MATH520 Homework 3

Fernando

February 20, 2024

Exercise 10.1

Part 1: The algorithm is well defined

It is enough to prove that $d^{(i)T}Qd^{(i)} \neq 0$ for all i. Since $Q \succ 0$ it is enough to show that $d^{(i)} \neq 0$. We proceed by induction. For the base case we have: $d^{(0)} = p^{(0)}$ which is clearly not 0. For the induction step if we had:

$$0 = d^{(k+1)} = \left(p^{(k+1)} - \sum_{i=0}^{k} \frac{p^{(k+1)T}Qd^{(i)}}{d^{(i)T}Qd^{(i)}} d^{(i)} \right)$$

Then that implies

$$p^{(k+1)} = \sum_{i=0}^{k} \frac{p^{(k+1)T}Qd^{(i)}}{d^{(i)T}Qd^{(i)}} d^{(i)},$$

or in other words $p^{(k+1)} \in \operatorname{span}(d^{(0)}, \ldots, d^{(k)})$, but notice that by construction $d^{(i)} \in \operatorname{span}(p^{(0)}, \ldots, p^{(i)})$, then $p^{(k+1)} \in \operatorname{span}(p^{(0)}, \ldots, p^{(k)})$ which is a contradiction!

conclusion: $d^{(k+1)} \neq 0$, and that finishes the proof.

Part 2: $d^{(0)}, \ldots, d^{(n-1)}$ are Q-conjugate

We can use induction over k. For $k = 1$ we have

$$d^{(1)} = p^{(1)} - \frac{p^{(1)T}Qd^{(0)}}{d^{(0)T}Qd^{(0)}} d^{(0)},$$

and then:

$$\begin{aligned} d^{(0)T}Qd^{(1)} &= d^{(0)T}Q \left(p^{(1)} - \frac{p^{(1)T}Qd^{(0)}}{d^{(0)T}Qd^{(0)}} d^{(0)} \right) \\ &= d^{(0)T}Qp^{(1)} - p^{(1)T}Qd^{(0)} \text{ (by I.H.)} \\ &= 0. \end{aligned}$$

Now assuming the result is true for k we prove it for $k+1$. For $j = 1, \dots, k$ we have:

$$\begin{aligned} d^{(i)T} Q d^{(k+1)} &= d^{(j)T} Q \left(p^{(k+1)} - \sum_{i=0}^{k} \frac{p^{(k+1)T} Q d^{(i)}}{d^{(i)T} Q d^{(i)}} d^{(i)} \right) \\ &= d^{(j)T} Q p^{(k+1)} - p^{(k+1)T} Q d^{(j)} \text{ (by I.H.)} \\ &= 0. \end{aligned}$$

Exercise 10.7

$$\begin{aligned} \phi(a) &= \frac{1}{2}(x_0 + Da)^T Q (x_0 + Da) - (x_0 + Da)^T b \\ &= \frac{1}{2} x_0^T Q x_0 + \frac{1}{2} x_0^T Q D a + \frac{1}{2}(Da)^T Q x_0 + \frac{1}{2}(Da)^T Q D a - x_0^T b - (Da)^T b \\ &= \frac{1}{2} a^T D^T Q x_0 + \frac{1}{2} a^T D^T Q D a - a^T D^T b + \frac{1}{2} x_0^T Q D a + \frac{1}{2} x_0^T Q x_0 - x_0^T b \\ &= \frac{1}{2} a^T D^T Q D a - a^T D^T b + \frac{1}{2} a^T D^T Q x_0 + \frac{1}{2} a^T D^T Q x_0 + \frac{1}{2} x_0^T Q x_0 - x_0^T b \\ &= \frac{1}{2} a^T D^T Q D a - a^T D^T b + a^T D^T Q x_0 + \frac{1}{2} x_0^T Q x_0 - x_0^T b \\ &= \frac{1}{2} a^T \tilde{Q} a^T + a^T \tilde{b} + \tilde{c} \end{aligned}$$

Where: $\tilde{Q} = D^T Q D$, $\tilde{b} = D^T Q x_0 - D^T b$, $\tilde{c} = \frac{1}{2} x_0^T Q x_0 - x_0^T b$.

