure1, the parametric equation can be simplified to Equation 2. Where q is the distance of the vertex from the focal point. q is also known as the peri-apsis/peri-focal distance.

¹Parabolic and Hyperbolic orbit parameters: <http://astrowww.phys.uvic.ca/~tatum/celmechs/celm2.pdf>

²Parabolic orbit parameters: <http://www.purplemath.com/modules/parabola.html>

$$x = \frac{y^2}{4q} + q \quad (2)$$

Equation 2 is the final analytical expression where the parameter to be estimated is the perifocal distance q .

2.1 Monte Carlo

For Monte Carlo sampling the objective is to use the Y observations outlined in Table 1. q is generated in the range 0 to 0.5 AU. This is reasonably in line with the verification figure[1] [Lecture - Optimisation, Slide - 28]. q is generated with uniform random distribution. Different samples of 50, 100 and 500 are generated. For each no. of samples, 3 runs are conducted. Based on the values of Y observation and generated q , values of X are obtained. Optimum value of parameter q is chosen such that values of resulting X has the minimum standard deviation from the observations. It is important to note that observations are not perfect either, this is observed by the non-smooth curve in Figure1. Value of q for which minimum standard deviation is observed is chosen as the optimum. Script used for Monte Carlo sampling is presented in Section 4.1.

Figures 3, 4 and 5 represents the X and Y observations for different number of samples.

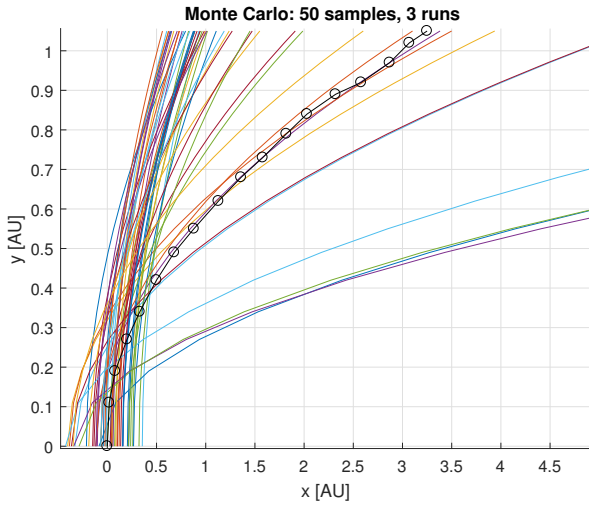


Figure 3: Parabolic parameter estimations with Monte Carlo for 50 samples.

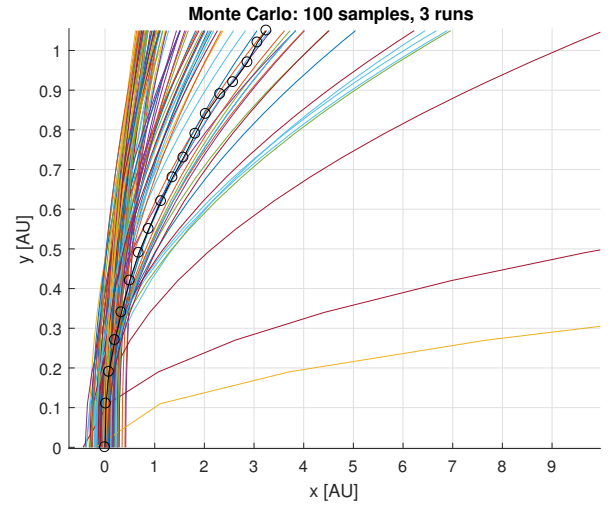


Figure 4: Parabolic parameter estimations with Monte Carlo for 100 samples.

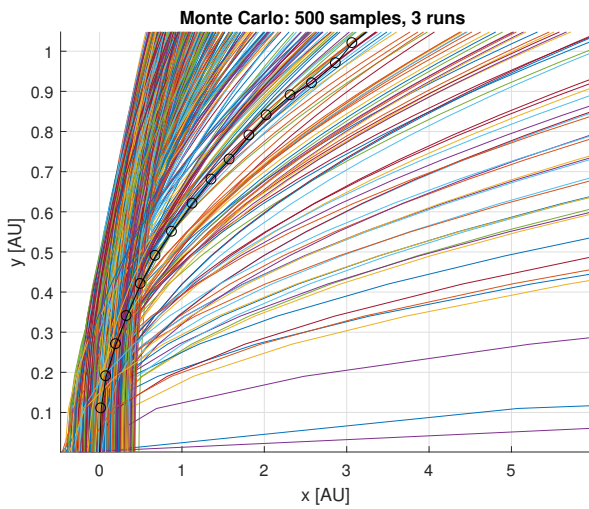


Figure 5: Parabolic parameter estimations with Monte Carlo for 500 samples.

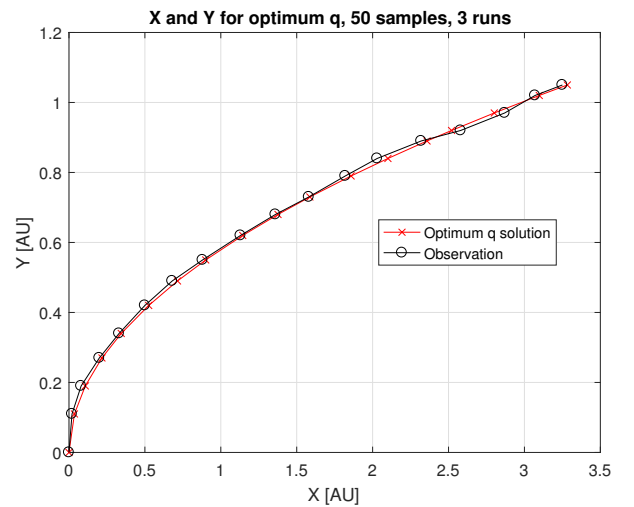


Figure 6: Optimal parabolic parameter estimation with Monte Carlo for 50 samples. $q = 0.0832$ AU and std of 0.0327 AU.

While Figures 6, 7 and 8 represent the observations for optimum values of q .

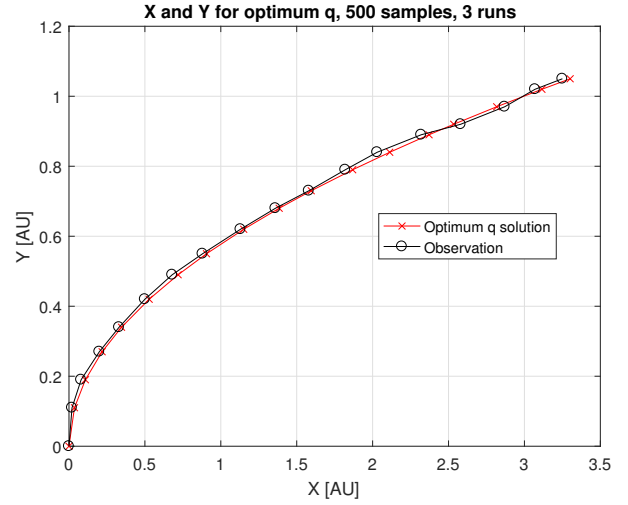
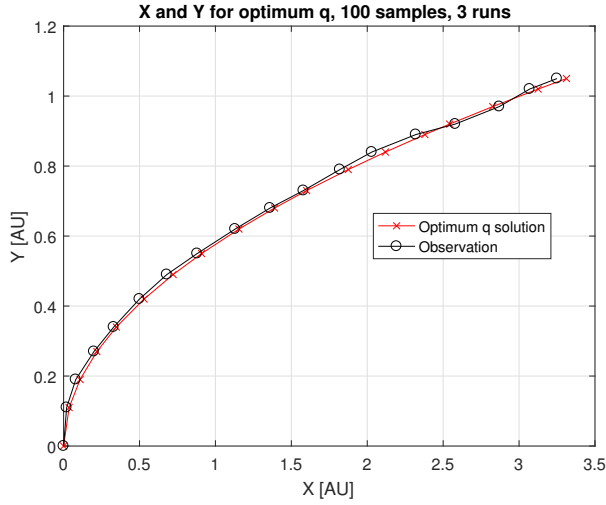


Figure 7: Optimal parabolic parameter estimation with Monte Carlo for 50 samples. $q = 0.0828$ AU and std of Monte Carlo for 50 samples. $q = 0.0836$ AU and std of 0.0339 AU. Figure 8: Optimal parabolic parameter estimation with Monte Carlo for 500 samples. $q = 0.0836$ AU and std of 0.0324 AU.

To conclude this section, table 2 outlines the minimum standard deviation for different number of samples along with the corresponding value of q obtained. From table 2 it can be concluded that, increasing the number of samples increases the accuracy of solution. This can be verified by the decreasing standard deviations for increasing number of samples. However, since the samples are generated in random manner, it could easily result in a scenario that lower no. of samples provide better fit than higher no. of samples. This is seen in this specific scenario when the no. of samples were increased from 50 to 100.

