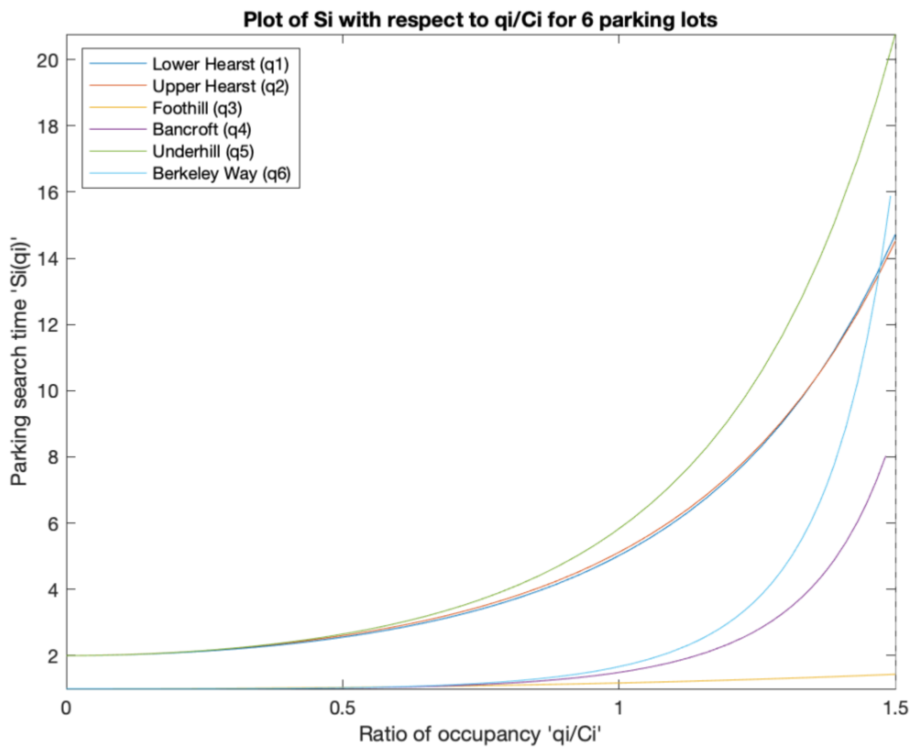
William Wijaya
3035992465

Lab #2
Due: Friday 2/27 at 11:59pm

Lab 2: Parking Guidance System

Problem 1

The graph for the problem is shown below:



The code for no 1 is boxed below:

```
%% 1

clear all,clc;

fprintf("No 1 plot loading... \n\n")
%Constructing a parameter table
Parking_Lots= {"Lower Hearst (q1)";"Upper Hearst (q2)";"Foothill (q3)";...
    "Bancroft (q4)";"Underhill (q5)";"Berkeley Way (q6)"};
alpha=[0.92;0.94;0.16;0.4;1.07;0.51];
beta=[1.91;1.84;2.02;4.19;1.93;4.23];
C=[600;280;183;232;920;100];
t0=[2;2;1;1;2;1];
tw=[3;4;13;5;7;7];
ParameterT=table(Parking_Lots,alpha,beta,C,t0,tw);

%Plot function
syms x
```

```
S(x)=t0.*exp(alpha.*(x).^beta);
fplot(S)
xlim([0,1.5])
title("Plot of Si with respect to qi/Ci for 6 parking lots")
xlabel("Ratio of occupancy 'qi/Ci'")
ylabel("Parking search time 'Si(qi)'")
legend(Parking_Lots,'Location','northwest')

fprintf("Done!\n")
fprintf("Press enter to continue no 2a: \n");pause;clc
```

Problem 2

(a)
The table for $q_i *$, S($q_i *$), and T($q_i *$) for the problem is shown below:

qn	q*
q1	838.269076
q2	380.270447
q3	270.825653
q4	350.107895
q5	1022.31772
q6	138.209208

Sn	S(q*)
S1	11.4236003
S2	10.4235998
S3	1.4235944
S4	9.42359955
S5	7.42360299
S6	7.42359943

Tn	T(q*)
T1	14.4236
T2	14.4236
T3	14.4236
T4	14.4236
T5	14.4236
T6	14.4236

Discussion:

It comes out that the required q for each parking lot to minimize the search time is shown on the first table on the left. If we see the third table on the left, we can see that all T(q*) in each Tn is the same. This does make sense. Drivers have perfect knowledge about parking costs and choose the parking lot with minimal cost. Thus, under user deterministic equilibrium, if each person wants to choose parking lot with minimal cost, it is expected that the total time cost of parking lot “i” will be the same for all “i”.

