|  |
| --- |
| By Cliff Rodriguez |
| Math Econ HW\_3 |
| Date: September 13, 2018 |

|  |
| --- |
| Cliff Rodriguez |

Contents

[Question 1 2](#_Toc524621143)

[Question 2 2](#_Toc524621144)

[Question 3 4](#_Toc524621145)

[Question 4 6](#_Toc524621146)

[Question 5 12](#_Toc524621147)

[Question 6 14](#_Toc524621148)

[Question 7 16](#_Toc524621149)

[Question 8 21](#_Toc524621150)

# Question 1

Only on paper

# Question 2

MatLab Code

clear;

syms Q;

TC = Q.^3 - 5\*Q.^2 + 60\*Q; % Total Cost

MC = **diff**(TC, Q);% Marginal Cost

AC = Q^2-5\*Q+60; % Average Cost

**disp**('Average Cost');

**pretty**(AC);

AC1 = **diff**(AC,Q);

**disp**('AC first derivative w.r.t. Q');

**pretty**(AC1);

AC2 = **diff**(AC1,Q);

% Second Order Derivative Test

**disp**('Second Order Derivative Test')

if (AC2 < 0)

**disp**('Second Order Derivative Test value is constant and less than zero.')

**disp**('Function is strictly concave and at a global maximum');

**disp**('AC" w.r.t. Q');

**pretty**(AC2);

%AC2 = 'double';

**elseif** (AC2 == 0)

**disp**('Test is inconclusive');

**disp**('AC" w.r.t. Q');

**pretty**(AC2);

%AC2 = 'double';

**elseif** (AC2 > 0)

**disp**('Second Order Derivative Test value is constant and is greater than zero')

**disp**('Function is strictly convex and at a global minimum');

**disp**('AC" w.r.t. Q');

**pretty**(AC2);

%AC2 = 'double';

else

**disp**('further evaluation needed, continue taking high order derivatives or stop now and test is inconclusive')

**disp**('AC" w.r.t. Q');

**pretty**(AC2);

%insert higher order derivative tests

end

% find AC min

eqn1 = AC1 == 0;

ACmin = **solve**(eqn1);

**subs**(AC,Q,ACmin);

double(ans);

Qstar = ans;

**disp**('Q\* =');

**disp**(ACmin);

**disp**('AC(Q\*)');

**disp**(Qstar);

**fplot**(AC,[0, 5], 'b'); % plot line

hold on

**fplot**(TC,[0, 5],'g');

**fplot**(MC,[0, 5], 'black');

% standard lines for each image

**legend**('show','Location','best'); % add legend to graph

% comment out to remove critical points

**plot**(double(ACmin), double(**subs**(AC,ACmin)),'ro'); % plot critical points

% **plot**(double(ACQstar), double(**subs**(AC,ACQstar)),'ro'); % plot critical points

**title**('Optimization Graph');

hold off

Console Output

>> HW3\_2

Average Cost

Q^2 - 5 Q + 60

AC first derivative w.r.t. Q

2 Q - 5

Second Order Derivative Test

Second Order Derivative Test value is constant and is greater than zero

Function is strictly convex and at a global minimum

AC" w.r.t. Q

2

Q\* =

5/2

AC(Q\*)

53.7500

Image



# Question 3

MatLab Code

clear;

syms Q;

P = 12.5\***exp**(-.005\*Q); % Price

TR = (12.5\***exp**(-.005\*Q))\*Q; % Total Revenue

%**disp**('Total Revenue');

%**pretty**(TR);

TR1 = **diff**(TR,Q);

%**disp**('TR first derivative w.r.t. Q');

%**pretty**(TR1);

% start of solving the three equations

eqn1 = TR1 == 0;

Qstar = **solve**(eqn1);

**disp**('Qstar =');

**pretty**(Qstar);

**subs**(TR,Q,Qstar);

double(ans);

TRmax = ans;

**disp**('Total Revenue Max');

**disp**(TRmax);

**fplot**(TR,[0, 4000], 'b'); % plot line

hold on

% standard lines for each image

**legend**('show','Location','best'); % add legend to graph

% comment out to remove critical points

**plot**(double(Qstar), double(**subs**(TR,Qstar)),'ro'); % plot critical points

**title**('Optimization Graph');

Console Output

>> HW3\_3

Qstar =

200

Total Revenue Max

919.6986

Image



# Question 4

MatLab Code

clear;

syms x y lambda;

u = x^.5 \* y^.5; % objective function

c = 200 - 2\*x - 4\*y; % constraint function

% setup Lagrange

L = u+(lambda\*c);