Table 2: Variation of q and standard deviation for Monte Carlo sampling.

No. of Samples	Perifocal distance q [AU]	Standard deviation [AU]
50	0.0832	0.0327
100	0.0828	0.0339
500	0.0836	0.0324

2.2 Grid Search

The idea behind grid sample is analogous to that of Monte Carlo. Instead of using a random number generator, arithmetic progression of stencils is used. Equation 2 is used as the parametric expression. Similar to Monte Carlo estimations, values of Y are feed into Equation 2, while incrementally varying the grid values for q in the same range [0AU, 0.5AU]. Running once instead of three times is sufficient, since both the function and the grid meshes are constant. Optimum value of parameter q is chosen such that resulting values of X have the minimum standard deviation from the observations. Value of q for which minimum standard deviation is observed is chosen as the optimum. Script used for grid search sampling is outlined in Section 4.1.

Figures 9, 10 and 11 represents the X and Y observations for different number of samples.

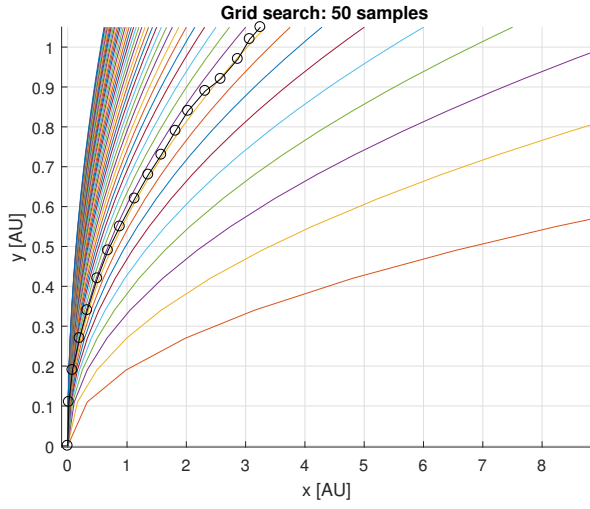


Figure 9: Parabolic parameter estimations with grid search for 50 samples.

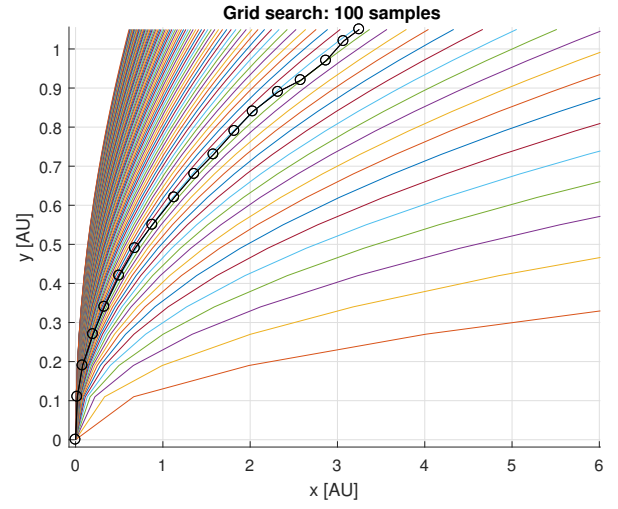


Figure 10: Parabolic parameter estimations with grid search for 100 samples.

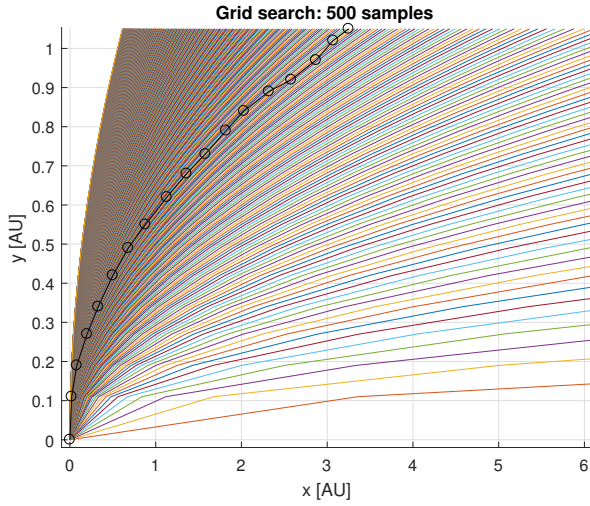


Figure 11: Parabolic parameter estimations with grid search for 500 samples.

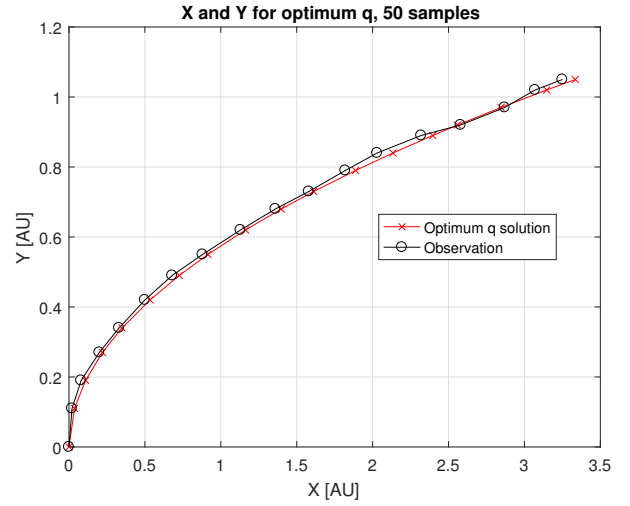


Figure 12: Optimal parabolic parameter estimation with grid search for 50 samples. $q = 0.0827$ AU and std of 0.0346 AU.

While Figures 12, 13 and 14 represent the observations for optimum values of q . Table 3 outlines the minimum standard deviation for different number of samples along with the corresponding value of q obtained.

Table 3: Values of optimum q and the corresponding standard deviations for different grid samples.

No. of Samples	Perifocal distance q [AU]	Standard deviation [AU]
50	0.0827	0.0346
100	0.0818	0.0402
500	0.0839	0.0326

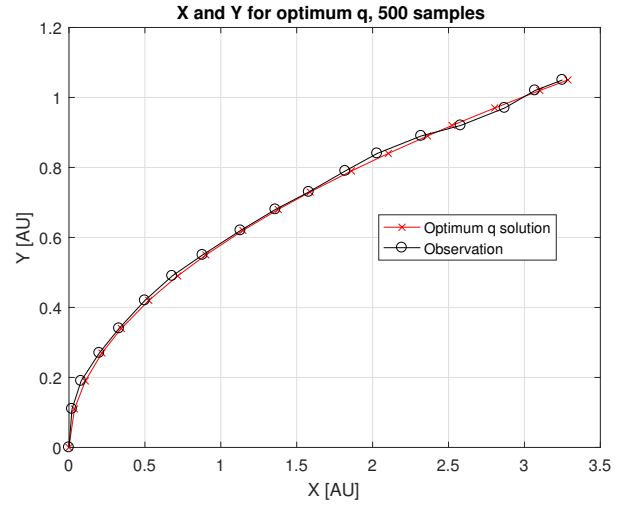
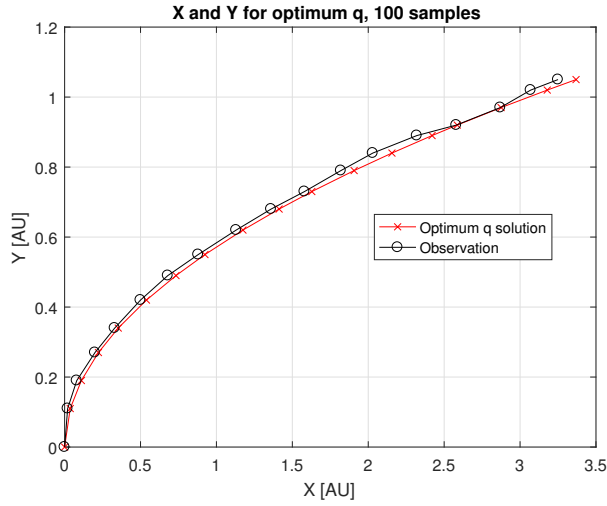


Figure 13: Optimal parabolic parameter estimation with grid search for 100 samples. $q = 0.0818$ AU and std of 0.0402 AU. Figure 14: Optimal parabolic parameter estimation with grid search for 500 samples. $q = 0.0839$ AU and std of 0.0326 AU.

From table 3 it can be concluded that, even if one might expect that increasing the number of samples increases the accuracy of solution this is not always true. This is observed when the no. of samples is increased from 50 to 100. But in general when the no of grid samples cross a certain threshold, the standard deviations either go down or remain constant for a convex function.