The code for 2(a) is shown below:

```
%% 2a

%Objective function

% Defining gradient
Q0=3000;

T1=@(q1) 2.*exp(0.92.*(q1./600).^1.91)+3;
T2=@(q2) 2.*exp(0.94.*(q2./280).^1.84)+4;
T3=@(q3) 1.*exp(0.16.*(q3./183).^2.02)+13;
T4=@(q4) 1.*exp(0.4.*(q4./232).^4.19)+5;
T5=@(q5) 2.*exp(1.07.*(q5./920).^1.93)+7;
T6=@(q1,q2,q3,q4,q5) 1.*exp(0.51.*((Q0-(q1+q2+q3+q4+q5))./100).^4.23)+7;

% Start point
C1_5=[600;280;183;232;920];

x1=(C1_5(1)/sum(C1_5))*Q0;
x2=(C1_5(2)/sum(C1_5))*Q0;
x3=(C1_5(3)/sum(C1_5))*Q0;
x4=(C1_5(4)/sum(C1_5))*Q0;
x5=(C1_5(5)/sum(C1_5))*Q0;

% Gradient method first iteration

a= 0.1; %alpha

vector=[x1,x2,x3,x4,x5]'-a.*...
    [T1(x1)-T6(x1,x2,x3,x4,x5),...
    T2(x2)-T6(x1,x2,x3,x4,x5),...
    T3(x3)-T6(x1,x2,x3,x4,x5),...
    T4(x4)-T6(x1,x2,x3,x4,x5),...
    T5(x5)-T6(x1,x2,x3,x4,x5)]';

% Store first iteration and second iteration in matrix P
P_2a=zeros(5,3);
P_2a(:,1)=[x1,x2,x3,x4,x5];
P_2a(:,2)=[vector(1),vector(2),vector(3),vector(4),vector(5)];
i=1;

% Matrix A is the stopping condition
A=(sum(abs(P_2a(:,1+1)-P_2a(:,1))));

% Gradient method iteration while loop
 while A>= 10^(-6)
     x1=vector(1);
     x2=vector(2);
     x3=vector(3);
     x4=vector(4);
     x5=vector(5);
```

```
    %Initialize iteration of gradient method
    vector=[x1,x2,x3,x4,x5]'-a.*...
        [T1(x1)-T6(x1,x2,x3,x4,x5),...
        T2(x2)-T6(x1,x2,x3,x4,x5),...
        T3(x3)-T6(x1,x2,x3,x4,x5),...
        T4(x4)-T6(x1,x2,x3,x4,x5),...
        T5(x5)-T6(x1,x2,x3,x4,x5)]';

    % Store each iteration in Matrix P
    P_2a(:,i+2)=[vector(1),vector(2),vector(3),vector(4),vector(5)];

    i=i+1;

    % Update the stopping condition
    A=(sum(abs(P_2a(:,i+1)-P_2a(:,i))));
end

% q* table
fprintf("Table of qi* below:\n")
q1_star=P_2a(1,end);
q2_star=P_2a(2,end);
q3_star=P_2a(3,end);
q4_star=P_2a(4,end);
q5_star=P_2a(5,end);
q6_star=Q0-(q1_star+q2_star+q3_star+q4_star+q5_star);
qn=["q1";"q2";"q3";"q4";"q5";"q6"];
qsol=[q1_star;q2_star;q3_star;q4_star;q5_star;q6_star];
qsolT=table(qn,qsol)

% Si(qi*) Table
fprintf("Table of Si* below:\n")

S1=@(q1) 2.*exp(0.92.*(q1./600).^1.91);
S2=@(q2) 2.*exp(0.94.*(q2./280).^1.84);
S3=@(q3) 1.*exp(0.16.*(q3./183).^2.02);
S4=@(q4) 1.*exp(0.4.*(q4./232).^4.19);
S5=@(q5) 2.*exp(1.07.*(q5./920).^1.93);
S6=@(q1,q2,q3,q4,q5) 1.*exp(0.51.*((Q0-(q1+q2+q3+q4+q5))./100).^4.23);

Sn=["S1","S2","S3","S4","S5","S6"]';
Ssol=[S1(q1_star),S2(q2_star),S3(q3_star),S4(q4_star),S5(q5_star),...
    S6(q1_star,q2_star,q3_star,q4_star,q5_star)]';
SsolT=table(Sn,Ssol)

% Ti(qi*) Table
fprintf("Table of Ti* below:\n")

Tn=["T1","T2","T3","T4","T5","T6"]';
Tsol=[T1(q1_star),T2(q2_star),T3(q3_star),T4(q4_star),T5(q5_star),...
    T6(q1_star,q2_star,q3_star,q4_star,q5_star)]';
TsolT=table(Tn,Tsol)

fprintf("Press enter to continue 2d: \n");pause;clc
```

(b) We need to formulate the nonlinear problem into the Laplacian function. Thus,

$$\mathcal{L}(\vec{q}, \lambda_1) = \left(\sum_{i=1}^{6} \int_0^{q_i} T_i(x)dx\right) + \lambda_1 \left(\left(\sum_{i=1}^{6} q_i\right) - Q_0\right)$$

Formulate the gradient of the Laplacian and it needs to be equal to zero:

$$\nabla \mathcal{L}_{\vec{q}}(\vec{q}, \lambda_1) = T_i(q_i)^* + \lambda_1{}^* = 0$$

Solving for $\lambda_1{}^*$, where $T_i(q_i)^*$ is already known from the table 2(a)

$$\lambda_1{}^* = -T_i(q_i)^*$$

$$\lambda_1{}^* = -[14.4236\ \ 14.4236\ \ 14.4236\ \ 14.4236\ \ 14.4236\ \ 14.4236]^T$$

Thus, we can see if $q_i{}^*$ are all strictly positive, then total parking time for each parking lot $T_i(q_i)^*$ will be the same.