**disp**('Lagrange Function');

**disp**(L);

% Lagrange First order differentials

L1x = **diff**(L,x);

L1y = **diff**(L,y);

L1l = **diff**(L,lambda);

% Lagrange First order differentials

L2xx = **diff**(L1x, x);

L2yy = **diff**(L1y, y);

L2xy = **diff**(L1x, y);

**disp**('Lagrange First Order Conditions');

**disp**('first derivative wrt A');

**pretty**(L1x);

**disp**('first derivative -> C');

**pretty**(L1y);

**disp**('first derivative -> l');

**pretty**(L1l);

**disp**('second derivative -> xx');

**pretty**(L2xx);

**disp**('second derivative -> yy');

**pretty**(L2yy);

**disp**('second derivative -> xy');

**pretty**(L2xy);

% start of solving the three equations

eqn1 = L1x == 0;

eqn2 = L1y == 0;

eqn3 = L1l == 0;

**disp**('Lagrange first order conditions');

**disp**('first derivative -> x = 0');

**pretty**(eqn1);

**disp**('first derivative -> y = 0');

**pretty**(eqn2);

**disp**('first derivative -> l = 0');

**pretty**(eqn3);

sol = **solve**([eqn1, eqn2, eqn3], [x, y, lambda]);

xSol = sol.x;

ySol = sol.y;

lambdaSol = sol.lambda;

**disp**('x\* =');

**disp**(xSol);

**disp**('y\* =');

**disp**(ySol);

**disp**('lambda\* =');

**disp**(lambdaSol);

**subs**(u,x,xSol);

**disp**(ans);

%**subs**(u,x,xSol)

usuby = ans;

clear ans;

**subs**(usuby,y,ySol);

**disp**(ans);

%image for problem

**fsurf**(u, [0 100]);

hold on

**fsurf**(c);

**text**(**X**(0,50),**Y**(0,25),**Z**(50,25),'Peak');

%**plot**(double(xSol), double(**subs**(u,xSol)),'ro'); % plot critical points

hold off;

Console Output

Lagrange Function

x^(1/2)\*y^(1/2) - lambda\*(2\*x + 4\*y - 200)

Lagrange First Order Conditions

first derivative wrt A

y^(1/2)/(2\*x^(1/2)) - 2\*lambda

first derivative w.r.t C

x^(1/2)/(2\*y^(1/2)) - 4\*lambda

first derivative w.r.t l

200 - 4\*y - 2\*x

second derivative w.r.t xx

-y^(1/2)/(4\*x^(3/2))

second derivative w.r.t yy

-x^(1/2)/(4\*y^(3/2))

second derivative w.r.t xy

1/(4\*x^(1/2)\*y^(1/2))

Lagrange first order conditions

first derivative w.r.t x = 0

y^(1/2)/(2\*x^(1/2)) - 2\*lambda == 0

first derivative w.r.t y = 0

x^(1/2)/(2\*y^(1/2)) - 4\*lambda == 0

first derivative w.r.t l = 0

200 - 4\*y - 2\*x == 0

x\* =

50

y\* =

25

lambda\* =

2^(1/2)/8

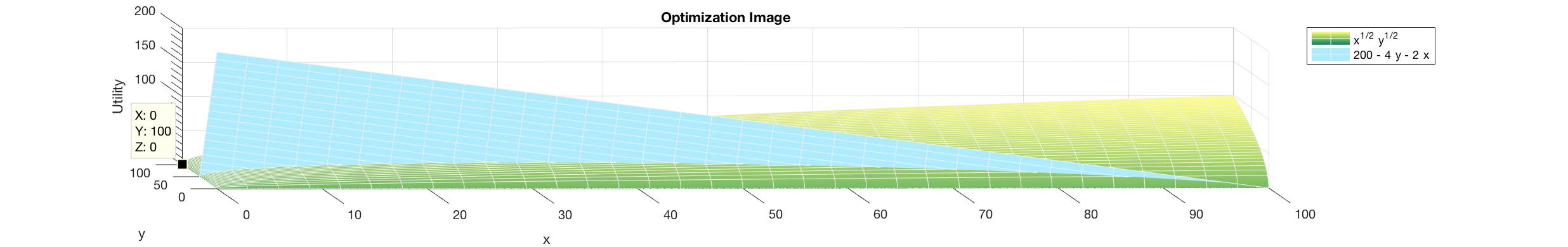
U(x\*,y\*)

