AE 551: Introduction to Optimal Control **(Due: 2020/05/15)**

Homework #7 Submission

Instructor: Min-Jea Tahk *Name:* Joshua Julian Damanik, *Student ID:* 20194701

Main code for problem 1 and 2 is attached at the Appendix section. However the rest of the code can be accessed from `https://github.com/joshuadamanik/Homework-7`.

Problem 1: Augmented Lagrangian Method for Equality Constraints

$$\min \quad f(x_1, x_2) = \frac{1}{2}(x_1 - 1)^2 + 10(x_2 - 1)^2 \tag{1}$$

$$\text{subject to} \quad c(x_1, x_2) = (x_1 - 2)^2 + 2 - x_2 = 0 \tag{2}$$

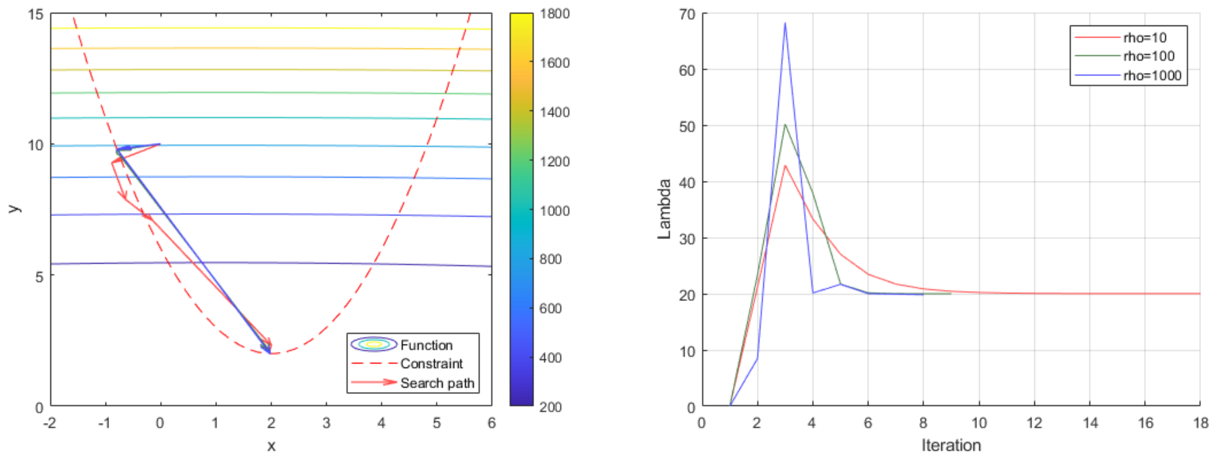
(Solution)

To solve the optimization problem on equation 1, Augmented Lagrangian method for equality constraint is used. It defines an augmented Lagrange function as

$$L_A(X, \lambda_k, \rho) = f(X) + \lambda_k^T c(X) + \rho c(X)^T c(X) \tag{3}$$

where $X = (x_1, x_2)$, $\rho = \{10, 100, 1000\}$, and $\lambda_k = \lambda_{k-1} + 2\rho c(\bar{X})$ with $\lambda_0 = 0$.

The result of the simulation is shown at figure 1. Figure 1a shows the path of the search which is initialized at $X_0 = (x_1, x_2)_0 = (0, 10)$. The search took 18, 9, and 8 iterations for $\rho = 10, 100, 1000$ respectively. Figure 1b shows the value of λ_k at each iteration. Table 1 shows the value of X and $c(X)$ at each iterations.



(a) Search path (b) λ_k value at each iteration

Figure 1: Simulation result of problem 1

Table 1: Simulation data of problem 1

(a) $\rho = 10$

Iteration	x_1	x_2	$c(x_1, x_2)$
1	0	10	-4
2	-0.8868	9.2751	1.058514
3	-0.6397	7.8836	1.084416
4	-0.1447	7.0784	-0.47866
5	2.0026	2.314	-0.31399
6	1.977	2.1782	-0.17767
7	1.9767	2.0875	-0.08696
8	1.9761	2.0436	-0.04303
9	1.976	2.0218	-0.02122
10	1.9758	2.011	-0.01041
11	1.9759	2.0058	-0.00522
12	1.9759	2.0032	-0.00262
13	1.9759	2.0019	-0.00132
14	1.9759	2.0013	-0.00072
15	1.9759	2.0009	-0.00032
16	1.9759	2.0008	-0.00022
17	1.9759	2.0007	-0.00012
18	1.9759	2.0007	-0.00012