(c) From KKT condition,

Stationarity condition of KKT condition:

$$T_i(q_i{}^*) + \lambda_i - \mu_i = 0$$

Suppose that $q_i{}^* = 0$

$$T_i(0) + \lambda_i - \mu_i = 0$$

$$T_i(0) - \mu_i = -\lambda_i$$

Since that $T_i(0) > -\lambda_i$, then $\mu_i > 0$

But from complimentary condition:

$$\mu_i * q_i(0) = 0$$

Since $\mu_i > 0$, then $q_i(0) = 0$

Q.E.D

d) To prove the claim of the problem, remove q_3 from the problem (i.e., assume that $q_3 = 0$). Then, it is expected that ${q_i}^*$ are all positive for other five lots. After coding was done, I got the following q* in a table:

qn	q*
q1	703.989622
q2	304.001825
q3	0
q4	323.14486
q5	549.233623
q6	119.63007

We can see that ${q_i}^*$ are all positive for other five lots. Now, to prove whether $q_3 = 0$ is a valid optimal solution, let us look the table of Tn:

Tn	Tsol
T1	9.96986094
T2	9.96986268
T3	14
T4	9.96986361
T5	9.96986917
T6	9.96986402

From 2(c) claim, if $T_3(0) > -\lambda_3$, then $q_3 = 0$ is a valid optimal solution. $T_3(0) > -\lambda_3$ holds true if we look from the table of Tn, so $q_3 = 0$ is a valid optimal solution. Thus, it is true that the inequality constraint for q_3 is active, while the ${q_i}^*$ are all positive for other five lots.

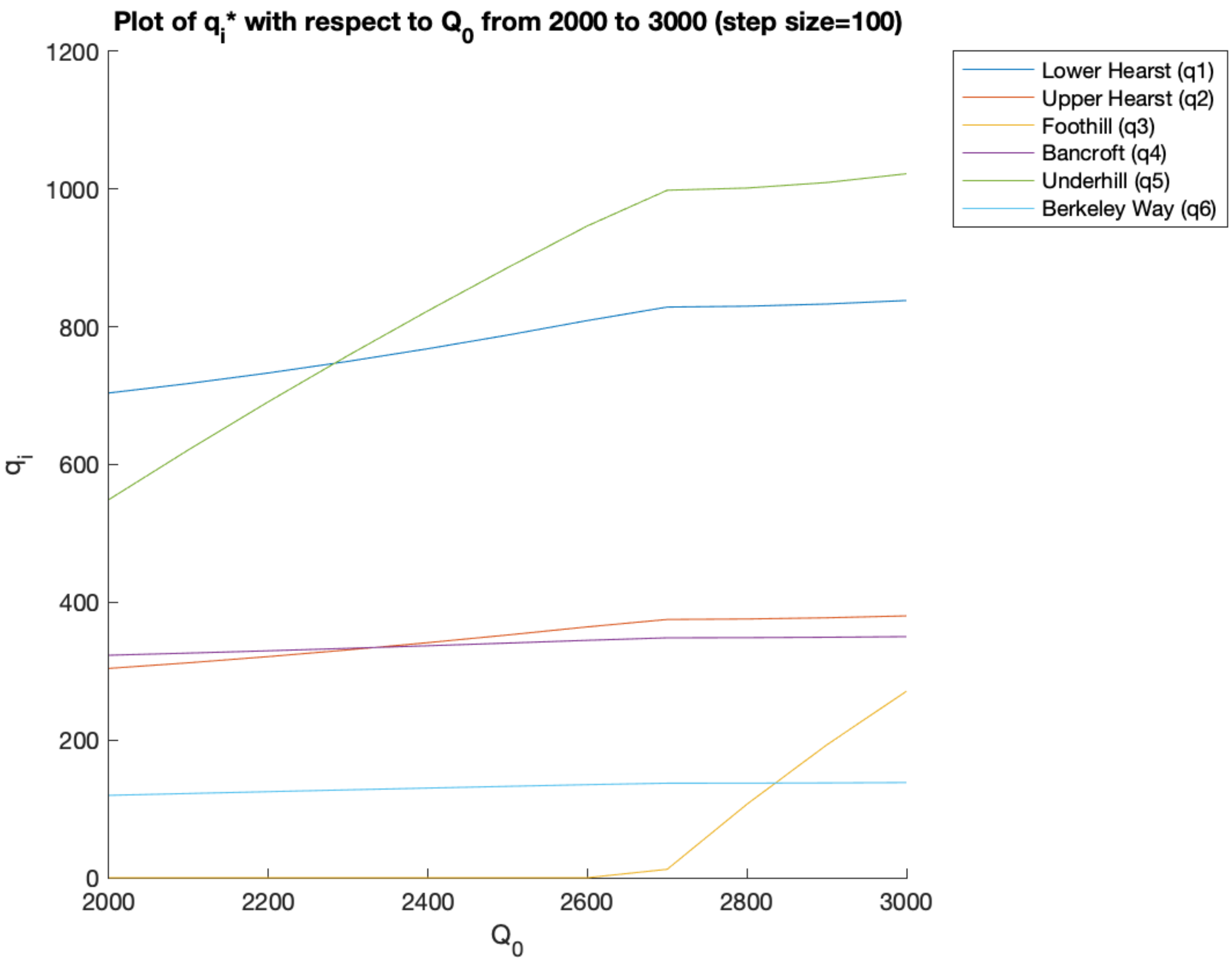
Coding:

```
%% 2d

% In this problem, remove q3 from the nonlinear problem.
Q0=2000;

% Start point
C1_4=[600;280;232;920];%Remove C3

x1=(C1_4(1)/sum(C1_4))*Q0;
x2=(C1_4(2)/sum(C1_4))*Q0;
x4=(C1_4(3)/sum(C1_4))*Q0;
x5=(C1_4(4)/sum(C1_4))*Q0;

T6=@(q1,q2,q4,q5) 1.*exp(0.51.*((Q0-(q1+q2+q4+q5))./100).^4.23)+7;
```

```matlab
%Gradient method first iteration

vector=[x1,x2,x4,x5]'-a.*...
    [T1(x1)-T6(x1,x2,x4,x5),...
    T2(x2)-T6(x1,x2,x4,x5),...
    T4(x4)-T6(x1,x2,x4,x5),...
    T5(x5)-T6(x1,x2,x4,x5)]';



P_2d=zeros(4,3);
P_2d(:,1)=[x1,x2,x4,x5];
P_2d(:,2)=[vector(1),vector(2),vector(3),vector(4)];
i=1;

% Matrix A is the stopping condition
A_2d=(sum(abs(P_2d(:,1+1)-P_2d(:,1))));

% Gradient method iteration while loop
while A_2d>= 10^(-6)
     x1=vector(1);
     x2=vector(2);
     x4=vector(3);
     x5=vector(4);

     %Initialize iteration of gradient method
     vector=[x1,x2,x4,x5]'-a.*...
         [T1(x1)-T6(x1,x2,x4,x5),...
         T2(x2)-T6(x1,x2,x4,x5),...
         T4(x4)-T6(x1,x2,x4,x5),...
         T5(x5)-T6(x1,x2,x4,x5)]';

     % Store first iteration and second iteration in matrix P
     P_2d(:,i+2)=[vector(1),vector(2),vector(3),vector(4)];

     i=i+1;

     % Update the stopping condition.
     A_2d=(sum(abs(P_2d(:,i+1)-P_2d(:,i))));
end

% q* table
fprintf("Table of qi* below:\n")
q1_star2d=P_2d(1,end);
q2_star2d=P_2d(2,end);
q3_star2d=0;
q4_star2d=P_2d(3,end);
q5_star2d=P_2d(4,end);
q6_star2d=Q0-(q1_star2d+q2_star2d+q3_star2d+q4_star2d+q5_star2d);
qn=["q1";"q2";"q3";"q4";"q5";"q6"];
qsol=[q1_star2d;q2_star2d;q3_star2d;q4_star2d;q5_star2d;q6_star2d];
qsolT=table(qn,qsol)

fprintf("Suppose inequality constraint for q3 is inactive \n")
```

```
fprintf("If we remove q3 from the problem,")
fprintf(" then the other qi will be strictly positive \n\n")

fprintf("Press enter to continue no 2e: \n");pause;clc
```