5\*50^(1/2)

or

35.3553

Image



# Question 5

MatLab Code

clear;

syms k l lambda;

q = k^.3 \* l^.5; % objective function

c = 384 - 6\*k - 2\*l; % constraint function

**fsurf**(@(k,l) k^.3 \* l^.5,[0 75])

hold on

**fsurf**(c);

hold off

% setup Lagrange

**disp**('objective function');

**disp**(q);

Q = q+(lambda\*c);

**disp**('Lagrange Function');

**disp**('Q = ');

**disp**(Q);

Q1k = **diff**(Q,k);

Q1l = **diff**(Q,l);

Q1lambda = **diff**(Q,lambda);

Q2kk = **diff**(Q1k, k);

Q2ll = **diff**(Q1l, l);

Q2kl = **diff**(Q1k, l);

**disp**('first derivative of Q w.r.t K');

**pretty**(Q1k);

**disp**('first derivative Q w.r.t L');

**pretty**(Q1l);

**disp**('first derivative Q w.r.t lambda');

**pretty**(Q1lambda);

**disp**('second derivative Q w.r.t kk');

**pretty**(Q2kk);

**disp**('second derivative Q w.r.t ll');

**pretty**(Q2ll);

**disp**('second derivative Q w.r.t kl');

**pretty**(Q2kl);

% start of solving the three equations

eqn1 = Q1k == 0;

eqn2 = Q1l == 0;

eqn3 = Q1lambda == 0;

sol = **solve**([eqn1, eqn2, eqn3], [k, l, lambda]);

kSol = sol.k;

lSol = sol.l;

lambdaSol = sol.lambda;

**disp**('k\* =');

**disp**(kSol);

**disp**('l\* =');

**disp**(lSol);

**disp**('lambda\* =');

double(lambdaSol);

lambdaSol = ans;

clear ans;

**disp**(lambdaSol);

**disp**('max output assuming second order conditions are met is:');

**disp**('q(k\*,l\*) = ');

**subs**(q,l,lSol);

maxOutput = ans;

**subs**(maxOutput,k,kSol);

double(ans);

**disp**(ans)

maxOutput = ans;

clear ans;

Console Output

# Question 6

MatLab Code

clear;

syms x y lambda;

p = 80\*x - 2\*x^2 - x \* y - 3\*y^2 + 100\*y; % objective function

c = 12 - x - y; % constraint function

**fsurf**(@(x,y) 80.\*x - 2.\*x^2 - x.\*y - 3.\*y^2 + 100.\*y,[0 12])

hold on

**fsurf**(c);

hold off

% setup Lagrange

P = p+(lambda\*c);

**disp**('Profit equals');

**disp**(P);

P1x = **diff**(P,x);

P1y = **diff**(P,y);

P1l = **diff**(P,lambda);

P2xx = **diff**(P1x, x);

P2yy = **diff**(P1y, y);

P2xy = **diff**(P1x, y);

**disp**('first derivative -> x');

**pretty**(P1x);

**disp**('first derivative -> y');

**pretty**(P1y);

**disp**('first derivative -> l');

**pretty**(P1l);

**disp**('second derivative -> xx');

**pretty**(P2xx);

**disp**('second derivative -> yy');

**pretty**(P2yy);

**disp**('second derivative -> xy');

**pretty**(P2xy);

% start of solving the three equations

eqn1 = P1x == 0;

eqn2 = P1y == 0;

eqn3 = P1l == 0;

sol = **solve**([eqn1, eqn2, eqn3], [x, y, lambda]);

xSol = sol.x;

ySol = sol.y;

lambdaSol = sol.lambda;