2.3 Comparison: Quality vs Time

For Monte Carlo process the computational effort is at least 3 times higher than the computational effort of grid search method. This is justified by the fact that there are three runs. This excludes the fact that generating random number by Monte Carlo adds more computational effort on top of that, compared to a predefined grid. The term computational effort indicate calculation to get estimations for x . Results for one sample is defined as one calculation unit. Based on this a plot can be generated with no. of calculations (function of time) on the x axis and the corresponding standard deviation (function of accuracy) on the y axis. This is shown with the aid of Figure15[Sample points: 50, 100 and 500 order left to right].

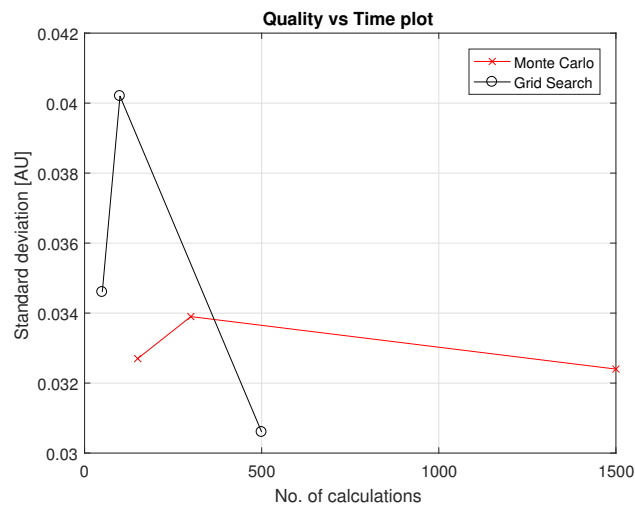


Figure 15: Standard deviations against no. of calculations for Monte Carlo and grid search sampling methods.

In conclusion, for lower number of samples the quality to time ratio for Monte Carlo could be higher than that of grid search. However for increasing samples, grid search demonstrates higher quality to time ratio than Monte Carlo. This is true at least for the problem under consideration.

3 Hyperbolic orbit

The objective of this Section is to assume that the observations are hyperbolic. This is rather unlikely, observing the results for the parabolic functions in the previous section. Since the observations were well submerged within parabolic estimation. But no conclusion can be made yet. The aim of this chapter is to make use of a Hyperbolic parametric equation and use Monte Carlo and grid searching to observe how well the estimations fit the observations. An analytic expression for x and y positions is expressed with the aid of Equation 3³. Figure 16 presents an overview for the co-ordinate system for a hyperbolic orbit⁴.

$$\frac{x^2}{a^2} - \frac{y^2}{b^2} = 1$$

Where a is the semi-transverse axis. $a < 0$ for Hyperbolic orbit.

a can be considered as the distance between directrix and the vertex of hyperbola. (3)

c is the distance from the origin to the focal point.

While $b^2 = c^2 - a^2$. Where b = semi-conjugate axis.

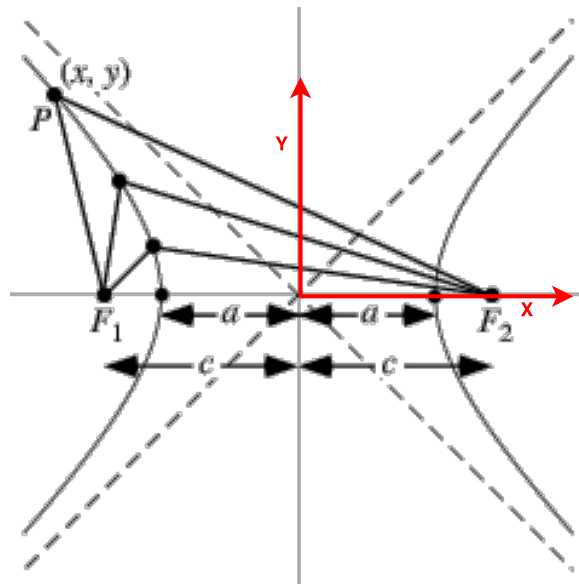


Figure 16: Co-ordinate frame for a hyperbolic orbit.

Since x coordinate of observations bear an offset of amount a, this is included into the parametric equation in Equation 3. Equation 4 expresses the analytic parametric equation for hyperbolic orbit. Where the parameters to be estimated are a and b.

$$\frac{(x - a)^2}{a^2} - \frac{y^2}{b^2} = 1 \quad (4)$$

3.1 Monte Carlo

For Monte Carlo sampling the objective is to use the Y observations outlined in Table 1. a is generated in the range -0.5 to 0.5 AU. This is reasonably in line with the verification figure[1] [Lecture - Optimisation, Slide - 28]. a is generated with uniform random distribution. b is related to the eccentricity and a as shown in Equation 5.

³Hyperbolic parametric equation: <http://www.nabla.hr/PC-ParametricEqu4.htm>

⁴Hyperbolic co-ordinate: <http://mathworld.wolfram.com/Hyperbola.html>

$$b^2 = a^2 \cdot (e^2 - 1)$$

for Hyperbolic orbit $e > 1$. Since $a^2 = (\pm 0.5)^2$

If eccentricity is chosen between 1 and 1.4. The range for b, lies in the range:

$$b^2 = [0, 0.24]$$

(5)

b is also generated by uniform distribution in the given range. Different samples of 50, 100 and 500 are generated. For each no. of samples, 3 runs are conducted. Based on the values of Y observation and generated a and b, values of X are obtained. Optimum value of parameters (a,b) is chosen such that values of resulting X has the minimum standard deviation from the observations. It is important to note that observations are not perfect either, this is observed by the non-smooth curve in Figure1. Value of (a,b) for which minimum standard deviation is observed is chosen as the optimum combination. Script used for Monte Carlo sampling is presented in Section 4.2.

Figures 17, 18 and 19 represents the X and Y observations for different number of samples. While Figures 20, 21 and 22 represent the observations for optimum values of a and e.

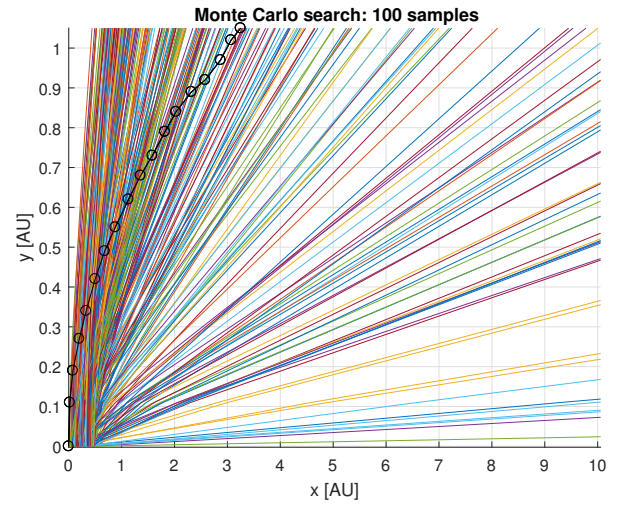
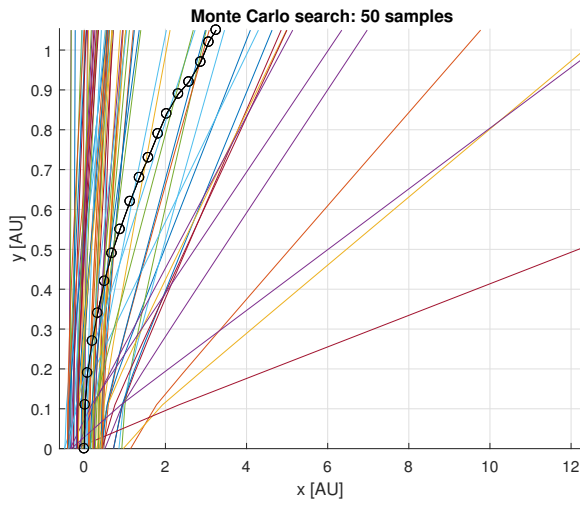


Figure 17: Hyperbolic parameter estimations with Monte Carlo for 50 samples.

Figure 18: Hyperbolic parameter estimations with Monte Carlo for 100 samples.

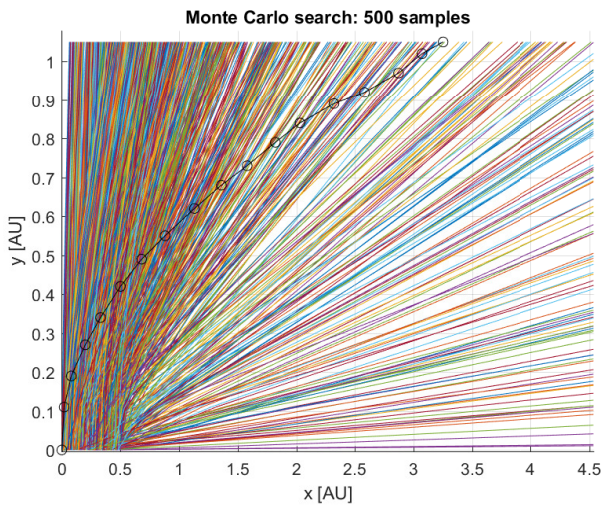


Figure 19: Hyperbolic parameter estimations with Monte Carlo for 500 samples.