(e) After trial and error, I found that when q_3 should be equal to zero, when $Q_0 = 2000$ until $Q_0 = 2600$. And, q_3 should be added when $Q_0 = 2700$ until $Q_0 = 3000$. After coding was done, the following graph is generated from $Q_0 = 2000$ until $Q_0 = 3000$:



From the graph, we can see that q_3 start rising when $Q_0 = 2700$. In addition, when q_3 starts to be strictly positive at $Q_0 = 2700$, the graph for q_3 starts to increase at a faster rate, while the graph for other q_i starts to increase at a slower rate.

Coding:

```
%% 2e

%Objective function

% When I try testing each Q0 manually, I find that:
% The q3 should be removed when Q0=2000 until Q0=2600
% The q3 should be added when Q0=2700 until Q0=3000

clear all

% Defining a matrix where each column represents each Q0.
j=0;
Qgraph=zeros(6,11);

for Q0=2000:100:2600
% In this problem, remove q3 from the nonlinear problem.
a=0.1;

% Start point
C1_4=[600;280;232;920];%Remove C3

x1=(C1_4(1)/sum(C1_4))*Q0;
x2=(C1_4(2)/sum(C1_4))*Q0;
x4=(C1_4(3)/sum(C1_4))*Q0;
x5=(C1_4(4)/sum(C1_4))*Q0;

T1=@(q1) 2.*exp(0.92.*(q1./600).^1.91)+3;
T2=@(q2) 2.*exp(0.94.*(q2./280).^1.84)+4;
T3=@(q3) 1.*exp(0.16.*(q3./183).^2.02)+13;
T4=@(q4) 1.*exp(0.4.*(q4./232).^4.19)+5;
T5=@(q5) 2.*exp(1.07.*(q5./920).^1.93)+7;
T6=@(q1,q2,q4,q5) 1.*exp(0.51.*((Q0-(q1+q2+q4+q5))./100).^4.23)+7;

%Gradient method first iteration

vector=[x1,x2,x4,x5]'-a.*...
    [T1(x1)-T6(x1,x2,x4,x5),...
    T2(x2)-T6(x1,x2,x4,x5),...
    T4(x4)-T6(x1,x2,x4,x5),...
    T5(x5)-T6(x1,x2,x4,x5)]';


P_2d=zeros(4,3);
P_2d(:,1)=[x1,x2,x4,x5];
P_2d(:,2)=[vector(1),vector(2),vector(3),vector(4)];
i=1;
```

```
% Matrix A is the stopping condition
A_2d=(sum(abs(P_2d(:,1+1)-P_2d(:,1))));

% Gradient method iteration while loop
while A_2d>= 10^(-6)
     x1=vector(1);
     x2=vector(2);
     x4=vector(3);
     x5=vector(4);

     %Initialize iteration of gradient method
     vector=[x1,x2,x4,x5]'-a.*...
         [T1(x1)-T6(x1,x2,x4,x5),...
         T2(x2)-T6(x1,x2,x4,x5),...
         T4(x4)-T6(x1,x2,x4,x5),...
         T5(x5)-T6(x1,x2,x4,x5)]';

     % Store first iteration and second iteration in matrix P
     P_2d(:,i+2)=[vector(1),vector(2),vector(3),vector(4)];

     i=i+1;

     % Update the stopping condition.
     A_2d=(sum(abs(P_2d(:,i+1)-P_2d(:,i))));
end

% q* table
fprintf("Table of qi* (Q0=%4.2d) below:\n", Q0)
q1_star2d=P_2d(1,end);
q2_star2d=P_2d(2,end);
q3_star2d=0;
q4_star2d=P_2d(3,end);
q5_star2d=P_2d(4,end);
q6_star2d=Q0-(q1_star2d+q2_star2d+q3_star2d+q4_star2d+q5_star2d);
qn=["q1";"q2";"q3";"q4";"q5";"q6"];
qsol=[q1_star2d;q2_star2d;q3_star2d;q4_star2d;q5_star2d;q6_star2d];
qsolT=table(qn,qsol)

% Save each q* from each Q0 to matrix Qgraph