**disp**('x =');

**disp**(xSol);

**disp**('y =');

**disp**(ySol);

**disp**('lambda =');

**disp**(lambdaSol);

Console Output

Profit equals

80\*x + 100\*y - lambda\*(x + y - 12) - x\*y - 2\*x^2 - 3\*y^2

first derivative -> x

80 - 4 x - y - lambda

first derivative -> y

100 - x - 6 y - lambda

first derivative -> l

12 - y - x

second derivative -> xx

-4

second derivative -> yy

-6

second derivative -> xy

-1

x =

5

y =

7

lambda =

53

# Question 7

MatLab Code

clear;

syms C A lambda;

p = 2\*C^.5 + A^.5; % objective function

c = 150 - .5\*A - C; % constraint function

**fsurf**(@(C,A) 2\*C^.5 + A^.5,[0 150])

% setup Lagrange

P = p+(lambda\*c);

**disp**('Profit equals');

**disp**(P);

P1A = **diff**(P,A);

P1C = **diff**(P,C);

P1l = **diff**(P,lambda);

P2AA = **diff**(P1A, A);

P2CC = **diff**(P1C, C);

P2AC = **diff**(P1A, C);

**disp**('first derivative -> A');

**disp**(P1A);

**disp**('first derivative -> C');

**disp**(P1C);

**disp**('first derivative -> l');

**disp**(P1l);

**disp**('second derivative -> xx');

**disp**(P2AA);

**disp**('second derivative -> yy');

**disp**(P2CC);

**disp**('second derivative -> xy');

**disp**(P2AC);

% start of solving the three equations

eqn1 = P1A == 0;

eqn2 = P1C == 0;

eqn3 = P1l == 0;

sol = **solve**([eqn1, eqn2, eqn3], [A, C, lambda]);

ASol = sol.A;

CSol = sol.C;

lambdaSol = sol.lambda;

**disp**('A =');

**disp**(ASol);

**disp**('C =');

**disp**(CSol);

**disp**('lambda =');

**disp**(lambdaSol);

part d.

clear;

syms C A lambda;

p = 2\*C^.5 + A^.5; % objective function

c = 300 - .5\*A - C; % constraint function

**fsurf**(@(C,A) 2\*C^.5 + A^.5,[0 150])

% setup Lagrange

P = p+(lambda\*c);

**disp**('Profit equals');

**disp**(P);

P1A = **diff**(P,A);

P1C = **diff**(P,C);

P1l = **diff**(P,lambda);

P2AA = **diff**(P1A, A);

P2CC = **diff**(P1C, C);

P2AC = **diff**(P1A, C);

**disp**('first derivative -> A');

**disp**(P1A);

**disp**('first derivative -> C');

**disp**(P1C);

**disp**('first derivative -> l');

**disp**(P1l);

**disp**('second derivative -> xx');

**disp**(P2AA);

**disp**('second derivative -> yy');

**disp**(P2CC);

**disp**('second derivative -> xy');

**disp**(P2AC);

% start of solving the three equations

eqn1 = P1A == 0;

eqn2 = P1C == 0;

eqn3 = P1l == 0;

sol = **solve**([eqn1, eqn2, eqn3], [A, C, lambda]);

ASol = sol.A;

CSol = sol.C;

lambdaSol = sol.lambda;

**disp**('A =');

**disp**(ASol);

**disp**('C =');

**disp**(CSol);

**disp**('lambda =');

**disp**(lambdaSol);

double(lambdaSol)

lambdasol= ans;

clear ans;

Console Output

Profit equals

A^(1/2) - lambda\*(A/2 + C - 150) + 2\*C^(1/2)

first derivative -> A

1/(2\*A^(1/2)) - lambda/2

first derivative -> C

1/C^(1/2) - lambda

first derivative -> l

150 - C - A/2

second derivative -> xx

-1/(4\*A^(3/2))

second derivative -> yy

-1/(2\*C^(3/2))

second derivative -> xy

0

A =

100

C =

100

lambda =

1/10

Profit equals

A^(1/2) - lambda\*(A/2 + C - 300) + 2\*C^(1/2)

first derivative -> A

1/(2\*A^(1/2)) - lambda/2

first derivative -> C

1/C^(1/2) - lambda

first derivative -> l

300 - C - A/2

second derivative -> xx

-1/(4\*A^(3/2))

second derivative -> yy

-1/(2\*C^(3/2))

second derivative -> xy

0

A =

200

C =

200

lambda =

2^(1/2)/20

ans =

0.0707

# Question 8

MatLab Code

clear;

syms K L lambda;

q = 100\*(.2\*K^.5 + .8\*L^.5)^2; % objective function

c = 4100 - 10\*K - 4\*L; % constraint function

% **fsurf**(@(K,L) 100\*(.2\*x^.5 + .8\*y^.5)^2,[0 410])