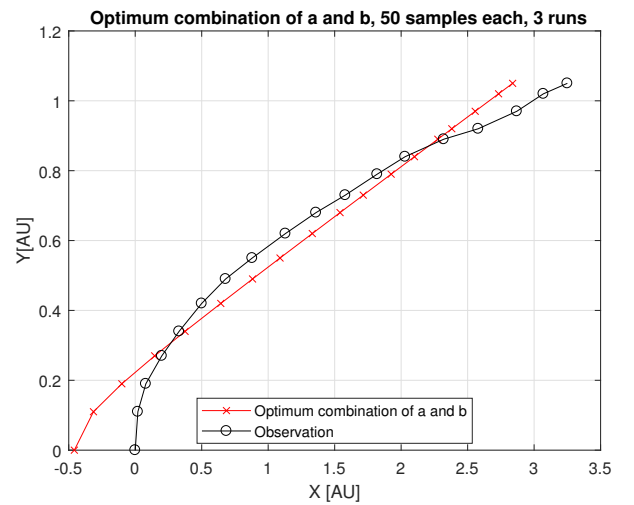


Figure 20: Optimal hyperbolic parameter estimation with Monte Carlo for 50 samples. $a = -0.4578$ AU, $b = 0.1290$ AU and std of 0.2344 AU.

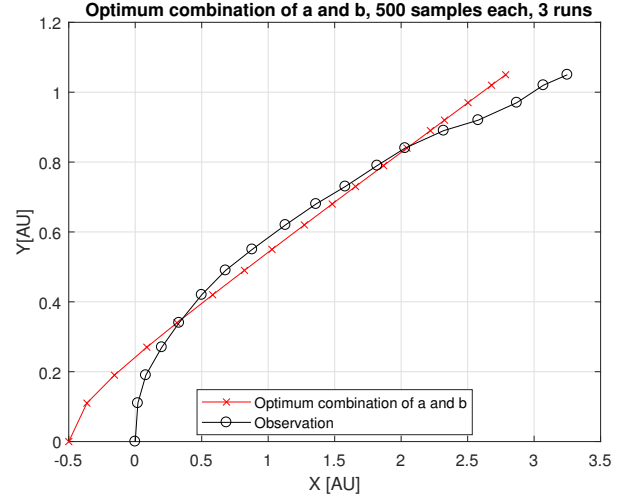
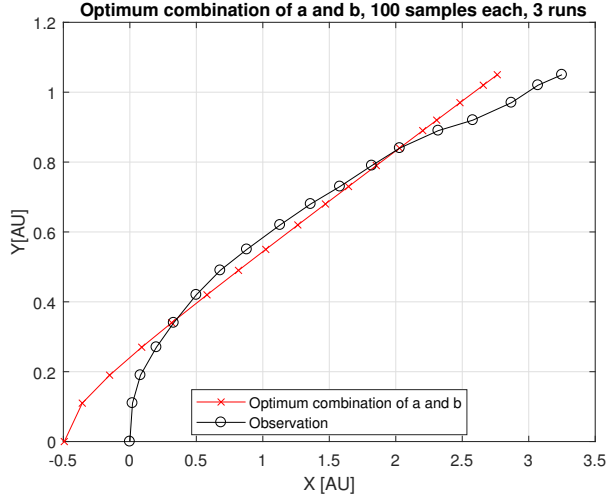


Figure 21: Optimal hyperbolic parameter estimation with Monte Carlo for 100 samples. $a = -0.4931$ AU, $b = 0.1393$ AU and std of 0.2311 AU .
Figure 22: Optimal hyperbolic parameter estimation with Monte Carlo for 500 samples. $a = -0.4996$ AU, $b = 0.1398$ AU and std of 0.2303 AU .

To conclude this section, table 4 outlines the minimum standard deviation for different number of samples along with the corresponding value of optimum a and b obtained. From table 2 it can be concluded that, increasing the number of samples increases the accuracy of solution. This can be verified by the decreasing standard deviations for increasing number of samples. However, the standard deviations are quite off compared to the values obtained for a parabolic orbit scenario. The reason is obvious, an orbit cannot be parabolic and hyperbolic at the same time. Based on the observations the orbit is rather parabolic.

Table 4: Values of optimum a and b parameters and the corresponding standard deviations for Monte Carlo simulations.

No. of Samples	Semi major axis, a [AU]	Semi conjugate axis, b [AU]	Standard deviation [AU]
50	-0.4578	0.1290	0.2344
100	-0.4931	0.1393	0.2311
500	-0.4996	0.1398	0.2303

3.2 Grid Search

The idea behind grid sample is analogous to that of Monte Carlo. Instead of using a random number generator, arithmetic progression of stencils is used. Equation 4 is used as the parametric expression. Similar to Monte Carlo estimations, values of Y are feed into Equation 4, while incrementally varying the grid values for (a,b) in the same range as presented for the Monte Carlo case. i.e. $a = [-0.5 \text{ AU}, 0.5 \text{ AU}]$ and $b^2 = [0 \text{ AU}, 0.24 \text{ AU}]$. Running once instead of three times is sufficient, since both the function and the grid meshes are constant. Optimum value of the combinations of parameters (a,b) is chosen such that resulting values of X have the minimum standard deviation from the observations. Combinations of (a,b) for which minimum standard deviation is observed is chosen as the optimum. Script used for grid search sampling is outlined in Section 4.2.

Figures 23, 24 and 25 represents the X and Y observations for different number of samples.

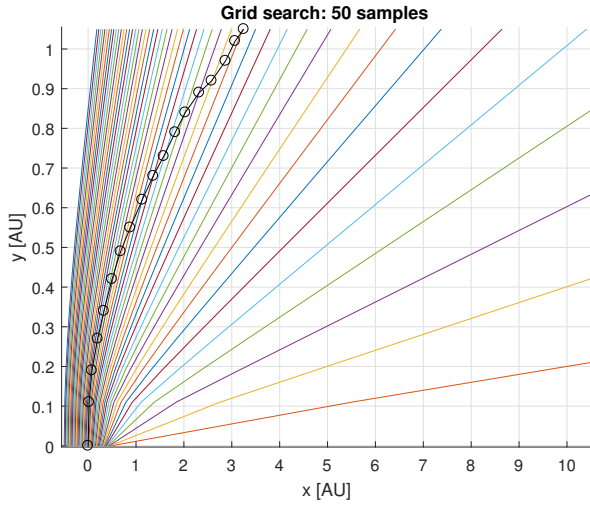


Figure 23: Hyperbolic parameter estimations with grid search for 50 samples.

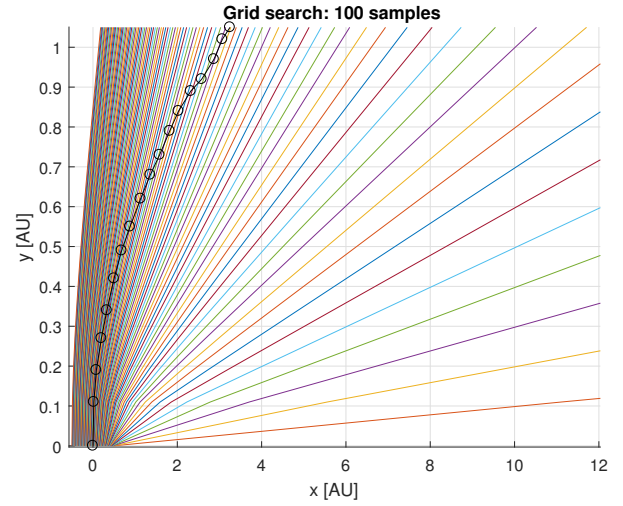


Figure 24: Hyperbolic parameter estimations with grid search for 100 samples.

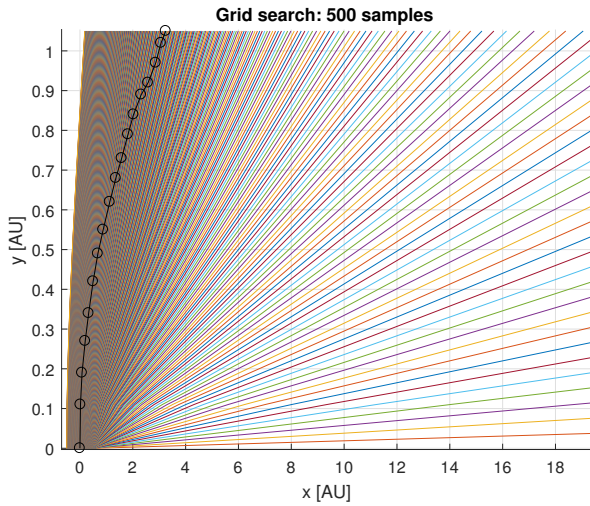


Figure 25: Hyperbolic parameter estimations with grid search for 500 samples.

