Solutions to Part B of Problem Sheet 6

Solution (6.5) From the first block of rows we have

$$\boldsymbol{A}^\top \Delta \boldsymbol{y} + \Delta \boldsymbol{s} = \boldsymbol{0} \iff \Delta \boldsymbol{s} = -\boldsymbol{A}^\top \Delta \boldsymbol{y}.$$

From the second block of rows we have

$$\boldsymbol{A}\Delta \boldsymbol{x} = \boldsymbol{0}.$$

Putting these two together, we get

$$\langle \Delta \boldsymbol{x}, \Delta \boldsymbol{s}\rangle = \langle \Delta \boldsymbol{x}, -\boldsymbol{A}^\top \Delta \boldsymbol{y}\rangle = -\langle \boldsymbol{A}\Delta \boldsymbol{x}, \Delta \boldsymbol{y}\rangle = 0,$$

which shows the claim.

Solution (6.6) The last block of rows reads as

$$\boldsymbol{S}\Delta \boldsymbol{x} + \boldsymbol{X}\Delta \boldsymbol{s} = -\boldsymbol{X}\boldsymbol{S}\boldsymbol{e} + \sigma\mu \boldsymbol{e},$$

so that multiplying by $\boldsymbol{X}^{-1}$ (the diagonal matrix $\boldsymbol{X}$ is non-singular, since we are in $\mathcal{F}^\circ$) and solving for $\Delta \boldsymbol{s}$, we get

$$\Delta \boldsymbol{s} = -\boldsymbol{S}\boldsymbol{e} - \boldsymbol{X}^{-1}\boldsymbol{S}\Delta \boldsymbol{x} + \sigma\mu \boldsymbol{X}^{-1}\boldsymbol{e}.$$

Substituting $\Delta \boldsymbol{s}$ into the first block of rows, we get

$$\boldsymbol{A}^\top \Delta \boldsymbol{y} + \boldsymbol{X}^{-1}\boldsymbol{S}\Delta \boldsymbol{x} = \boldsymbol{S}\boldsymbol{e} - \sigma\mu \boldsymbol{X}^{-1}\boldsymbol{e}.$$

Using $\boldsymbol{D} = \boldsymbol{S}^{-1/2}\boldsymbol{X}^{1/2}$, we get the new system

$$\begin{pmatrix} \boldsymbol{0} & \boldsymbol{A} \\ \boldsymbol{A}^\top & -\boldsymbol{D}^{-2} \end{pmatrix} \begin{pmatrix} \Delta \boldsymbol{y} \\ \Delta \boldsymbol{x} \end{pmatrix} = \begin{pmatrix} \boldsymbol{0} \\ \boldsymbol{s} - \sigma\mu \boldsymbol{X}^{-1}\boldsymbol{e} \end{pmatrix} \tag{1}$$

$$\Delta \boldsymbol{s} = -\boldsymbol{S}\boldsymbol{e} - \boldsymbol{X}^{-1}\boldsymbol{S}\Delta \boldsymbol{x} + \sigma\mu \boldsymbol{X}^{-1}\boldsymbol{e}. \tag{2}$$

Multiplying $\boldsymbol{A}\boldsymbol{D}^2$ to the second block of rows, we get

$$\boldsymbol{A}\boldsymbol{D}^2\boldsymbol{A}^\top \Delta \boldsymbol{y} - \boldsymbol{A}\Delta \boldsymbol{x} = \boldsymbol{A}\boldsymbol{D}^2\boldsymbol{s} - \sigma\mu \boldsymbol{A}\boldsymbol{D}^2\boldsymbol{X}^{-1}\boldsymbol{e},$$

which in view of $\boldsymbol{A}\Delta \boldsymbol{x} = \boldsymbol{0}$, $\boldsymbol{D}^2\boldsymbol{X}^{-1} = \boldsymbol{S}^{-1}$ and $\boldsymbol{D}^2\boldsymbol{s} = \boldsymbol{x}$ simplifies to

$$(\boldsymbol{A}\boldsymbol{D}^2\boldsymbol{A}^\top)\Delta \boldsymbol{y} = \boldsymbol{b} - \sigma\mu \boldsymbol{A}\boldsymbol{S}^{-1}\boldsymbol{e}.$$

This gives a system of equations for recovering $\Delta \boldsymbol{y}$, with a symmetric coefficient matrix $\boldsymbol{A}\boldsymbol{D}^2\boldsymbol{A}^\top$. From the second block of (1) we also get

$$\boldsymbol{X}^{-1}\boldsymbol{S}\Delta \boldsymbol{x} - \sigma\mu \boldsymbol{X}^{-1}\boldsymbol{e} = \boldsymbol{A}^\top \Delta \boldsymbol{y} - \boldsymbol{s},$$

and substituting this into the expression for $\Delta \boldsymbol{s}$ in (2) above gives

$$\Delta \boldsymbol{s} = -\boldsymbol{A}^\top \Delta \boldsymbol{y}.$$

Finally, we can obtain $\Delta \boldsymbol{x}$ from $\Delta \boldsymbol{s}$ via (2),

$$\Delta \boldsymbol{x} = -\boldsymbol{x} - \boldsymbol{S}^{-1}\boldsymbol{X}\Delta \boldsymbol{s} + \sigma\mu \boldsymbol{S}^{-1}\boldsymbol{e}.$$

This is the last of the three equations. The benefit is that one only has to solve one system of equations with symmetric coefficient matrix $\boldsymbol{A}\boldsymbol{D}^2\boldsymbol{A}^\top$, which can be done efficiently using, for example, Cholesky factorization or other methods. Once $\Delta \boldsymbol{y}$ is found, one can compute the other parts $\Delta \boldsymbol{x}$ and $\Delta \boldsymbol{s}$ easily.

