Assignment #7

Shao Hua, Huang 0750727
ECM5901 - Optimization Theory and Application

January 14, 2020

Note that I use MATLAB to complete the following two exercises.

Exercise 1. (Equality constrained entropy maximization.) Consider the equality constrained entropy maximization problem

$$\begin{array}{ll} \text{minimize} & f(x) = \sum_{i=1}^{5} x_i \log x_i \\ \text{subject to} & Ax = b \end{array}$$

where $A = \begin{bmatrix} 4 & 3 & 2 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$ and $b = \begin{bmatrix} 20 \\ 15 \end{bmatrix}$ with $\mathbf{dom}\, f = \mathbb{R}^n_{++}$.
Compute the solution of the problem using the following methods.

(a) Standard Newton method (Algorithm 10.1) with initial point $x^{(0)} = [1\ 2\ 3\ 4\ 5]^T$. (30%)

(b) Infeasible start Newton method (Algorithm 10.2) with initial point $\nu^{(0)} = 0$, $x^{(0)} = [1\ 2\ 3\ 4\ 5]^T$, and also $x^{(0)} = [5\ 2\ 3\ 4\ 5]^T$. (30%)

Verify that the two methods compute the same optimal point. Note that $\mathbf{dom}\, f$ is not $\mathbb{R}^5$ and thus in the update step of x, you have to check that $x + t\Delta x \in \mathbf{dom}\, f$.

Solution. Shared parameters

```
% file: hw7_1.m
A = [4 3 2 1 0; 1 1 1 1 1];
b = [20 15]';
[m, n] = size(A);
maxiters = 50;
alpha = 0.01;
beta = 0.5;
nttol = 1e-7;
```

The code and the result of (a) are shown in the next page.

```
% (a) Feasible start Newton method
x = [1 2 3 4 5]';
disp('(a) Feasible start Newton method');
disp('    Iterate optimal points and values:');
disp(['           x(0) = [ ', sprintf('%f ', x), ']']);
disp(['      optval(0) = ', num2str(sum(x.*log(x)), '%f')]);
for i = 1:maxiters
    g = log(x) + 1;
    H = diag(1 ./ x);
    KKTleft = [H A'; A zeros(m)];
    KKTright = [-g; zeros(m, 1)];
    KKTv = KKTleft \ KKTright;
    xnt = KKTv(1:n);
    fprime = g' * xnt;
    if ((-fprime / 2) < nttol)
        break;
    end
    t = 1;
    val = sum(x .* log(x));
    while true
        newx = x + t * xnt;
        if (min(newx) >= 0)
            newval = sum((newx) .* log(newx));
            if (newval < val + alpha * t * fprime)
                break;
            end
        end
        t = beta * t;
    end
    x = x + t * xnt;
    disp(['           x(', int2str(i), ') = [ ', sprintf('%f ', x), ']']);
    disp(['      optval(', int2str(i), ') = ', num2str(sum(x.*log(x)), '%f')]);
end
```

```
(a) Feasible start Newton method
Iterate optimal points and values:
       x(0) = [ 1.000000 2.000000 3.000000 4.000000 5.000000 ]
  optval(0) = 18.274498
       x(1) = [ 1.295984 1.897411 2.667326 3.789180 5.350099 ]
  optval(1) = 18.188621
       x(2) = [ 1.319344 1.873759 2.661133 3.779082 5.366682 ]
  optval(2) = 18.188216
```

Code of (b) with changing x to required initial feasible point $(1, 2, 3, 4, 5)$ and infeasible point $(5, 2, 3, 4, 5)$.

```
% (b) Infeasible start Newton method with (1, 2, 3, 4, 5)
x = [1 2 3 4 5]';
nu = zeros(m, 1);
disp('(b) Infeasible start Newton method with x = (1, 2, 3, 4,
   5)');
disp('    Iterate optimal points and values:');
disp(['            x(0) = [ ', sprintf('%f ', x), ']']);
disp(['       optval(0) = ', num2str(sum(x.*log(x)), '%f')]);
for i = 1:maxiters
    g = log(x) + 1;
    H = diag(1 ./ x);
    r = [g + A' * nu; A * x - b];
    KKTleft = [H A'; A zeros(m)];
    KKTright = -r;
    KKTv = KKTleft \ KKTright;
    xnt = KKTv(1:n);
    nunt = KKTv(n+1:n+m);
    t = 1;
    while true
        newx = x + t * xnt;
        newnu = nu + t * nunt;
        if (min(newx) >= 0)
            newr = [(log(newx) + 1) + A' * (newnu); A * (newx)
               - b];
            if norm(newr) < (1 - alpha) * norm(r)
                break;
            end
        end
        t = beta * t;
    end
    x = x + t * xnt;
    nu = nu + t * nunt;
    disp(['            x(', int2str(i), ') = [ ', sprintf('%f ',
       x), ']']);
    disp(['       optval(', int2str(i), ') = ', num2str(sum(x.*
      log(x)), '%f')]);
    if (norm(newr) < nttol)
        break;
    end
end
```