Qgraph(:,j+1)=qsol;

j=j+1;
end

for Q0=2700:100:3000

% Defining gradient

T1=@(q1) 2.*exp(0.92.*(q1./600).^1.91)+3;
T2=@(q2) 2.*exp(0.94.*(q2./280).^1.84)+4;
T3=@(q3) 1.*exp(0.16.*(q3./183).^2.02)+13;
T4=@(q4) 1.*exp(0.4.*(q4./232).^4.19)+5;
T5=@(q5) 2.*exp(1.07.*(q5./920).^1.93)+7;
T6=@(q1,q2,q3,q4,q5) 1.*exp(0.51.*((Q0-(q1+q2+q3+q4+q5))./100).^4.23)+7;
```

```
% Start point
C1_5=[600;280;183;232;920];

x1=(C1_5(1)/sum(C1_5))*Q0;
x2=(C1_5(2)/sum(C1_5))*Q0;
x3=(C1_5(3)/sum(C1_5))*Q0;
x4=(C1_5(4)/sum(C1_5))*Q0;
x5=(C1_5(5)/sum(C1_5))*Q0;

% Gradient method first iteration

a= 0.1; %alpha

vector=[x1,x2,x3,x4,x5]'-a.*...
    [T1(x1)-T6(x1,x2,x3,x4,x5),...
    T2(x2)-T6(x1,x2,x3,x4,x5),...
    T3(x3)-T6(x1,x2,x3,x4,x5),...
    T4(x4)-T6(x1,x2,x3,x4,x5),...
    T5(x5)-T6(x1,x2,x3,x4,x5)]';

% Store first iteration and second iteration in matrix P
P_2a=zeros(5,3);
P_2a(:,1)=[x1,x2,x3,x4,x5];
P_2a(:,2)=[vector(1),vector(2),vector(3),vector(4),vector(5)];
i=1;

% Matrix A is the stopping condition
A=(sum(abs(P_2a(:,i+1)-P_2a(:,1))));

% Gradient method iteration while loop
 while A>= 10^(-6)
     x1=vector(1);
     x2=vector(2);
     x3=vector(3);
     x4=vector(4);
     x5=vector(5);

     %Initialize iteration of gradient method
     vector=[x1,x2,x3,x4,x5]'-a.*...
         [T1(x1)-T6(x1,x2,x3,x4,x5),...
         T2(x2)-T6(x1,x2,x3,x4,x5),...
         T3(x3)-T6(x1,x2,x3,x4,x5),...
         T4(x4)-T6(x1,x2,x3,x4,x5),...
         T5(x5)-T6(x1,x2,x3,x4,x5)]';

     % Store each iteration in Matrix P
     P_2a(:,i+2)=[vector(1),vector(2),vector(3),vector(4),vector(5)];

     i=i+1;

     % Update the stopping condition
     A=(sum(abs(P_2a(:,i+1)-P_2a(:,i))));
 end
```

```
% q* table
fprintf("Table of qi* (Q0= %3.2d) below:\n", Q0)
q1_star=P_2a(1,end);
q2_star=P_2a(2,end);
q3_star=P_2a(3,end);
q4_star=P_2a(4,end);
q5_star=P_2a(5,end);
q6_star=Q0-(q1_star+q2_star+q3_star+q4_star+q5_star);
qn=["q1";"q2";"q3";"q4";"q5";"q6"];
qsol=[q1_star;q2_star;q3_star;q4_star;q5_star;q6_star];
qsolT=table(qn,qsol)

% Save each q* from each Q0 to matrix Qgraph
Qgraph(:,j+1)=qsol;

j=j+1;
end

% Plotting the q* with respect to each Q0

fprintf("Graph 2d loading...\n")
Parking_Lots= {"Lower Hearst (q1)","Upper Hearst (q2)","Foothill (q3)",...
    "Bancroft (q4)","Underhill (q5)","Berkeley Way (q6)"}';

% Graph using Qgraph and for loop
for k=1:6
Q0=2000:100:3000;
hold on
plot(Q0,Qgraph(k,:))
end

hold off

% Labelling the graph
title("Plot of q_i* with respect to Q_0 from 2000 to 3000 (step size=100)")
xlabel("Q_0")
ylabel("q_i")
legend(Parking_Lots, 'Location','bestoutside')

fprintf("Done!\n\n")
fprintf("Press enter to continue no 3a: \n");pause;clc
```