Solution (6.7)

(a) We first write down the matrix $\boldsymbol{A}$:

$$\boldsymbol{A} = \begin{pmatrix} 0 & 0.2 & 0.4 & 0.6 & 0.8 & 1 & 1.2 & 1.4 & 1.6 & 1.8 & 2 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

and the vectors $\boldsymbol{b}$ and $\boldsymbol{c}$:

$$\boldsymbol{b} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \ \boldsymbol{c} = \begin{pmatrix} 1 & 1.01 & 1.04 & 1.09 & 1.16 & 1.25 & 1.36 & 1.49 & 1.64 & 1.81 & 2 \end{pmatrix}^\top$$

The primal version of this problem is

$$\begin{aligned} \text{minimize} \quad & x_1 + 1.01x_2 + 1.04x_3 + 1.09x_4 + 1.16x_5 + 1.25x_6 + 1.36x_7 \\ & + 1.49x_8 + 1.64x_9 + 1.81x_{10} + 2x_{11} \\ \text{subject to} \quad & 0.2x_2 + 0.4x_3 + 0.6x_4 + 0.8x_5 + x_6 + 1.2x_7 + 1.4x_8 + 1.6x_9 \\ & + 1.8x_{10} + 2x_{11} = 1 \\ & x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 + x_9 + x_{10} + x_{11} = 1 \\ & x_i \geq 0. \end{aligned}$$

(b) The problem has $m = 2$ dual variables y_1 and y_2, so the projection of the trajectory on the $\boldsymbol{y}$ plane can be easily visualized. A naive implementation is shown below. The trajectories are shown in the figures.

In [1]:

```
import numpy as np
import numpy.linalg as la

v = np.linspace(0,1,11)
n = len(v)
A = np.concatenate((2*v.reshape((1,n)), np.ones((1,n))), axis=0)
c = 1+v**2
b = np.array([1,1])
```

Define function F and Jacobian matrix M.

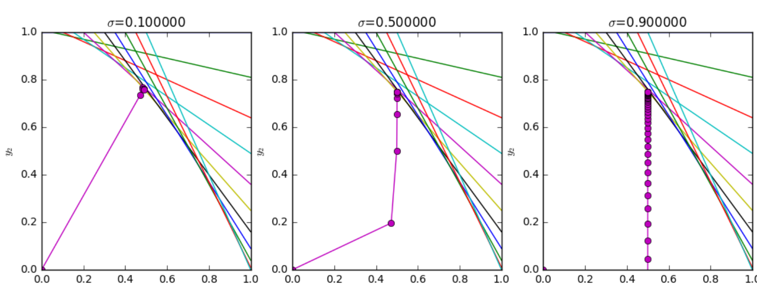
In [2]:

```
def F(x, y, s):
    C1 = np.dot(A.T,y)+s-c
    C2 = np.dot(A,x)-b
    C3 = x*s
    return np.concatenate((C1, C2, C3))

def M(x, y, s):
    return np.asarray(np.bmat([[np.zeros((n,n)), A.T, np.eye(n)],
                    [A, np.zeros((2,2)), np.zeros((2,n))],
                    [np.diag(s), np.zeros((n,2)), np.diag(x)]]))
```

In [3]:
```
x = np.ones(n)/11.
y = np.array([0,0])
s = c-np.dot(A.T, y)
```

In [4]:
```
def longstep(x, y, s, sigma, gamma=1e-3, tol=1e-4):
    mu = 1
    i = 1
    yy = np.zeros((2,50))
    while mu>tol and i<50:
        a = 1
        mu = np.dot(x,s)/11.
        rhs = F(x,y,s)-np.concatenate((np.zeros(n+2), sigma*mu*np.ones(11)))
        delta = -la.solve(M(x,y,s), rhs)
        xs = np.concatenate((x,s))
        deltaxs = np.concatenate((delta[:11], delta[13:]))

        I = np.argmin(xs+deltaxs)
        m = xs[I]+deltaxs[I]
        if m<gamma*mu:
            a = np.amin(-xs[I]/deltaxs[I])

        x = x+a*delta[:11]
        y = y+a*delta[11:13]
        s = s+a*delta[13:]

        yy[:,i] = y
        i+=1
    return yy[:,:i]
```

In [5]:
```
import matplotlib.pyplot as plt
% matplotlib inline
fig, ax = plt.subplots(1,3, figsize=(12, 4))
xx = np.linspace(0,1,100)
sigmas = [0.1, 0.5, 0.9]

for k in range(3):
    yy = longstep(x,y,s,sigmas[k])
    ax[k].set_ylim([0,1])
    for j in range(n):
        ax[k].plot(xx,c[j]-np.dot(A[0,j],xx))
    ax[k].plot(yy[0,:], yy[1,:], ’-o’)
    ax[k].set_title("$\sigma$={:f}".format(sigmas[k]))
    ax[k].set_xlabel(’$y_1$’)
    ax[k].set_ylabel(’$y_2$’)
plt.show()
```



(c) In the figure, the central path is shown as the vertical line in the $\boldsymbol{y}$ plane. The code is exactly the same as above, but instead of recording the trajectory in the y variables, we use the $x_1 s_1$ against the $x_5 s_5$ axis.