% setup Lagrange

Q = q+(lambda\*c);

**disp**('objective function Q =');

**disp**(Q);

Q1k = **diff**(Q,K);

Q1l = **diff**(Q,L);

Q1lambda = **diff**(Q,lambda);

Q2kk = **diff**(Q1k, K);

Q2ll = **diff**(Q1l, L);

Q2kl = **diff**(Q1k, L);

**disp**('first order derivative of Q w.r.t K');

**disp**(Q1k);

**disp**('first order derivative of Q w.r.t L');

**disp**(Q1l);

**disp**('first order derivative of Q w.r.t lambda');

**disp**(Q1lambda);

**disp**('second order derivative of Q w.r.t kk');

**disp**(Q2kk);

**disp**('second order derivative of Q w.r.t ll');

**disp**(Q2ll);

**disp**('second order derivative of Q w.r.t kl');

**disp**(Q2kl);

% start of solving the three equations

eqn1 = Q1k == 0;

eqn2 = Q1l == 0;

eqn3 = Q1lambda == 0;

sol = **solve**([eqn1, eqn2, eqn3], [K, L, lambda]);

kSol = sol.K;

lSol = sol.L;

lambdaSol = sol.lambda;

**disp**('K\* =');

**disp**(kSol);

**disp**('L\* =');

**disp**(lSol);

**disp**('lambda\* =');

**disp**(lambdaSol);

**disp**('max output assuming second order conditions are met is:');

**disp**('q(k\*,l\*) = ');

**subs**(q,L,lSol);

maxOutput = ans;

**subs**(maxOutput,K,kSol);

double(ans);

**disp**(ans)

maxOutput = ans;

clear ans;

Console Output

>> HW3\_8

objective function Q =

100\*(K^(1/2)/5 + (4\*L^(1/2))/5)^2 - lambda\*(10\*K + 4\*L - 4100)

first order derivative of Q w.r.t K

(20\*(K^(1/2)/5 + (4\*L^(1/2))/5))/K^(1/2) - 10\*lambda

first order derivative of Q w.r.t L

(80\*(K^(1/2)/5 + (4\*L^(1/2))/5))/L^(1/2) - 4\*lambda

first order derivative of Q w.r.t lambda

4100 - 4\*L - 10\*K

second order derivative of Q w.r.t kk

2/K - (10\*(K^(1/2)/5 + (4\*L^(1/2))/5))/K^(3/2)

second order derivative of Q w.r.t ll

32/L - (40\*(K^(1/2)/5 + (4\*L^(1/2))/5))/L^(3/2)

second order derivative of Q w.r.t kl

8/(K^(1/2)\*L^(1/2))

K\* =

10

L\* =

1000

Lambda\* =

82/5

max output assuming second order conditions are met is:

q(k\*,l\*) =

67240

>> HW3\_8

objective function

100\*(K^(1/2)/5 + (4\*L^(1/2))/5)^2 - lambda\*(10\*K + 4\*L - 4100)

first order derivative of Q w.r.t K

(20\*(K^(1/2)/5 + (4\*L^(1/2))/5))/K^(1/2) - 10\*lambda

first order derivative of Q w.r.t L

(80\*(K^(1/2)/5 + (4\*L^(1/2))/5))/L^(1/2) - 4\*lambda

first order derivative of Q w.r.t lambda

4100 - 4\*L - 10\*K

second order derivative of Q w.r.t kk

2/K - (10\*(K^(1/2)/5 + (4\*L^(1/2))/5))/K^(3/2)

second order derivative of Q w.r.t ll

32/L - (40\*(K^(1/2)/5 + (4\*L^(1/2))/5))/L^(3/2)

second order derivative of Q w.r.t kl

8/(K^(1/2)\*L^(1/2))

K\* =

10

L\* =

1000

lambda\* =

82/5

max output assuming second order conditions are met is:

q(k\*,l\*) =

67240