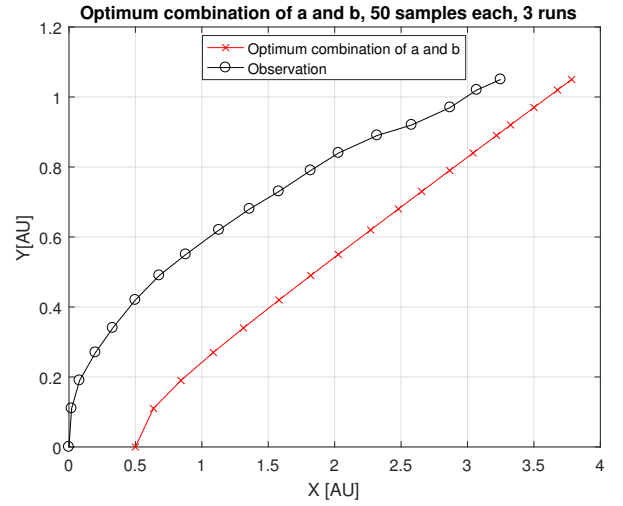


Figure 26: Optimal hyperbolic parameter estimation with grid search for 50 samples. $a = 0.500$ AU, $b = 0.140$ AU and std of 0.2302 AU.

While Figures 26, 27 and 28 represent the observations for optimum values of q . Table 5 outlines the minimum standard deviation for different number of samples along with the corresponding value of (a,b) obtained.

Table 5: Values of optimum a and b parameters and the corresponding standard deviations for grid search.

No. of Samples	Semi major axis, a [AU]	Semi conjugate axis, b [AU]	Standard deviation [AU]
50	0.5000	0.1400	0.2302
100	-0.5000	0.1386	0.2306
500	-0.5000	0.1404	0.2302

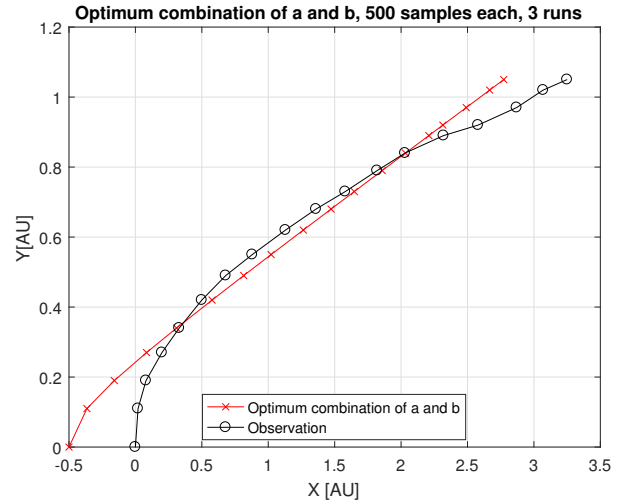
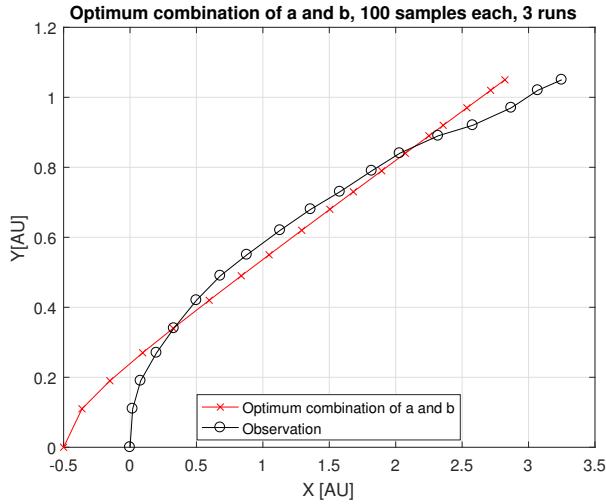


Figure 27: Optimal hyperbolic parameter estimation with grid search for 100 samples. $a = -0.5$ AU, $b = 0.1386$ and std of 0.2306 AU. Figure 28: Optimal hyperbolic parameter estimation with grid search for 500 samples. $a = -0.500$ AU, $b = 0.1404$ and std of 0.2302 AU.

From table 5 it can be concluded that, even if one might expect that increasing the number of samples increases the accuracy of solution this is not always true. This is observed when the no. of samples is increased from 50 to 100. But in general when the no of grid samples cross a certain threshold, the standard deviations either go down or remain constant for a convex function. However, a very important observation is that for no. of samples = 50, the best fit solution for a is positive. However, for hyperbolic orbit it is known that " a " should be negative.

3.3 Comparison: Quality vs Time

For Monte Carlo process the computational effort is at least 3 times higher than the computational effort of grid search method. This is justified by the fact that there are three runs. This excludes the fact that generating random number by Monte Carlo adds more computational effort on top of that, compared to a predefined grid. The term computational effort indicate calculation to get estimations for x . Results for one sample is defined as one calculation unit. Based on this a plot can be generated with no. of calculations (function of time) on the x axis and the corresponding standard deviation (function of accuracy) on the y axis. This is shown with the aid of Figure 29 [Sample points: 50, 100 and 500 order left to right].

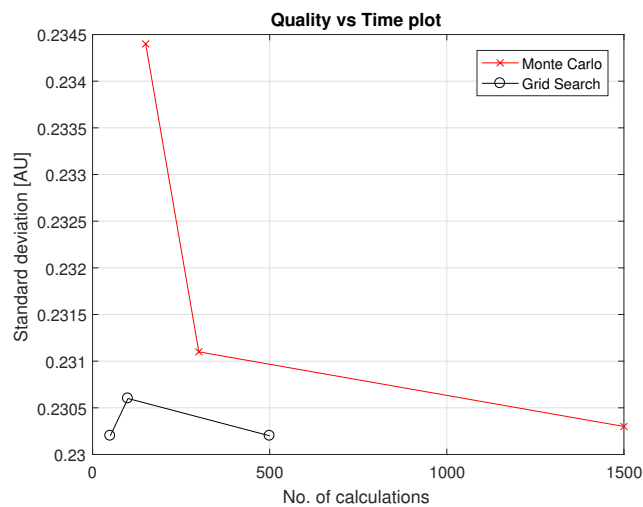


Figure 29: Standard deviations against no. of calculations for Monte Carlo and grid search sampling methods.

In conclusion, grid search demonstrates higher quality to time ratio than Monte Carlo. However, the standard deviations in general are significantly higher than that observed in Figure 15. There are two reason for it;

1.The orbit is parabolic, expecting a parabolic orbit to fit through a hyperbolic model will surely result in high standard deviations. 2.Grid based or Monte Carlo based sampling techniques lose resolution when the dimension of parameter is increased but the no. of samples are kept the same.

It can be concluded that observed X and Y values form a parabolic trajectory.

4 Matlab Script

4.1 Parabolic orbit

Following script is used for Monte Carlo sampling of parabolic orbit.

```

1  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2  %%% AE4878 Mission Geometry and Orbit Design %%%%%%%%%
3  %%% Assignment 4 - Week 5 - OPTIMIZATION - Version 1 %%%
4  %%% Author Info: Ali Nawaz; Student ID - 4276477 %%%
5  %%% Assigned tasks : OPTIM.5a, QUESTION 2 %%%%%%%%%
6  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
7
8  %% MONTE CARLO FOR PARABOLIC PARAMETER ESTIMATION
9  close all; clear all;
10 x = [ 0.00 0.02 0.08 0.20 0.33 0.50 0.68 0.88 1.13 1.36 1.58 1.82 2.03 2.32 2.58 2.87 3.07 ...
      3.25]; % Distance in AU from slide 28
11 y = [ 0.00 0.11 0.19 0.27 0.34 0.42 0.49 0.55 0.62 0.68 0.73 0.79 0.84 0.89 0.92 0.97 1.02 ...
      1.05]; % Distances in AU from slide 28
12
13 ub = 0.50; % Upper bound on perifocal distance q [AU]
14 lb = 0; % Lower bound on perifocal distance q [AU]
15
16 % Store the estimations of x and corresponding standard deviations
17
18 % For 50 samples
19 x_est1 = [];
20 std_est1 = [];
21
22 % For 100 samples
23 x_est2 = [];
24 std_est2 = [];
25
26 % For 500 samples
27 x_est3 = [];
28 std_est3 = [];
29
30 q1_lst = [];
31 q2_lst = [];
32 q3_lst = [];
33
34 for run = 1:3 % Repeating each run three times
35
36     % Random perifocal distance generation
37     q1 = lb + (ub).*rand(1,50); % 50 samples
38     q2 = lb + (ub).*rand(1,100); % 100 samples
39     q3 = lb + (ub).*rand(1,500); % 500 samples
40
41     for t = 1:length(q1)
42         x_buf1 = q1(t) + (y.^2)./(4*q1(t)); % store the value of x for q
43         std_est1 = [std_est1, std(x - (x_buf1- q1(t)))]; % estimate standard deviation
44         x_est1 = [x_est1,x_buf1']; % store the estimated x
45     end
46
47     for t = 1:length(q2)
48         x_buf2 = q2(t) + (y.^2)./(4*q2(t)); % store the value of x for q
49         std_est2 = [std_est2, std(x - (x_buf2- q2(t)))]; % estimate standard deviation
50         x_est2 = [x_est2,x_buf2']; % store the estimated x
51     end
52
53     for t = 1:length(q3)
54         x_buf3 = q3(t) + (y.^2)./(4*q3(t)); % store the value of x for q
55         std_est3 = [std_est3, std(x - (x_buf3- q3(t)))]; % estimate standard deviation
56         x_est3 = [x_est3,x_buf3']; % store the estimated x
57     end