Now we need to prove that $\tilde{Q}$ is positive definite (notice that it is symmetric). For this if we take $x \in \mathbb{R}^r$ we get that

$$x^T D^T Q D x = (Dx)^T Q (Dx) \geq 0.$$

So $\tilde{Q}$ is definitely positive semi-definite. For $\tilde{Q}$ to be positive definite we need to ensure that $Dx \neq 0$. Which is true if $r \leq n$ but false if $r > n$ (rank-nullity theorem).

Exercise 10.10

Part a

By comparing terms we get:

$$f(x) = \frac{1}{2} x^T \begin{bmatrix} 5 & 2 \\ 2 & 1 \end{bmatrix} x - x^T \begin{bmatrix} 3 \\ 1 \end{bmatrix}.$$

Part b

First attempt

$$
\begin{aligned}
g^{(0)} =& \nabla f([0,0]^T) = -[3,1]^T \\
d^{(0)} =& [3,1]^T \\
\alpha_0 =& -\frac{g^{(0)T} d^{(0)}}{d^{(0)T} Q d^{(0)}} = \frac{10}{58} \\
x^{(1)} =& \frac{10}{58} \cdot [3,1]^T = [30/58, 10/58]^T \\
g^{(1)} =& \nabla f(x^{(1)}) = \frac{2}{29}[-1,3]^T \\
\beta_0 =& \frac{g^{(1)T} Q d^{(0)}}{d^{(0)T} Q d^{(0)}} = \frac{8/29}{58} \\
d^{(1)} =& -[-2/29, 6/29]^T + \frac{8/29}{58} \cdot [3,1]^T \\
=& [70/841, -170/841]^T \\
\alpha_1 =& -\frac{g^{(1)T} d^{(1)}}{d^{(1)T} Q d^{(1)}} = \frac{-40/841}{200/24389} = \frac{29}{5} \\
x^{(2)} =& [30/58, 10/58]^T - \frac{29}{5}[70/841, -170/841]^T \\
=& [1/29, 39/29] \\
g^{(2)} =& [-4/29, 12/29]
\end{aligned}
$$

There is a mistake in the calculation but I can't find it, this computation is too tedious.

Second attempt

We can solve it with the following python code:

```
import numpy as np

Q=np.array([[5,2],[2,1]])
def grad(vector):
    return np.matmul(Q, vector)-np.array([[3],[1]])

def cgm():
    xk=np.array([[0],[0]])
    gk=grad(xk)
    print(gk)
    dk=-1*gk
    tol=1e-15
    maxiter=100
```

```
    for k in range(maxiter):
        print("iter " + str(k))
        ak=-1*(np.matmul(np.transpose(gk),dk))/\
        (np.matmul(np.matmul(np.transpose(dk),Q),dk))
        print("ak")
        print(ak)
        xk1=xk+ak*dk
        print("xk1")
        print(xk1)
        gk1=grad(xk1)
        print("gk1")
        print(gk1)
        if(np.linalg.norm(gk1)<tol):
            print("Found it!")
            print(xk1)
            return
        bk=\
        (np.matmul(np.matmul(np.transpose(gk1),Q),dk))/\
        (np.matmul(np.matmul(np.transpose(dk),Q),dk))
        print("bk")
        print(bk)
        dk1=-1*gk1+bk*dk
        print("dk1")
        print(dk1)
        # update stuff
        xk=xk1
        gk=gk1
        dk=dk1
cgm()
```

Which produces the following output:

```
[[-3]
 [-1]]
iter 0
ak
[[0.17241379]]
xk1
[[0.51724138]
 [0.17241379]]
gk1
[[-0.06896552]
 [ 0.20689655]]
bk
[[0.00475624]]
dk1
[[ 0.08323424]
```

```
 [-0.20214031]]
iter 1
ak
[[5.8]]
xk1
[[ 1.]
 [-1.]]
gk1
[[8.8817842e-16]
 [4.4408921e-16]]
Found it!
[[ 1.]
 [-1.]]
```

So the vector is $[1, -1]^T$ which is found on the second iteration as expected.

Part c

$$\nabla f(x) = \begin{bmatrix} 5 & 2 \\ 2 & 1 \end{bmatrix} x - \begin{bmatrix} 3 \\ 1 \end{bmatrix},$$

then

$$x = \begin{bmatrix} 1 \\ -1 \end{bmatrix}.$$

Which coincides with the previous part.