Example of Sequential QP method

Anh-Huy Phan

min f(x) = 1/2*((x(1)-x(3))^2 + (x(2)-x(4))^2)

subject to c(x) >=0

c(x) = [c1(x); c2(x)]

c1 = -[x(1) x(2)] * [1/4 0; 0 1] * [x(1); x(2)] + [x(1) x(2)]*[1/2 ; 0] + 3/4

c2 = -1/8* [x(3) x(4)] * [5 3 ; 3 5] * [x(3); x(4)] + [x(3) x(4)]*[11/2 ; 13/2] - 35/2

```
addpath('C:\Users\huyph\Documents\MATLAB\slp_sqp\slp_sqp')
addpath(genpath('C:\Users\huyph\Documents\MATLAB\cvx-w64\'))
```

```
clear all;clc

% define symbolic primal variable and dual variable

x = sym('x',[4 1]);
lambda = sym('lambda',[2,1]);

%  min f(x)
% subject to  c(x) >= 0

f = 1/2*((x(1)-x(3))^2 + (x(2)-x(4))^2);

c1 = -[x(1) x(2)] * [1/4 0; 0 1] * [x(1); x(2)] + [x(1) x(2)]*[1/2 ; 0] + 3/4; % >=0
c2 = -1/8* [x(3) x(4)] * [5 3 ; 3 5] * [x(3); x(4)] + [x(3) x(4)]*[11/2 ; 13/2] -
35/2; % >=0

c = [c1; c2];
% Langangian
L = f - lambda.' * c;

Hf = hessian(f);
Hc1 = hessian(c1,x);
Hc2 = hessian(c2,x);

gf = gradient(f);
gc1 = gradient(c1,x);
gc2 = gradient(c2,x);
Gc = [gc1 gc2];
```

```
HLxx = Hf-lambda(1)*Hc1-lambda(2)*Hc2; % hessian of the Lagrangian w.t. to x
```

Solve the problem using SQP toolbox

```
x0 = [1 0.5 2 3]';

problem = struct();
problem.objective = @mobj;
problem.x0 = x0;
problem.nonlcon = @mconstr;
%problem.options.HessFun=@Hessian;
[x_sqp,opts,v,H,status]=sqp( problem);

mobj(x_sqp)
```

```
ans = 1.4583
```

```
fprintf('Optimal solution x* = [%.4f, %.4f, %.4f, %.4f], f(x*) = %.4f
\n',x_sqp,mobj(x_sqp))
```

```
Optimal solution x* = [2.0447, 0.8527, 2.5449, 2.4856], f(x*) = 1.4583
```