Problem 3

a) Solution table:

qn	qsol
q1	62.5030689
q2	33.3838215
q3	251.375067
q4	10.5976404
q5	136.658575
q6	5.48182677

Sn	Ssol
S1	14.7647479
S2	13.7431343
S3	3.78276734
S4	12.7020853
S5	10.6333577
S6	10.6247359

Tn	Tsol
T1	17.7647479
T2	17.7431343
T3	16.7827673
T4	17.7020853
T5	17.6333577
T6	17.6247359

The total parking time for the Q1 driver with parking guidance system is 8165.42

Code:

```matlab
%% 3a

clear all; clc;
% Data
Q1=500;
qsol2a=[838.269075907968;380.270447426218;270.825653427179;...
    350.107894953272;1022.31771989453;138.209208390834];

% My choice of backtracking parameters (stepsize)
a = 1; %alpha
b = 0.8; %beta
c = a;
P_3a=zeros(5,1000);

%S function
S1=@(q1) 2.*exp(0.92.*((q1+qsol2a(1))./600).^1.91);
S2=@(q2) 2.*exp(0.94.*((q2+qsol2a(2))./280).^1.84);
S3=@(q3) 1.*exp(0.16.*((q3+qsol2a(3))./183).^2.02);
S4=@(q4) 1.*exp(0.4.*((q4+qsol2a(4))./232).^4.19);
S5=@(q5) 2.*exp(1.07.*((q5+qsol2a(5))./920).^1.93);
S6=@(q1,q2,q3,q4,q5) 1.*exp(0.51.*((Q1-(q1+q2+q3+q4+q5)+qsol2a(6))...
    ./100).^4.23);



%Objective function
Z=@(q1,q2,q3,q4,q5)...
    (2.*exp(0.92.*((q1+qsol2a(1))./600).^1.91)+3).*q1+...
    (2.*exp(0.94.*((q2+qsol2a(2))./280).^1.84)+4).*q2+...
    (1.*exp(0.16.*((q3+qsol2a(3))./183).^2.02)+13).*q3+...
    (1.*exp(0.4.*((q4+qsol2a(4))./232).^4.19)+5).*q4+...
    (2.*exp(1.07.*((q5+qsol2a(5))./920).^1.93)+7).*q5+...
    (1.*exp(0.51.*(((Q1-(q1+q2+q3+q4+q5)+qsol2a(6)))...
    ./100).^4.23)+7).*(Q1-(q1+q2+q3+q4+q5));



% Function needed for gradient
r1=@(q1) (((0.92*1.91/(600^1.91)).*q1.*(q1+qsol2a(1))^(1.91-1))+1);
r2=@(q2) (((0.94*1.84/(280^1.84)).*q2.*(q2+qsol2a(2))^(1.84-1))+1);
r3=@(q3) (((0.16*2.02/(183^2.02)).*q3.*(q3+qsol2a(3))^(2.02-1))+1);
r4=@(q4) (((0.4*4.19/(232^4.19)).*q4.*(q4+qsol2a(4))^(4.19-1))+1);
r5=@(q5) ((1.07*1.93/(920^1.93)).*q5.*(q5+qsol2a(5))^(1.93-1)+1);
r6=@(q1,q2,q3,q4,q5) (((0.51*4.23/(100^4.23)).*(Q1-(q1+q2+q3+q4+q5)).*...
    ((Q1-(q1+q2+q3+q4+q5))+qsol2a(6))^(4.23-1))+1);

% Start point
C1_6=[600;280;183;232;920];
x1=(C1_6(1)/sum(C1_6))*Q1;
x2=(C1_6(2)/sum(C1_6))*Q1;
x3=(C1_6(3)/sum(C1_6))*Q1;
x4=(C1_6(4)/sum(C1_6))*Q1;
x5=(C1_6(5)/sum(C1_6))*Q1;

% Store the x^(0) into the first column of the matrix
P_3a(:,1)=[x1,x2,x3,x4,x5];
```

```
%Function at x
fk=Z(x1,x2,x3,x4,x5);

%Gradient at x
gk=[...
    (r1(x1).*S1(x1)+3)-(r6(x1,x2,x3,x4,x5)*S6(x1,x2,x3,x4,x5)+7),...
    (r2(x2).*S2(x2)+4)-(r6(x1,x2,x3,x4,x5)*S6(x1,x2,x3,x4,x5)+7),...
    (r3(x3).*S3(x3)+13)-(r6(x1,x2,x3,x4,x5)*S6(x1,x2,x3,x4,x5)+7),...
    (r4(x4).*S4(x4)+5)-(r6(x1,x2,x3,x4,x5)*S6(x1,x2,x3,x4,x5)+7),...
    (r5(x5).*S5(x5)+7)-(r6(x1,x2,x3,x4,x5)*S6(x1,x2,x3,x4,x5)+7)];

xx=[x1,x2,x3,x4,x5];

%x^(1) = x^(0) - t (gradient vector)
vector=[x1,x2,x3,x4,x5]'-c.*gk';

% Function at x of first iteration
fk1=Z(vector(1),vector(2),vector(3),vector(4),vector(5));
i=1;