(b) $\rho = 100$

x_1	x_2	$c(x_1, x_2)$
0	10	-4
-0.8042	9.7468	0.116738
-0.7846	9.6197	0.134297
-0.7368	9.5516	-0.06153
1.9782	2.0813	-0.08082
1.9753	2.0084	-0.00779
1.9755	2.0013	-0.0007
1.976	2.0006	-2.4E-05
1.976	2.0006	-2.4E-05

(c) $\rho = 1000$

x_1	x_2	$c(x_1, x_2)$
0	10	-4
-0.793	9.7968	0.004049
-0.7936	9.7745	0.029701
-0.7836	9.7725	-0.02407
1.9823	1.9995	0.000813
1.9826	2.0011	-0.0008
1.9828	2.0003	-4.2E-06
1.9828	2.0003	-4.2E-06

Problem 2: Augmented Lagrangian Method for Inequality Constraints

$$\min \quad f(x_1, x_2) = (1 - x_1)^2 + 100(x_2 - x_1^2)^2 \tag{4}$$

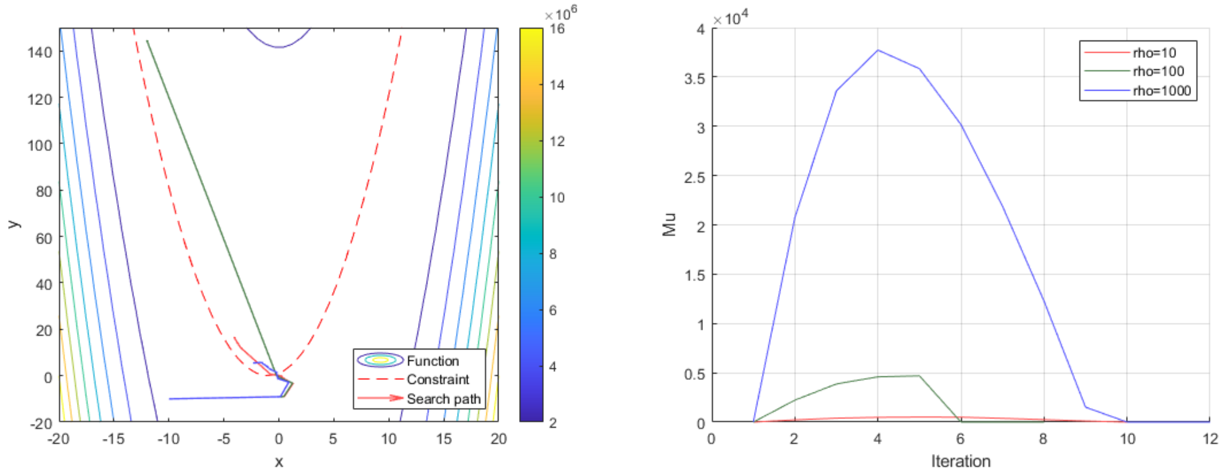
$$\text{subject to} \quad d(x_1, x_2) = (1 + x_1)^2 - x_2 \leq 0 \tag{5}$$

To solve the optimization problem on equation 4, Augmented Lagrangian method for inequality constraint is used. It defines an augmented Lagrange function as

$$L_A(X, \mu_k, \rho) = f(X) + \rho \sum_i \max^2 \left\{ d_i + \frac{\mu_{k,i}}{2\rho}, 0 \right\} - \sum_i \frac{\mu_{k,i}^2}{4\rho} \tag{6}$$

where $X = (x_1, x_2)$, $\rho = \{10, 100, 1000\}$, and $\mu_{k,i} = \max\left\{\mu_{k-1,i} + 2\rho d_i(\bar{X}), 0\right\}$ with $\mu_0 = 0$.

The result of the simulation is shown at figure 2. Figure 2a shows the path of the search which is initialized at $X_0 = (x_1, x_2)_0 = (-10, -10)$. The search took 11, 8, and 12 iterations for $\rho = 10, 100, 1000$ respectively. Figure 2b shows the value of μ_k at each iteration. Table 2 shows the value of X and $d(X)$ at each iterations.



