

## 1 Problem 1 code

```
1 clear
2 syms x1 x2 f(x1,x2) g(x1,x2)
3
4 f(x1,x2) = x1^4 - 2*x2*x1^2 + x2^2 + x1^2 - 2*x1 + 5;
5 g(x1,x2) = -(x1 + 0.25)^2 + 0.75*x2;
6 dfx1 = diff(f,x1);
7 dfx2 = diff(f,x2);
8 dgx1 = diff(g,x1);
9 dgx2 = diff(g,x2);
10 % Starting point
11 xn = [0.2533, -0.1583];
12 hesslagrange = [1, 0; 0, 1];
13 fn = f(xn(1), xn(2));
14
15 x3 = [-0.3566, -0.0109];
16 hesslagrange = [5.4616, 1.8157; 1.8157, 1.9805];
17 Pprev = 5.85;
18 % Step to calculate the hessian of the lagrange
19 [hl_1, xnew, P] = takeStepSQP(f,g,dfx1,dfx2,dgx1,dgx2,x3,hesslagrange,Pprev
    )
20
21 % First step
22 [hl_1, xnew, P] = takeStepSQP(f,g,dfx1,dfx2,dgx1,dgx2,xnew,hl_1,P)
23
24 % Second Step
25 [hl_1, xnew, P] = takeStepSQP(f,g,dfx1,dfx2,dgx1,dgx2,xnew,hl_1,P)
26
27 function [hl_1, xnew, P] = takeStepSQP(f,g,dfx1,dfx2,dgx1,dgx2,xn,
    hesslagrange,Pprev)
28     syms cx1 cx2 lambda
29     % Determine values of f and grad f at point
30     fn = f(xn(1), xn(2));
31     gradfn = [dfx1(xn(1), xn(2)); dfx2(xn(1), xn(2))];
32
33     % Determine values of g and grad g at point
34     gn = double(g(xn(1),xn(2)));
35     gradgn = double([dgx1(xn(1), xn(2)); dgx2(xn(1), xn(2))]);
36
37     % Make Taylor Series Expansion of function and constraints about point
38     changex = [cx1; cx2];
39     fa = fn + gradfn.'*changex + 1/2*changex.'*hesslagrange*changex;
40     ga = gn + gradgn.'*changex;
41     grad_fa_cx1 = diff(fa,cx1);
42     grad_fa_cx2 = diff(fa,cx2);
43     grad_ga_cx1 = diff(ga,cx1);
44     grad_ga_cx2 = diff(ga,cx2);
45
46     % Solve KKT conditions
47     solution = solve(grad_fa_cx1 - lambda*grad_ga_cx1 == 0, ...
48         grad_fa_cx2 - lambda*grad_ga_cx2 == 0, ...
49         ga == 0);
50
51     cx1 = double(solution.cx1);
```

```

52     cx2 = double(solution.cx2);
53     % If lambda is negative, drop the constraint from the equation
54     lambda = double(solution.lambda);
55
56     % Check to make sure penalty function decreased
57     xnew = [(xn(1) + cx1), (xn(2) + cx2)];
58     fnew = f(xnew(1), xnew(2));
59     gnew = g(xnew(1), xnew(2));
60     % Penalty is function + sum of penalty of violated constraints
61     if gnew < 0
62         P = fnew + lambda*abs(gnew);
63     else
64         P = fnew;
65     end
66     % Check if P is less than Pprev, if not reduce step size until it is.
67     if P >= Pprev
68         check = 0
69     end
70
71     % Update Lagrangian
72     gradfnew = [dfx1(xnew(1), xnew(2)); dfx2(xnew(1), xnew(2))];
73     gradgnew = [dgx1(xnew(1), xnew(2)); dgx2(xnew(1), xnew(2))];
74     hl = hesslagrange;
75
76     % Gamma is difference in grad lagrangians at x0 and x1 with updated
       lambda
77     gradlagr0 = gradfn - lambda*gradgn;
78     gradlagr1 = gradfnew - lambda*gradgnew;
79     gamma = double(gradlagr1 - gradlagr0);
80     cx = [cx1; cx2];
81     hl_1 = double(hl + (gamma*gamma.)/(gamma.*cx) - (hl*cx*cx.)*(hl)/(cx
       .*hl*cx));
82 end
83
84 function [hl_1, xnew, P] = takeStepSQP_noCon(f, dfx1, dfx2, xn, hesslagrange,
       Pprev)
85     syms cx1 cx2 lambda
86     % Determine values of f and grad f at point
87     fn = f(xn(1), xn(2));
88     gradfn = [dfx1(xn(1), xn(2)); dfx2(xn(1), xn(2))];
89
90     % Make Taylor Series Expansion of function and constraints about point
91     changex = [cx1; cx2];
92     fa = fn + gradfn.*changex + 1/2*changex.*hesslagrange*changex;
93     grad_fa_cx1 = diff(fa, cx1);
94     grad_fa_cx2 = diff(fa, cx2);
95
96     % Solve KKT conditions
97     solution = solve(grad_fa_cx1 == 0, ...
98         grad_fa_cx2 == 0);
99
100     cx1 = double(solution.cx1);
101     cx2 = double(solution.cx2);
102
103     % Check to make sure penalty function decreased
104     xnew = [(xn(1) + cx1), (xn(2) + cx2)];