Initialization

```
x0 = [1; 0.5; 2; 3];
x_curr = x0;
```

Main algorithm

Step 1: Hessian = identity matrix

From Step 2, ... Hessian can be updated or explicitly computed

```
for krun = 1:10
    gfx = double(subs(gf,x,x_curr));
    gcx = double(subs(Gc,x,x_curr));
    cx =  double(subs(c,x,x_curr));

    % Update Hessian
    if krun == 1
        H = eye(4); % Hessian
    else
        fullHessian = 1; % full Hessian or use the BFGS method
        if fullHessian
            H = double(subs(HLxx,[x;lambda],[x_curr; lambda_])); % Hessian
        else
            gLx_old = gfx_old - gcx_old*lambda_;
            gLx = gfx - gcx*lambda_;
            gamma = gLx-gLx_old;% difference of gradient
            dx = x_curr - x_old;
            if all(dx==0)
```

```
            H = eye(4);
        else
            w = H*dx;
            H = H + gamma*gamma'/(gamma'*dx) - (w*w')/(dx'*w);
        end

    end
end

% penalty function
fx = double(subs(f,x,x_curr));
if krun == 1
    phi = fx;
else
    phi = fx + lambda_'*(abs(cx) .* (cx<0));
end

% QP approximates the original problem
% min    1/2 * d'*H*d + gf'*d
% s.t.   cx + gcx'*d ==0
OPTIONS = optimoptions('quadprog','Display','off');
[dx,fval,~,output,lambda_] = quadprog(H,gfx,-gcx',cx,[],[],[],[],[],OPTIONS);
lambda_ = lambda_.ineqlin;

xnew = x_curr + dx;
% penalty function
fxnew = double(subs(f,x,xnew));
cxnew =  double(subs(c,x,xnew));

phinew = fxnew + lambda_'*(abs(cxnew) .* (cxnew<0));
if phinew <= phi
    x_old = x_curr;
    gfx_old = gfx;
    gcx_old = gcx;
    x_curr = xnew;
    fx = fxnew;

else
    alpha = fminsearch(@(alpha) meritfunction(alpha,f,c,x,x_curr,dx,lambda_),0);
    xnew = x_curr + alpha*dx;
    % penalty function
    fxnew = double(subs(f,x,xnew));
    cxnew =  double(subs(c,x,xnew));

    phinew = fxnew + lambda_'*(abs(cxnew) .* (cxnew<0));
    x_old = x_curr;
    gfx_old = gfx;
    gcx_old = gcx;
    x_curr = xnew;
    fx = fxnew;
```

```
    end

    fprintf('d = [%s] |',sprintf('%.4f ',dx)),
    fprintf('x = [%s] |',sprintf('%.4f ',x_curr)),
    fprintf('lambda %s | ',sprintf('%.4f ',lambda_)),
    fprintf('f(x) = %2.4f | phi = %.4f \n',fx,phinew)

    fx_(krun) = fx;
    % check stopping condition
end
```

```
d = [1.0000 0.7500 0.2353 -0.4412 ] |
x = [2.0000 1.2500 2.2353 2.5588 ] |
lambda 1.7500 1.6471  |
f(x) = 0.8842 | phi = 2.4352
d = [-0.0549 -0.3140 0.1967 -0.0470 ] |
x = [1.9451 0.9360 2.4320 2.5118 ] |
lambda 1.0700 1.0597  |
f(x) = 1.3602 | phi = 1.4863
d = [0.0960 -0.0773 0.1119 -0.0298 ] |
x = [2.0410 0.8587 2.5439 2.4820 ] |
lambda 0.9555 1.0889  |
f(x) = 1.4440 | phi = 1.4583
d = [0.0038 -0.0060 0.0011 0.0036 ] |
x = [2.0448 0.8527 2.5449 2.4856 ] |
lambda 0.9574 1.1001  |
f(x) = 1.4582 | phi = 1.4583
d = [-0.0000 -0.0000 -0.0000 0.0000 ] |
x = [2.0447 0.8527 2.5449 2.4856 ] |
lambda 0.9575 1.1001  |
f(x) = 1.4583 | phi = 1.4583
d = [-0.0000 -0.0000 -0.0000 0.0000 ] |
x = [2.0447 0.8527 2.5449 2.4856 ] |
lambda 0.9575 1.1001  |
f(x) = 1.4583 | phi = 1.4583
d = [-0.0000 -0.0000 -0.0000 0.0000 ] |
x = [2.0447 0.8527 2.5449 2.4856 ] |
lambda 0.9575 1.1001  |
f(x) = 1.4583 | phi = 1.4583
d = [-0.0000 -0.0000 -0.0000 0.0000 ] |
x = [2.0447 0.8527 2.5449 2.4856 ] |
lambda 0.9575 1.1001  |
f(x) = 1.4583 | phi = 1.4583
d = [-0.0000 -0.0000 -0.0000 0.0000 ] |
x = [2.0447 0.8527 2.5449 2.4856 ] |
lambda 0.9575 1.1001  |
f(x) = 1.4583 | phi = 1.4583
d = [-0.0000 -0.0000 -0.0000 0.0000 ] |
x = [2.0447 0.8527 2.5449 2.4856 ] |
lambda 0.9575 1.1001  |
f(x) = 1.4583 | phi = 1.4583
```

```
function  [f,g] = mobj(u)
% f : objective function
% g: gradient of f wrt u
x1 = u(1);x2 = u(2);x3 = u(3); x4 = u(4);
f =1/2*((x1-x3)^2 + (x2-x4)^2);
```

```
g =[x1 - x3
    x2 - x4
    x3 - x1
    x4 - x2];
end

function  [cieq,ceq,gieq,geq] = mconstr(u)
% cieq: inequality constraint function
% ceq:  equality constraint function
% gieq: gradient of cieq
% geq: gradient of ceq
%
x1 = u(1);x2 = u(2);x3 = u(3); x4 = u(4);
ceq = [];

cieq = [[x1 x2] * [1/4 0; 0 1] * [x1; x2] - [x1 x2]*[1/2 ; 0] - 3/4
    1/8* [x3 x4] * [5 3 ; 3 5] * [x3; x4] - [x3 x4]*[11/2 ; 13/2] + 35/2];

gieq  = [x1/2 - 1/2   0
    2*x2                 0
    0          (5*x3)/4 + (3*x4)/4 - 11/2
    0          (3*x3)/4 + (5*x4)/4 - 13/2];
geq = [];
end


function H = Hessian(u,lambda)
x = u(1);y = u(2); z = u(3);t = u(4);
lambda1 = lambda(1);
lambda2 = lambda(2);
H = [lambda1/2 + 1,             0,                -1,                 0
    0, 2*lambda1 + 1,                 0,                -1
    -1,             0, (5*lambda2)/4 + 1,     (3*lambda2)/4
    0,            -1,     (3*lambda2)/4, (5*lambda2)/4 + 1];
end

function phi = meritfunction(alpha,f,c,x,x_curr,dx,lambda_)
xnew = x_curr + alpha*dx;
% penalty function
fxnew = double(subs(f,x,xnew));
cxnew = double(subs(c,x,xnew));

phi = fxnew + lambda_'*(abs(cxnew) .* (cxnew<0));
end
```