(a) Search path

(b) μ_k value at each iteration

Figure 2: Simulation result of problem 2

Table 2: Simulation data of problem 2

(a) $\rho = 10$

Iteration	x_1	x_2	$c(x_1, x_2)$
1	-10	-10	91
2	0.4953	-9.021	11.25692
3	1.2975	-3.3125	8.591006
4	0.6054	-1.7188	4.296109
5	0.1384	-0.0026	1.298555
6	-0.7683	0.6203	-0.56662
7	-3.4773	12.0751	-5.93808
8	-3.7711	14.1817	-6.5027
9	-4.0541	16.4001	-7.07257
10	-4.049	16.4006	-7.1042
11	-4.049	16.4006	-7.1042

(b) $\rho = 100$

x_1	x_2	$c(x_1, x_2)$
-10	-10	91
0.4434	-9.0412	11.12460356
1.248	-3.1612	8.214704
0.4189	-1.6031	3.61637721
-0.2192	0.1219	0.48774864
-11.9967	144.3666	-23.43918911
-12.015	144.3658	-23.035575
-12.015	144.3658	-23.035575

(c) $\rho = 1000$

x_1	x_2	$c(x_1, x_2)$
-10	-10	91
0.139	-9.0673	10.36462
0.8997	-2.8187	6.42756
-0.0957	-1.2534	2.071158
-0.2267	1.543	-0.94501
-0.8201	2.8653	-2.83294
-1.124	4.1552	-4.13982
-1.3992	4.97	-4.81064
-1.5834	5.7193	-5.37894
-2.3393	5.4793	-3.68558
-2.3392	5.4793	-3.68584
-2.3392	5.4793	-3.68584

Appendix

Listing 1: Main code for problem 1

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% HOMEWORK #7
% Joshua Julian Damanik (20194701)
% AE551 — Introduction to Optimal Control
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear, clc, close all;
addpath('lib');

eps = 0.1;
k = 15;

rho_list = [10, 100, 1000];
delta_rho = 2;

iter = 0;
```

```
X_init = [0; 10];

f = @(X) (0.5*(X(1)-1).^2+10*(X(2)-1).^2);
c = @(X) (X(1)-2)^2+2-X(2);

for jj = 1:3
    X = X_init;
    rho = rho_list(jj);
    iter = iter + 1;

    La = @(X, lambda) (f(X) + lambda'*c(X)+rho.*c(X)'*c(X));

    lambda = zeros(length(1),1);
    X_data{jj} = X;
    lambda_data{jj} = lambda;

    for ii = 2:10000
        LaX = @(X) La(X,lambda);
        X_bar = minimize(X, eps, k, LaX);

        lambda = lambda + 2*rho*c(X_bar);
        fprintf('jj=%d,ii=%d\n',jj,ii);

        X_data{jj}(:,ii) = X_bar;
        lambda_data{jj}(:,ii) = lambda;

        X = X_bar;

        if norm(X_data{jj}(:,ii)-X_data{jj}(:,ii-1))/norm(X_data{jj}(:,ii)) < 1e-10
            break;
        end
    end
end

%% Function Graph

N_grid = 50;
axis_xy = [-2 6 0 15];
x_cont = linspace(axis_xy(1), axis_xy(2), N_grid);
y_cont = linspace(axis_xy(3), axis_xy(4), N_grid);
[X_cont, Y_cont] = meshgrid(x_cont, y_cont);

F_cont = zeros(N_grid);
Cy_cont = zeros(N_grid);

for i=1:N_grid
    for j=1:N_grid
        F_cont(i,j) = f([X_cont(i,j), Y_cont(i,j)]');
    end
end

figure(1);
s = contour(X_cont, Y_cont, F_cont);
colorbar;
hold on;

%% Constraint Graph

C_data = (x_cont-2).^2+2;
plot(x_cont, C_data, 'r--');
axis(axis_xy);

%% Search Path Graph
```

```
color = [1, 0.3, 0.3;
         0.3, 0.5, 0.3;
         0.3, 0.3, 1];
for j=1:3
    points = X_data{j};
    for i=1:size(points,2)-1
        F_data = f(points(:,i));
        qlen = [points(:,i+1) - points(:,i)];% f(X_data(:,i+1))-F_data];
        quiver(points(1,i), points(2,i), ...% F_data, ...
                    qlen(1), qlen(2), ... % qlen(3), ...
                    'r', 'AutoScale', 'off', 'LineWidth', 1, ...
                    'MaxHeadSize', min(1 / norm(qlen),1), ...
                    'color', color(j,:));
    end
end
xlabel('x');
ylabel('y');
zlabel('z');
legend('Function', 'Constraint', 'Search path', 'Location', 'SouthEast');