```

```

105     fnew = f(xnew(1),xnew(2));
106     P = fnew;
107     % Check if P is less than Pprev, if not reduce step size until it is.
108     if P >= Pprev
109         cx1 = cx1*0.5;
110         cx2 = cx2*0.5;
111         % Check to make sure penalty function decreased
112         xnew = [(xn(1) + cx1), (xn(2) + cx2)];
113         fnew = f(xnew(1),xnew(2));
114         P = fnew
115     end
116     % Update Lagrangian
117     gradfnew = [dfx1(xnew(1), xnew(2)); dfx2(xnew(1), xnew(2))];
118     hl = hesslagrange;
119
120     % Gamma is difference in grad lagrangians at x0 and x1 with updated
        lambda
121     gradlagr0 = gradfn;
122     gradlagr1 = gradfnew;
123     gamma = double(gradlagr1 - gradlagr0);
124     cx = [cx1; cx2];
125     hl_1 = double(hl + (gamma*gamma.')./(gamma.*cx) - (hl*cx*cx.'*hl)/(cx
        .*hl*cx));
126 end

```

## 2 Problem 2 code

```

1  clear
2  syms x1 x2 f(x1,x2) g(x1,x2) lambda
3
4  f(x1,x2) = x1^4 - 2*x2*x1^2 + x2^2 + x1^2 - 2*x1 + 5;
5  g(x1,x2) = -(x1 + 0.25)^2 + 0.75*x2;
6  dfx1 = diff(f,x1);
7  dfx2 = diff(f,x2);
8  dgx1 = diff(g,x1);
9  dgx2 = diff(g,x2);
10
11  x1 = [-1.695,2.157];
12  xn = x1;
13  % Determine values of f and grad f at point
14  fn1 = f(xn(1), xn(2));
15  gradfn1 = [dfx1(xn(1), xn(2)); dfx2(xn(1), xn(2))];
16
17  % Determine values of g and grad g at point
18  gn1 = double(g(xn(1),xn(2)));
19  gradgn1 = double([dgx1(xn(1), xn(2)); dgx2(xn(1), xn(2))]);
20
21  x2 = [-0.592,-1.162];
22  xn = x2;
23  % Determine values of f and grad f at point
24  fn2 = f(xn(1), xn(2));
25  gradfn2 = [dfx1(xn(1), xn(2)); dfx2(xn(1), xn(2))];
26
27  % Determine values of g and grad g at point
28  gn2 = double(g(xn(1),xn(2)));
29  gradgn2 = double([dgx1(xn(1), xn(2)); dgx2(xn(1), xn(2))]);

```

```

30 h1 = [20.762,5.629;5.629,1.910];
31 lambda = 0;
32
33 cx1_prev = 1.104;
34 cx2_prev = -3.319;
35 % Find update hessian
36 gradlagr1 = gradfn1 - lambda*gradgn1;
37 gradlagr2 = gradfn2 - lambda*gradgn2;
38 gamma = double(gradlagr2 - gradlagr1);
39 cx = [cx1_prev; cx2_prev];
40 h1_1 = double(h1 + (gamma*gamma.')./(gamma.*cx) - (h1*cx*cx.'*h1)/(cx.*h1*
    cx));
41 s = 0.230;
42 syms cx1 cx2 clambda cs
43 A = [18.5612, 5.1258, 0, -0.6840; 5.1258, 2.1849, 0, -0.75; 0, 0, 0, 0.23;
    0.6840, 0.750, -1, 0];
44 b = [-6.7655; -3.0249; -0.2; -1.2180];
45 x = mldivide(A,b)
46 xnew = [(x2(1) + x(1)), (x2(2) + x(2))];
47 f3 = double(f(xnew(1),xnew(2)))
48 g3 = double(g(xnew(1),xnew(2)))

```