The result of $x^{(0)} = (1, 2, 3, 4, 5)$:

```
(b) Infeasible start Newton method with x = (1, 2, 3, 4, 5)
    Iterate optimal points and values:
           x(0) = [ 1.000000 2.000000 3.000000 4.000000 5.000000 ]
      optval(0) = 18.274498
           x(1) = [ 1.295984 1.897411 2.667326 3.789180 5.350099 ]
      optval(1) = 18.188621
           x(2) = [ 1.319344 1.873759 2.661133 3.779082 5.366682 ]
      optval(2) = 18.188216
           x(3) = [ 1.319414 1.873761 2.661015 3.779032 5.366779 ]
      optval(3) = 18.188216
```

The result of $x^{(0)} = (5, 2, 3, 4, 5)$:

```
(b) Infeasible start Newton method with x = (5, 2, 3, 4, 5)
    Iterate optimal points and values:
           x(0) = [ 5.000000 2.000000 3.000000 4.000000 5.000000 ]
      optval(0) = 26.321688
           x(1) = [ 0.571943 2.489808 3.160990 3.920824 4.856435 ]
      optval(1) = 18.621276
           x(2) = [ 1.128127 2.021496 2.793087 3.836830 5.220460 ]
      optval(2) = 18.214160
           x(3) = [ 1.312610 1.878115 2.666233 3.782747 5.360294 ]
      optval(3) = 18.188250
           x(4) = [ 1.319406 1.873767 2.661020 3.779036 5.366772 ]
      optval(4) = 18.188216
           x(5) = [ 1.319414 1.873761 2.661015 3.779032 5.366779 ]
      optval(5) = 18.188216
```

Exercise 2. You were asked to prove that $x^* = (1, 1/2, -1)$ is optimal for the following optimization problem in HW#4:

$$\begin{array}{ll} \text{minimize} & f_0(x) = (1/2)x^T P x + q^T x + r \\ \text{subject to} & -1 \le x_i \le 1, \; i = 1,2,3 \end{array}$$

where

$$P = \begin{bmatrix} 13 & 12 & -2 \\ 12 & 17 & 6 \\ -2 & 6 & 12 \end{bmatrix}, \; q = \begin{bmatrix} -22 \\ -14.5 \\ 13 \end{bmatrix}, \; r = 1.$$

Implement a barrier method for solving this QP. Assume that the initial point is $x^{(0)} = 0$. Plot the duality gap versus Newton steps (such as Fig. 11.4). Verify that the barrier method computes the optimal point. (40%)

Solution. With readily derivation, we transform the inequality constraints to $Ax \preceq b$ where

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Therefore, the corresponding central path problem is

$$\text{minimize} \quad t f_0(x) + \phi(x)$$

where $\phi(x) = -\sum_{i=1}^{6} \log(b_i - a_i^T x)$ with $a_1^T, \ldots, a_6^T$ are rows of A

Also, we know that

$$\nabla f_0(x) = Px + q, \quad \nabla^2 f_0(x) = P$$

$$\nabla \phi(x) = \sum_{i=1}^{6} \frac{a_i}{b_i - a_i^T x} = A^T d \quad \nabla^2 \phi(x) = \sum_{i=1}^{6} \frac{a_i a_i^T}{(b_i - a_i^T x)^2} = A^T \,\mathbf{diag}(d)^2 A$$

where $d \in \mathbb{R}^6$, $d_i = 1/(b_i - a_i^T x)$.