%% Lambda Graph
figure(2); hold on;
for j=1:3
    p=plot(1:length(lambda_data{j}), lambda_data{j}, 'Color', color(j,:));
end
grid on;
xlabel('Iteration');
ylabel('Lambda');
legend('rho=10', 'rho=100', 'rho=1000');
```

Listing 2: Main code for problem 2

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% HOMEWORK #7
% Joshua Julian Damanik (20194701)
% AE551 - Introduction to Optimal Control
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear, clc, close all;
addpath('lib');

eps = 0.1;
k = 15;

rho_list = [10, 100, 1000];
delta_rho = 2;

iter = 0;

X_init = [-10, -10]';

f = @(X) (1-X(1)).^2+100*(X(2)-X(1).^2).^2;
d = @(X) ((1+X(1)).^2-X(2));

for jj = 1:3
    X = X_init;
    rho = rho_list(jj);
    iter = iter + 1;

    La = @(X, mu) (f(X) + rho*(max(0,d(X)+mu/(2*rho))-mu.^2/(4*rho));

    mu = zeros(length(1),1);
    X_data{jj} = X;
```

```
    mu_data{jj} = mu;

    for ii = 2:10000
        LaX = @(X) La(X,mu);
        X_bar = minimize(X, eps, k, LaX);

        mu = max(0,mu + 2*rho*d(X_bar));
        fprintf('jj=%d,ii=%d\n',jj,ii);

        X_data{jj}(:,ii) = X_bar;
        mu_data{jj}(:,ii) = mu;

        X = X_bar;

        if norm(X_data{jj}(:,ii)-X_data{jj}(:,ii-1))/norm(X_data{jj}(:,ii)) < 1e-10
            break;
        end
    end
end

%% Function Graph

N_grid = 50;
axis_xy = [-20 20 -20 150];
x_cont = linspace(axis_xy(1), axis_xy(2), N_grid);
y_cont = linspace(axis_xy(3), axis_xy(4), N_grid);
[X_cont, Y_cont] = meshgrid(x_cont, y_cont);

F_cont = zeros(N_grid);
Cy_cont = zeros(N_grid);

for i=1:N_grid
    for j=1:N_grid
        F_cont(i,j) = f([X_cont(i,j), Y_cont(i,j)]');
    end
end

figure(1);
s = contour(X_cont, Y_cont, F_cont);
colorbar;
hold on;

%% Constraint Graph

C_data = (1+x_cont).^2;
plot(x_cont, C_data, 'r--');
axis(axis_xy);

%% Search Path Graph
color = [1, 0.3, 0.3;
         0.3, 0.5, 0.3;
         0.3, 0.3, 1];
for j=1:3
    points = X_data{j};
    for i=1:size(points,2)-1
        F_data = f(points(:,i));
        qlen = [points(:,i+1) - points(:,i)];% f(X_data(:,i+1))-F_data];
        quiver(points(1,i), points(2,i), ...% F_data, ...
                        qlen(1), qlen(2), ... % qlen(3), ...
                        'r', 'AutoScale', 'off', 'LineWidth', 1, ...
                        'MaxHeadSize', min(1 / norm(qlen),1), ...
                        'color', color(j,:));
    end
```

```
end
xlabel('x');
ylabel('y');
zlabel('z');
legend('Function', 'Constraint', 'Search path', 'Location', 'SouthEast');

%% Mu Graph
figure(2); hold on;
for j=1:3
    p=plot(1:length(mu_data{j}), mu_data{j}, 'Color', color(j,:));
end
grid on;
xlabel('Iteration');
ylabel('Mu');
legend('rho=10', 'rho=100', 'rho=1000');
```

Listing 3: Code for function *minimize*

```
function X_star = minimize(X, eps, k, f)
    quasi = quasi_newton_class(length(X));
    for n = 1:1000
        p = quasi.bgfs(X, eps, f);
        [Xa, Xb] = unimodal_interval(X, eps, p, f);
        X_star = fibonacci_search(Xa, Xb, k, f);

        err = norm(X_star - X);
        if (err < eps * 0.01)
            break;
        end
        X = X_star;
    end
end
```