```

```

58
59     q1_lst = [q1_lst,q1];
60     q2_lst = [q2_lst,q2];
61     q3_lst = [q3_lst,q3];
62 end
63
64 % Plot figure for 50 samples, 3 runs case
65 figure(1)
66 hold on
67 for k = 1:length(q1)
68     plot(x_est1(:,k)-q1(k),y');
69 end
70 plot(x,y,'k-o');
71 grid on
72 title('Monte Carlo: 50 samples, 3 runs');
73 xlabel('x [AU]');
74 ylabel('y [AU]');
75 hold off
76
77 % Plot figure for 100 samples, 3 runs case
78 figure(2)
79 hold on
80 for k = 1:length(q2)
81     plot(x_est2(:,k)-q2(k),y');
82 end
83 plot(x,y,'k-o');
84 grid on
85 title('Monte Carlo: 100 samples, 3 runs');
86 xlabel('x [AU]');
87 ylabel('y [AU]');
88 hold off
89
90 % Plot figure for 500 samples, 3 runs case
91 figure(3)
92 hold on
93 for k = 1:length(q3)
94     plot(x_est3(:,k)-q3(k),y');
95 end
96 plot(x,y,'k-o');
97 grid on
98 title('Monte Carlo: 500 samples, 3 runs');
99 xlabel('x [AU]');
100 ylabel('y [AU]');
101 hold off
102
103 % Find the index for minimum standard deviation
104 index1 = find(std_est1 == min(std_est1));
105 index2 = find(std_est2 == min(std_est2));
106 index3 = find(std_est3 == min(std_est3));
107
108 % Find the optimum value of q for which min. std happens.
109 opti_q1 = q1_lst(index1);
110 opti_q2 = q2_lst(index2);
111 opti_q3 = q3_lst(index3);
112
113 %% Plot X and Y for optimum values of q
114 figure(11)
115 plot(x_est1(:,index1) - q1_lst(index1),y', 'r-x',x,y,'k-o');
116 grid on
117 legend('Optimum q solution','Observation','Location','best');
118 xlabel('X [AU]');
119 ylabel('Y [AU]');
120 title('X and Y for optimum q, 50 samples, 3 runs');
121
122 figure(22)
123 plot(x_est2(:,index2)- q2_lst(index2),y', 'r-x',x,y,'k-o');
124 grid on
125 legend('Optimum q solution','Observation','Location','best');
126 xlabel('X [AU]');
127 ylabel('Y [AU]');
128 title('X and Y for optimum q, 100 samples, 3 runs');
129
130 figure(33)
131 plot(x_est3(:,index3)- q3_lst(index3),y', 'r-x',x,y,'k-o');
132 grid on
133 legend('Optimum q solution','Observation','Location','best');

```

```

134 xlabel('X [AU]');
135 ylabel('Y [AU]');
136 title('X and Y for optimum q, 500 samples, 3 runs');

```

Following is the script used for grid search sampling of parabolic orbit.

```

1  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2  %%% AE4878 Mission Geometry and Orbit Design %%%%%%%%%
3  %%% Assignment 4 - Week 5 - OPTIMIZATION - Version 1 %%%
4  %%% Author Info: Ali Nawaz; Student ID - 4276477 %%%
5  %%% Assigned tasks : OPTIM.5a, Question 3 %%%%%%%%%
6  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
7
8  %% GRID SEARCH FOR PARABOLIC PARAMETER ESTIMATION
9  close all; clear all;
10 x = [ 0.00 0.02 0.08 0.20 0.33 0.50 0.68 0.88 1.13 1.36 1.58 1.82 2.03 2.32 2.58 2.87 3.07 ...
      3.25]; % Distance in AU from slide 28
11 y = [ 0.00 0.11 0.19 0.27 0.34 0.42 0.49 0.55 0.62 0.68 0.73 0.79 0.84 0.89 0.92 0.97 1.02 ...
      1.05]; % Distances in AU from slide 28
12
13 ub = 0.45; % Upper bound on perifocal distance q [AU]
14 lb = 0; % Lower bound on perifocal distance q [AU]
15
16 % Store the estimations of x and corresponding standard deviations
17
18 % For 50 samples
19 x_est1 = [];
20 std_est1 = [];
21
22 % For 100 samples
23 x_est2 = [];
24 std_est2 = [];
25
26 % For 500 samples
27 x_est3 = [];
28 std_est3 = [];
29
30 q1_lst = [];
31 q2_lst = [];
32 q3_lst = [];
33
34 % Grided perifocal distance generation
35 q1 = 0:(ub/(50-1)):ub; % 50 samples
36 q2 = 0:(ub/(100-1)):ub; % 100 samples
37 q3 = 0:(ub/(500-1)):ub; % 500 samples
38
39 for t = 1:length(q1)
40     x_buf1 = q1(t) + (y.^2)./(4*q1(t)); % store the value of x for q
41     std_est1 = [std_est1, std(x - (x_buf1- q1(t)))]; % estimate standard deviation
42     x_est1 = [x_est1,x_buf1']; % store the estimated x
43 end
44
45 for t = 1:length(q2)
46     x_buf2 = q2(t) + (y.^2)./(4*q2(t)); % store the value of x for q
47     std_est2 = [std_est2, std(x - (x_buf2- q2(t)))]; % estimate standard deviation
48     x_est2 = [x_est2,x_buf2']; % store the estimated x
49 end
50
51 for t = 1:length(q3)
52     x_buf3 = q3(t) + (y.^2)./(4*q3(t)); % store the value of x for q
53     std_est3 = [std_est3, std(x - (x_buf3- q3(t)))]; % estimate standard deviation
54     x_est3 = [x_est3,x_buf3']; % store the estimated x
55 end
56
57 q1_lst = [q1_lst,q1];
58 q2_lst = [q2_lst,q2];
59 q3_lst = [q3_lst,q3];
60
61
62 % Plot figure for 50 samples, 3 runs case
63 figure(1)
64 hold on
65 for k = 1:length(q1)
66     plot(x_est1(:,k)-q1(k),y');
67 end

```



```

68 plot(x,y,'k-o');
69 grid on
70 title('Grid search: 50 samples');
71 xlabel('x [AU]');
72 ylabel('y [AU]');
73 hold off
74
75 % Plot figure for 100 samples, 3 runs case
76 figure(2)
77 hold on
78 for k = 1:length(q2)
79     plot(x_est2(:,k)-q2(k),y');
80 end
81 plot(x,y,'k-o');
82 plot(x,y,'k-o');
83 grid on
84 title('Grid search: 100 samples');
85 xlabel('x [AU]');
86 ylabel('y [AU]');
87 hold off
88
89 % Plot figure for 500 samples, 3 runs case
90 figure(3)
91 hold on
92 for k = 1:length(q3)
93     plot(x_est3(:,k)-q3(k),y');
94 end
95 plot(x,y,'k-o');
96 grid on
97 title('Grid search: 500 samples');
98 xlabel('x [AU]');
99 ylabel('y [AU]');
100 hold off
101
102 % Find the index for minimum standard deviation
103 index1 = find(std_est1 == min(std_est1));
104 index2 = find(std_est2 == min(std_est2));
105 index3 = find(std_est3 == min(std_est3));
106
107 % Find the optimum value of q for which min. std happens.
108 opti.q1 = q1_lst(index1);
109 opti.q2 = q2_lst(index2);
110 opti.q3 = q3_lst(index3);
111
112 %% Plot X and Y for optimum values of q
113 figure(11)
114 plot(x_est1(:,index1) - q1_lst(index1),y', 'r-x',x,y,'k-o');
115 grid on
116 legend('Optimum q solution','Observation','Location','best');
117 xlabel('X [AU]');
118 ylabel('Y [AU]');
119 title('X and Y for optimum q, 50 samples');
120
121 figure(22)
122 plot(x_est2(:,index2)- q2_lst(index2),y', 'r-x',x,y,'k-o');
123 grid on
124 legend('Optimum q solution','Observation','Location','best');
125 xlabel('X [AU]');
126 ylabel('Y [AU]');
127 title('X and Y for optimum q, 100 samples');
128
129 figure(33)
130 plot(x_est3(:,index3)- q3_lst(index3),y', 'r-x',x,y,'k-o');
131 grid on
132 legend('Optimum q solution','Observation','Location','best');
133 xlabel('X [AU]');
134 ylabel('Y [AU]');
135 title('X and Y for optimum q, 500 samples');