The MATLAB code and the result are shown below.

```
% file: hw7_2.m

P = [13 12 -2; 12 17 6; -2 6 12];
q = [-22 -14.5 13]';
r = 1;
A = [1 0 0; 0 1 0; 0 0 1; -1 0 0; 0 -1 0; 0 0 -1];
b = [1 1 1 1 1 1]';
```

```
[m, n] = size(A);
maxiters = 200;
alpha = 0.01;
beta = 0.5;
nttol = 1e-6;
tol = 1e-3;
mu = 20;
t = 1;
x = [0 0 0]';
inniters = [];
gaps = [];
for i = 1:maxiters
    d = b - A * x;
    val = t * (.5 * x' * P * x + q' * x + r) - sum(log(d));
    g = t * (P * x + q) + A' * (1 ./ d);
    H = t * P + A' * diag(1 ./ d .^ 2) * A;
    xnt = -H \ g;
    fprime = g' * xnt;
    s = 1;
    dd = -A * xnt;
    while (min(d + s * dd) <= 0)
        s = beta * s;
    end
    while (t * (.5 * (x + s * xnt)' * P * (x + s * xnt) + q' *
       (x + s * xnt) + r) - sum(log(d + s * dd)) >= val + alpha
        * s * fprime)
        s = beta * s;
    end
    x = x + s * xnt;
    if ((-fprime / 2) < nttol)
        gap = m / t;
        inniters = [inniters, i];
        gaps = [gaps gap];
        if (gap < tol)
            break;
        end
        t = mu * t;
    end
    disp(['x = [ ', sprintf('%f ', x), '], val = ', num2str
       (((1/2) * x' * P * x + q' * x + r), '%f')]);
end
inniters = [inniters, i];
gaps = [gaps gap];
figure(1)
iters1 = [];
```

```
gaps1 = [];
for i = 1:length(gaps)-1
    iters1 = [iters1 inniters(i)-1 inniters(i+1)-1];
    gaps1 = [gaps1 gaps(i) gaps(i)];
end;
iters1 = [iters1 inniters(length(gaps))-1];
gaps1 = [gaps1 gaps(length(gaps))];
semilogy(iters1, gaps1);
axis([0 40 1e-6 1e2]);
xlabel('Newton iterations');  ylabel('duality gap');
```

```
>> run hw7_2.m
...
...
x = [ 0.999887 0.500048 -0.999938 ], val = -21.624762
x = [ 0.999876 0.500055 -0.999938 ], val = -21.624751
x = [ 0.999875 0.500056 -0.999938 ], val = -21.624750
```

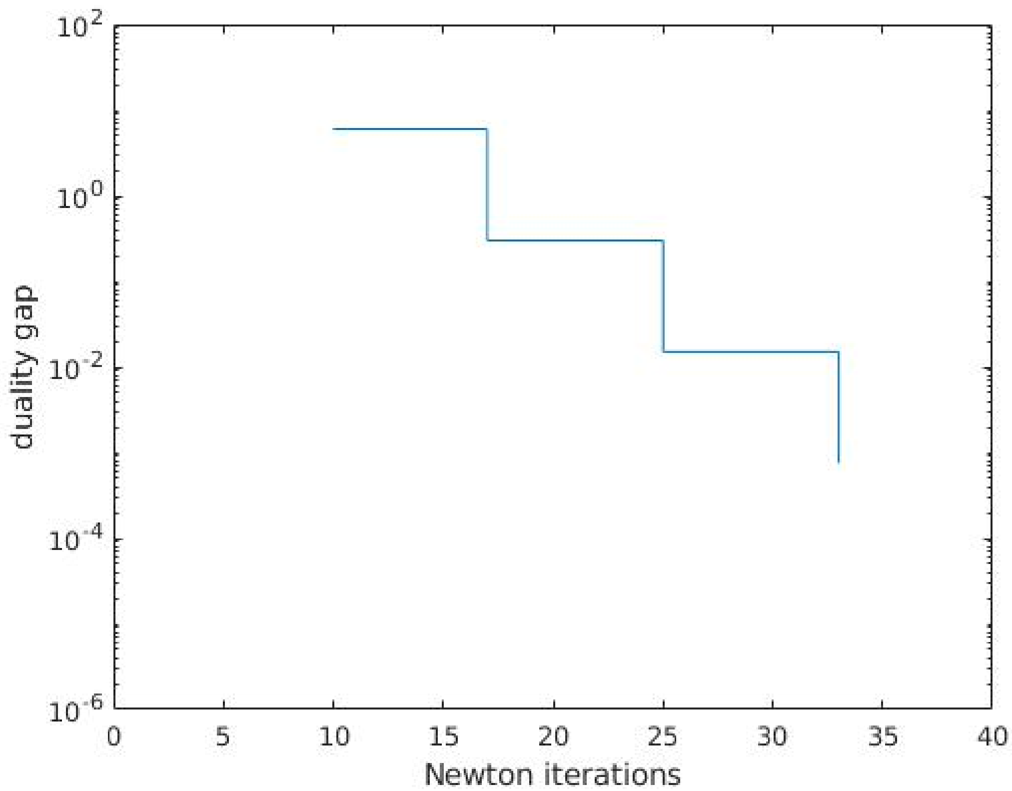

Figure 1: Relation between Newton iterations and duality gap when $\mu = 20$