% Matrix A is the stopping condition
A_3a=(sum(abs(P_3a(:,1+1)-P_3a(:,1))));

while A_3a>=10^-6
    while (fk-(10^-2)*c*gk*gk'< fk1)

          c = c * b;
    vector=xx'-c.*[...
           (r1(x1).*S1(x1)+3)-(r6(x1,x2,x3,x4,x5)*S6(x1,x2,x3,x4,x5)+7),...
           (r2(x2).*S2(x2)+4)-(r6(x1,x2,x3,x4,x5)*S6(x1,x2,x3,x4,x5)+7),...
           (r3(x3).*S3(x3)+13)-(r6(x1,x2,x3,x4,x5)*S6(x1,x2,x3,x4,x5)+7),...
           (r4(x4).*S4(x4)+5)-(r6(x1,x2,x3,x4,x5)*S6(x1,x2,x3,x4,x5)+7),...
           (r5(x5).*S5(x5)+7)-(r6(x1,x2,x3,x4,x5)*S6(x1,x2,x3,x4,x5)+7)]';

    fk1=Z(vector(1),vector(2),vector(3),vector(4),vector(5));


    end


    x1=vector(1);
    x2=vector(2);
    x3=vector(3);
    x4=vector(4);
    x5=vector(5);


   vector=[x1,x2,x3,x4,x5]'-c.*[...
           (r1(x1).*S1(x1)+3)-(r6(x1,x2,x3,x4,x5)*S6(x1,x2,x3,x4,x5)+7),...
           (r2(x2).*S2(x2)+4)-(r6(x1,x2,x3,x4,x5)*S6(x1,x2,x3,x4,x5)+7),...
           (r3(x3).*S3(x3)+13)-(r6(x1,x2,x3,x4,x5)*S6(x1,x2,x3,x4,x5)+7),...
           (r4(x4).*S4(x4)+5)-(r6(x1,x2,x3,x4,x5)*S6(x1,x2,x3,x4,x5)+7),...
           (r5(x5).*S5(x5)+7)-(r6(x1,x2,x3,x4,x5)*S6(x1,x2,x3,x4,x5)+7)]';
```

```
    P_3a(:,i+1)=[vector(1),vector(2),vector(3),vector(4),vector(5)];


    % Update the stopping condition
     A_3a=(sum(abs(P_3a(:,i+1)-P_3a(:,i))));
    i=i+1;



end

fprintf("Table of qi* below:\n")
q1_star3a=P_3a(1,end);
q2_star3a=P_3a(2,end);
q3_star3a=P_3a(3,end);
q4_star3a=P_3a(4,end);
q5_star3a=P_3a(5,end);
q6_star3a=Q1-(q1_star3a+q2_star3a+q3_star3a+q4_star3a+q5_star3a);
qn=["q1";"q2";"q3";"q4";"q5";"q6"];
qsol=[q1_star3a;q2_star3a;q3_star3a;q4_star3a;q5_star3a;q6_star3a];
qsolT=table(qn,qsol)

fprintf("Table of Si* below:\n")
Sn=["S1","S2","S3","S4","S5","S6"]';
Ssol=[S1(q1_star3a),S2(q2_star3a),S3(q3_star3a),S4(q4_star3a),S5(q5_star3a),
...
    S6(q1_star3a,q2_star3a,q3_star3a,q4_star3a,q5_star3a)]';
SsolT=table(Sn,Ssol)

% Ti(qi*) Table
fprintf("Table of Ti* below:\n")

T1=@(q1) 2.*exp(0.92.*((q1+qsol2a(1))./600).^1.91)+3;
T2=@(q2) 2.*exp(0.94.*((q2+qsol2a(2))./280).^1.84)+4;
T3=@(q3) 1.*exp(0.16.*((q3+qsol2a(3))./183).^2.02)+13;
T4=@(q4) 1.*exp(0.4.*((q4+qsol2a(4))./232).^4.19)+5;
T5=@(q5) 2.*exp(1.07.*((q5+qsol2a(5))./920).^1.93)+7;
T6=@(q1,q2,q3,q4,q5) 1.*exp(0.51.*((Q1-
(q1+q2+q3+q4+q5)+qsol2a(6))./100).^4.23)+7;

Tn=["T1","T2","T3","T4","T5","T6"]';
Tsol=[T1(q1_star3a),T2(q2_star3a),T3(q3_star3a),T4(q4_star3a),T5(q5_star3a),
...
    T6(q1_star3a,q2_star3a,q3_star3a,q4_star3a,q5_star3a)]';
TsolT=table(Tn,Tsol)



obj=Z(q1_star3a,q2_star3a,q3_star3a,q4_star3a,q5_star3a);
fprintf("Total parking time for Q1 driver is: %4.2f \n\n",obj)

fprintf("Press enter to continue no 3b: \n");pause;clc
```

b) The total parking time for Q1 drivers without parking guidance system is 8650.61. So, it is preferable to use the parking guidance system since that the total parking time is lower.