```

Following is the script for quality vs time comparison of parabolic orbit:

```

1 %% COMPARING MONTE CARLO AND GRID SEARCH COMPUTATIONAL EFFORT VS QUALITY
2 %no. of samples
3 samples = [ 50 100 500 ];
4

```

```

5 %no. of calculations, m = Monte Carlo, g = grid search
6 calc_m = 3.*samples;
7 calc_g = samples;
8
9 %corresponding standard deviations
10 std_m = [0.0327 0.0339 0.0324];
11 std_g = [ 0.0346 0.0402 0.0306];
12
13 % Plot the results
14 figure(41)
15 plot(calc_m,std_m,'r-x',calc_g,std_g,'k-o');
16 grid on
17 legend('Monte Carlo','Grid Search');
18 ylabel('Standard deviation [AU]');
19 xlabel('No. of calculations');
20 title('Quality vs Time plot');

```

4.2 Hyperbolic orbit

Following is the script used for Monte Carlo sampling of Hyperbolic orbit.

```

1 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2 %%% AE4878 Mission Geometry and Orbit Design %%%%%%%%%
3 %%% Assignment 4 - Week 5 - OPTIMIZATION - Version 1 %%%
4 %%% Author Info: Ali Nawaz; Student ID - 4276477 %%%
5 %%% Assigned tasks : OPTIM_5a, QUESTION 5 %%%%%%%%%
6 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
7
8 %% MONTE CARLO FOR HYPERBOLIC PARAMETER ESTIMATION, PARAMETERS : semi transverse axis a and ...
9   semi-conjugate axis b
10 % close all; clear all;
11 x = [ 0.00 0.02 0.08 0.20 0.33 0.50 0.68 0.88 1.13 1.36 1.58 1.82 2.03 2.32 2.58 2.87 3.07 ...
12       3.25]; % Distance in AU from slide 28
13 y = [ 0.00 0.11 0.19 0.27 0.34 0.42 0.49 0.55 0.62 0.68 0.73 0.79 0.84 0.89 0.92 0.97 1.02 ...
14       1.05]; % Distances in AU from slide 28
15
16 uba = 0.5; % Upper bound on a, semi transverse axis
17 lba = -0.5; % Lower bound on a, semi conjugate a
18
19 ubb = sqrt(0.240); % Upper bound on b
20 lbb = 0; % Lower bound on b
21
22 % Store the estimations of x and corresponding standard deviations
23
24 % For 50 samples
25 x_est1 = [];
26 std_est1 = [];
27
28 % For 100 samples
29 x_est2 = [];
30 std_est2 = [];
31
32 % For 500 samples
33 x_est3 = [];
34 std_est3 = [];
35
36 % List of semi major axis a for sample size 50, 100 and 500
37 a1_lst = [];
38 a2_lst = [];
39 a3_lst = [];
40
41 % List of semi conjugagte axis b for sample size 50, 100 and 500
42 b1_lst = [];
43 b2_lst = [];
44 b3_lst = [];
45
46 % Storing a and b parameters for each calculation
47 para_lst1 = []; % 50 samples
48 para_lst2 = []; % 100 samples
49 para_lst3 = []; % 500 samples
50
51 for run = 1:3 % Repeating each run three times

```

```

49 % Random perifocal distance generation
50 a1 = lba + (uba + uba).*rand(1,50); % 50 samples
51 a2 = lba + (uba).*rand(1,100); % 100 samples
52 a3 = lba + (uba).*rand(1,500); % 500 samples
53
54 % Random eccentricity generation
55 b1 = lbb + (ubb).*rand(1,50); % 50 samples
56 b2 = lbb + (ubb).*rand(1,100); % 100 samples
57 b3 = lbb + (ubb).*rand(1,500); % 500 samples
58
59
60 for t = 1:length(a1)
61     for j = 1:length(b1)
62         x_buf1 = sqrt( (1 + (y.^2)./(b1(j)^2) ).*a1(t)^2) + a1(t); % Estimate the value ...
63             of x for given q,e and y
64         std_est1 = [std_est1, std(x - (x_buf1- a1(t)))]; % estimate standard deviation
65         x_est1 = [x_est1,x_buf1']; % store the estimated x
66         para_lst1 = [ para_lst1, [a1(t);b1(j)] ]; % Store parameters a and b in a list
67     end
68 end
69
70 for t = 1:length(a2)
71     for j = 1:length(b2)
72         x_buf2 = sqrt( (1 + (y.^2)./(b2(j)^2) ).*a2(t)^2) + a2(t); % Estimate the value ...
73             of x for given q,e and y
74         std_est2 = [std_est2, std(x - (x_buf2- a2(t)))]; % estimate standard deviation
75         x_est2 = [x_est2,x_buf2']; % store the estimated x
76         para_lst2 = [ para_lst2, [a2(t);b2(j)] ]; % Store parameters a and b in a list
77     end
78 end
79
80 for t = 1:length(a3)
81     for j = 1:length(b3)
82         x_buf3 = sqrt( (1 + (y.^2)./(b3(j)^2) ).*a3(t)^2) + a3(t); % Estimate the value ...
83             of x for given q,e and y
84         std_est3 = [std_est3, std(x - (x_buf3- a3(t)))]; % estimate standard deviation
85         x_est3 = [x_est3,x_buf3']; % store the estimated x
86         para_lst3 = [ para_lst3, [a3(t);b3(j)] ]; % store parameters a and b in a list
87     end
88 end
89
90 %% Plot figure for 50 samples, 3 runs case
91 figure(1)
92 hold on
93 for k = 1:length(a1)
94     plot(x_est1(:,k) - a1(:,k),y');
95 end
96 plot(x,y,'k-o');
97 grid on
98 title('Monte Carlo search: 50 samples, 3 runs');
99 xlabel('x [AU]');
100 ylabel('y [AU]');
101 hold off
102
103 figure(2)
104 hold on
105 for k = 1:length(a2)
106     plot(x_est2(:,k) - a2(:,k),y');
107 end
108 plot(x,y,'k-o');
109 grid on
110 title('Monte Carlo search: 100 samples, 3 runs');
111 xlabel('x [AU]');
112 ylabel('y [AU]');
113 hold off
114
115 figure(3)
116 hold on
117 for k = 1:length(a3)
118     plot(x_est3(:,k) - a3(:,k),y');
119 end
120 plot(x,y,'k-o');
121 grid on
122 title('Monte Carlo search: 500 samples, 3 runs');
123 xlabel('x [AU]');

```

```

122 ylabel('y [AU]');
123 hold off
124
125 %% FINDING OPTIMUM PARAMETERS
126 % Finding the index for minimum standard deviation
127 index1 = find(std_est1 == min(std_est1));
128 index2 = find(std_est2 == min(std_est2));
129 index3 = find(std_est3 == min(std_est3));
130
131
132 % Finding the optimum parameter combination for minimum standard deviation
133 opti_alb1 = para_lst1(:,index1);
134 opti_alb2 = para_lst2(:,index2);
135 opti_alb3 = para_lst3(:,index3);
136
137
138 %% PLOT X AND Y FOR OPTIMUM VALUES OF A AND B
139 figure(11)
140 plot(x_est1(:,index1) + para_lst1(1,index1),y','r-x',x,y,'k-o');
141 grid on
142 legend('Optimum combination of a and b','Observation','Location','best');
143 xlabel('X [AU]');
144 ylabel('Y[AU]');
145 title('Optimum combination of a and b, 50 samples each, 3 runs');
146
147 figure(22)
148 plot(x_est2(:,index2) + para_lst2(1,index2),y','r-x',x,y,'k-o');
149 grid on
150 legend('Optimum combination of a and b','Observation','Location','best');
151 xlabel('X [AU]');
152 ylabel('Y[AU]');
153 title('Optimum combination of a and b, 100 samples each, 3 runs');
154
155 figure(33)
156 plot(x_est3(:,index3) + para_lst3(1,index3),y','r-x',x,y,'k-o');
157 grid on
158 legend('Optimum combination of a and b','Observation','Location','best');
159 xlabel('X [AU]');
160 ylabel('Y[AU]');
161 title('Optimum combination of a and b, 500 samples each, 3 runs');