Code:

```
%% 3b

clear all;

% My choice of backtracking parameters (stepsize)
a = 0.1; %alpha
P_2a=zeros(5,1000);



% Defining gradient
Q0=3500;

T1=@(q1) 2.*exp(0.92.*(q1./600).^1.91)+3;
T2=@(q2) 2.*exp(0.94.*(q2./280).^1.84)+4;
T3=@(q3) 1.*exp(0.16.*(q3./183).^2.02)+13;
T4=@(q4) 1.*exp(0.4.*(q4./232).^4.19)+5;
T5=@(q5) 2.*exp(1.07.*(q5./920).^1.93)+7;
T6=@(q1,q2,q3,q4,q5) 1.*exp(0.51.*((Q0-(q1+q2+q3+q4+q5))./100).^4.23)+7;

% Start point
C1_5=[600;280;183;232;920];

x1=(C1_5(1)/sum(C1_5))*Q0;
x2=(C1_5(2)/sum(C1_5))*Q0;
x3=(C1_5(3)/sum(C1_5))*Q0;
x4=(C1_5(4)/sum(C1_5))*Q0;
x5=(C1_5(5)/sum(C1_5))*Q0;

% Gradient method first iteration

vector=[x1,x2,x3,x4,x5]'-a.*...
    [T1(x1)-T6(x1,x2,x3,x4,x5),...
    T2(x2)-T6(x1,x2,x3,x4,x5),...
    T3(x3)-T6(x1,x2,x3,x4,x5),...
    T4(x4)-T6(x1,x2,x3,x4,x5),...
    T5(x5)-T6(x1,x2,x3,x4,x5)]';

% Store first iteration and second iteration in matrix P
P_2a=zeros(5,3);
P_2a(:,1)=[x1,x2,x3,x4,x5];
P_2a(:,2)=[vector(1),vector(2),vector(3),vector(4),vector(5)];
i=1;

% Matrix A is the stopping condition
A=(sum(abs(P_2a(:,1+1)-P_2a(:,1))));

% Gradient method iteration while loop
 while A>= 10^(-6)
```

```
    x1=vector(1);
    x2=vector(2);
    x3=vector(3);
    x4=vector(4);
    x5=vector(5);

    %Initialize iteration of gradient method
    vector=[x1,x2,x3,x4,x5]'-a.*...
        [T1(x1)-T6(x1,x2,x3,x4,x5),...
        T2(x2)-T6(x1,x2,x3,x4,x5),...
        T3(x3)-T6(x1,x2,x3,x4,x5),...
        T4(x4)-T6(x1,x2,x3,x4,x5),...
        T5(x5)-T6(x1,x2,x3,x4,x5)]';

    % Store each iteration in Matrix P
    P_2a(:,i+2)=[vector(1),vector(2),vector(3),vector(4),vector(5)];

    i=i+1;

    % Update the stopping condition
    A=(sum(abs(P_2a(:,i+1)-P_2a(:,i))));
 end

% q* table
fprintf("Table of qi* below:\n")
q1_star=P_2a(1,end);
q2_star=P_2a(2,end);
q3_star=P_2a(3,end);
q4_star=P_2a(4,end);
q5_star=P_2a(5,end);
q6_star=Q0-(q1_star+q2_star+q3_star+q4_star+q5_star);
qn=["q1";"q2";"q3";"q4";"q5";"q6"];
qsol=[q1_star;q2_star;q3_star;q4_star;q5_star;q6_star];
qsolT=table(qn,qsol)

qsol2a=[838.269075907968;380.270447426218;270.825653427179;...
    350.107894953272;1022.31771989453;138.209208390834];

%Objective function
Z=@(q1,q2,q3,q4,q5)...
    (2.*exp(0.92.*((q1)./600).^1.91)+3).*(q1-qsol2a(1))+...
    (2.*exp(0.94.*((q2)./280).^1.84)+4).*(q2-qsol2a(2))+...
    (1.*exp(0.16.*((q3)./183).^2.02)+13).*(q3-qsol2a(3))+...
    (1.*exp(0.4.*((q4)./232).^4.19)+5).*(q4-qsol2a(4))+...
    (2.*exp(1.07.*((q5)./920).^1.93)+7).*(q5-qsol2a(5))+...
    (1.*exp(0.51.*(((Q0-(q1+q2+q3+q4+q5)))...
    ./100).^4.23)+7).*(Q0-(q1+q2+q3+q4+q5)-qsol2a(6));


obj3b=Z(q1_star,q2_star,q3_star,q4_star,q5_star);
fprintf("Total parking time is: %4.2f \n\n",obj3b)

fprintf("Lab 2 CIVENG 191 Done!\n")
```