```

Following is the script used for grid search of Hyperbolic orbit:

```

1  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2  %%% AE4878 Mission Geometry and Orbit Design %%%%%%%%%
3  %%% Assignment 4 - Week 5 - OPTIMIZATION - Version 1 %%%
4  %%% Author Info: Ali Nawaz; Student ID - 4276477 %%%
5  %%% Assigned tasks : OPTIM-5a, QUESTION 6 %%%%%%%%%
6  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
7
8  %% GRID SEARCH FOR HYPERBOLIC PARAMETER ESTIMATION, PARAMETERS : semi transverse axis a and ...
9  % semi-conjugate axis b
10 % close all; clear all;
11 x = [ 0.00 0.02 0.08 0.20 0.33 0.50 0.68 0.88 1.13 1.36 1.58 1.82 2.03 2.32 2.58 2.87 3.07 ...
3.25]; % Distance in AU from slide 28
12 y = [ 0.00 0.11 0.19 0.27 0.34 0.42 0.49 0.55 0.62 0.68 0.73 0.79 0.84 0.89 0.92 0.97 1.02 ...
1.05]; % Distances in AU from slide 28
13
14 uba = 0.5; % Upper bound on a, semi transverse axis
15 lba = -0.5; % Lower bound on a, semi conjugate a
16
17 ubb = sqrt(0.240); % Upper bound on b
18 lbb = 0; % Lower bound on b
19
20 % Store the estimations of x and corresponding standard deviations
21
22 % For 50 samples
23 x_est1 = [];
24 std_est1 = [];
25
26 % For 100 samples
27 x_est2 = [];
28 std_est2 = [];
29
30 % For 500 samples
31 x_est3 = [];

```

```

30 std_est3 = [];
31
32 % List of semi major axis a for sample size 50, 100 and 500
33 a1_lst = [];
34 a2_lst = [];
35 a3_lst = [];
36
37 % List of semi conjugate axis b for sample size 50, 100 and 500
38 b1_lst = [];
39 b2_lst = [];
40 b3_lst = [];
41
42 % Storing a and b parameters for each calculation
43 para_lst1 = []; % 50 samples
44 para_lst2 = []; % 100 samples
45 para_lst3 = []; % 500 samples
46
47 % GRID SEARCH NO MULTIPLE RUNS REQUIRED
48
49 % Grid semi major axis distance generation
50 a1 = lba:((uba-lba)/(50-1)):uba; % 50 samples
51 a2 = lba:((uba-lba)/(100-1)):uba; % 100 samples
52 a3 = lba:((uba-lba)/(500-1)):uba; % 500 samples
53
54 % Grid semi conjugate axis generation
55 b1 = lbb:((ubbb-lbb)/(50-1)):ubbb; % 50 samples
56 b2 = lbb:((ubbb-lbb)/(100-1)):ubbb; % 100 samples
57 b3 = lbb:((ubbb-lbb)/(500-1)):ubbb; % 500 samples
58
59 disp('Running the set of 50 samples...');
60
61 for t = 1:length(a1)
62     for j = 1:length(b1)
63         x_buf1 = sqrt( (1 + (y.^2)./(b1(j)^2) ).*a1(t)^2) + a1(t); % Estimate the value of x ...
64             for given q,e and y
65                 std_est1 = [std_est1, std(x - (x_buf1- a1(t)))]; % estimate standard deviation
66                 x_est1 = [x_est1,x_buf1']; % store the estimated x
67                 para_lst1 = [ para_lst1, [a1(t);b1(j)] ]; % Store parameters a and b in a list
68             end
69         end
70     disp('Running the set of 100 samples...');
71     for t = 1:length(a2)
72         for j = 1:length(b2)
73             x_buf2 = sqrt( (1 + (y.^2)./(b2(j)^2) ).*a2(t)^2) + a2(t); % Estimate the value of x ...
74                 for given q,e and y
75                     std_est2 = [std_est2, std(x - (x_buf2- a2(t)))]; % estimate standard deviation
76                     x_est2 = [x_est2,x_buf2']; % store the estimated x
77                     para_lst2 = [ para_lst2, [a2(t);b2(j)] ]; % Store parameters a and b in a list
78                 end
79             end
80         disp('Running the set of 500 samples');
81         for t = 1:length(a3)
82             for j = 1:length(b3)
83                 x_buf3 = sqrt( (1 + (y.^2)./(b3(j)^2) ).*a3(t)^2) + a3(t); % Estimate the value of x ...
84                     for given q,e and y
85                         std_est3 = [std_est3, std(x - (x_buf3- a3(t)))]; % estimate standard deviation
86                         x_est3 = [x_est3,x_buf3']; % store the estimated x
87                         para_lst3 = [ para_lst3, [a3(t);b3(j)] ]; % store parameters a and b in a list
88                     end
89                 end
90             end
91         %% Plot figure for 50 samples, 3 runs case
92         figure(1)
93         hold on
94         for k = 1:length(a1)
95             plot(x_est1(:,k) - a1(:,k),y');
96         end
97         plot(x,y,'k-o');
98         grid on
99         title('Monte Carlo search: 50 samples, 3 runs');
100         xlabel('x [AU]');
101         ylabel('y [AU]');
102         hold off
103     end
104     figure(2)

```

```

103 hold on
104 for k = 1:length(a2)
105     plot(x_est2(:,k) - a2(:,k),y');
106 end
107 plot(x,y,'k-o');
108 grid on
109 title('Monte Carlo search: 100 samples, 3 runs');
110 xlabel('x [AU]');
111 ylabel('y [AU]');
112 hold off
113
114 figure(3)
115 hold on
116 for k = 1:length(a3)
117     plot(x_est3(:,k) - a3(:,k),y');
118 end
119 plot(x,y,'k-o');
120 grid on
121 title('Monte Carlo search: 500 samples, 3 runs');
122 xlabel('x [AU]');
123 ylabel('y [AU]');
124 hold off
125
126 %% FINDING OPTIMUM PARAMETERS
127 % Finding the index for minimum standard deviation
128 index1 = find(std_est1 == min(std_est1));
129 index2 = find(std_est2 == min(std_est2));
130 index3 = find(std_est3 == min(std_est3));
131
132
133 % Finding the optimum parameter combination for minimum standard deviation
134 opti_alb1 = para_lst1(:,index1);
135 opti_alb2 = para_lst2(:,index2);
136 opti_alb3 = para_lst3(:,index3);
137
138
139
140 %% PLOT X AND Y FOR OPTIMUM VALUES OF A AND B
141 figure(11)
142 plot(x_est1(:,index1(1,1)) - para_lst1(1,index1(1,1)),y', 'r-x',x,y,'k-o');
143 grid on
144 legend('Optimum combination of a and b','Observation','Location','best');
145 xlabel('X [AU]');
146 ylabel('Y[AU]');
147 title('Optimum combination of a and b, 50 samples each, 3 runs');
148
149 figure(22)
150 plot(x_est2(:,index2(1,1)) - para_lst2(1,index2(1,1)),y', 'r-x',x,y,'k-o');
151 grid on
152 legend('Optimum combination of a and b','Observation','Location','best');
153 xlabel('X [AU]');
154 ylabel('Y[AU]');
155 title('Optimum combination of a and b, 100 samples each, 3 runs');
156
157 figure(33)
158 plot(x_est3(:,index3(1,1)) - para_lst3(1,index3(1,1)),y', 'r-x',x,y,'k-o');
159 grid on
160 legend('Optimum combination of a and b','Observation','Location','best');
161 xlabel('X [AU]');
162 ylabel('Y[AU]');
163 title('Optimum combination of a and b, 500 samples each, 3 runs');

```

Following is the script for quality vs time comparison of hyperbolic orbit:

```

1  %% COMPARING MONTE CARLO AND GRID SEARCH COMPUTATIONAL EFFORT VS QUALITY
2  %no. of samples
3  samples = [ 50 100 500 ];
4
5  %no. of calculations, m = Monte Carlo, g = grid search
6  calc_m = 3.*samples;
7  calc_g = samples;
8
9  %corresponding standard deviations
10 std_m = [0.2344 0.2311 0.2303];
11 std_g = [ 0.2302 0.2306 0.2302];

```



```
12
13 % Plot the results
14 figure(41)
15 plot(calc_m,std_m,'r-x',calc_g,std_g,'k-o');
16 grid on
17 legend('Monte Carlo','Grid Search');
18 ylabel('Standard deviation [AU]');
19 xlabel('No. of calculations');
20 title('Quality vs Time plot');
```

References

- [1] R. Noomen. *AE4878 Mission Geometry and Orbit Design*. TU Delft, 2017.
- [2] James R. Wertz. *Orbit Constellation Design Management*. Springer, 2009.