Home assignment 3

Numerical Optimization and its Applications - Spring 2019
Gil Ben Shalom, 301908877
Tom Yaacov, 305578239

June 1, 2019

1 Equality constrained optimization

(a) The optimization problem that is given can be formulated to a single equality constrained optimization problem:

$$\max_{\mathbf{x}\in\mathbb{R}^3} x_1x_2 + x_2x_3 + x_1x_3 \quad s.t. \quad x_1 + x_2 + x_3 - 3 = 0$$

The Lagrangian of this method is

$$\mathcal{L}(\mathbf{x}, \lambda_1) = x_1x_2 + x_2x_3 + x_1x_3 + \lambda_1(x_1 + x_2 + x_3 - 3)$$

and the solution of this problem is given by

$$\nabla\mathcal{L} = 0 \Rightarrow \begin{cases} x_2 + x_3 + \lambda_1 = 0 \\ x_1 + x_3 + \lambda_1 = 0 \\ x_1 + x_2 + \lambda_1 = 0 \\ x_1 + x_2 + x_3 = 0 \end{cases} \qquad \Rightarrow \qquad \begin{cases} x_1 = 1 \\ x_2 = 1 \\ x_3 = 1 \\ \lambda_1 = -2 \end{cases}$$

(b) In order to show that this critical point is a maximum point we'll first compute the Hessian of $\mathcal{L}$:

$$\nabla^2_{\mathbf{x}}\mathcal{L}(\mathbf{x}, \lambda_1) = \begin{bmatrix} \frac{\partial^2\mathcal{L}}{\partial x_1^2} & \frac{\partial^2\mathcal{L}}{\partial x_1\partial x_2} & \frac{\partial^2\mathcal{L}}{\partial x_1\partial x_3} \\ \frac{\partial^2\mathcal{L}}{\partial x_2\partial x_1} & \frac{\partial^2\mathcal{L}}{\partial x_2^2} & \frac{\partial^2\mathcal{L}}{\partial x_2\partial x_3} \\ \frac{\partial^2\mathcal{L}}{\partial x_3\partial x_1} & \frac{\partial^2\mathcal{L}}{\partial x_3\partial x_2} & \frac{\partial^2\mathcal{L}}{\partial x_3^2} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$

Next we'll show that the Hessian of the Lagrangian is negative

$$\mathbf{y}^T\nabla^2_{\mathbf{x}}\mathbf{y} \quad \forall\mathbf{y}\in\mathbb{R}^3 \quad s.t. \quad \mathbf{y}^T\mathbf{1} = 0, \mathbf{y} \neq \mathbf{0}$$

$$\begin{aligned} \mathbf{y}^T\nabla^2_{\mathbf{x}}\mathbf{y} &= \begin{bmatrix} y_1 & y_2 & y_3 \end{bmatrix} \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} \\ &= \begin{bmatrix} y_2 + y_3 & y_1 + y_3 & y_1 + y_2 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} \end{aligned}$$

in addition,

$$\mathbf{y}^T\mathbf{1} = 0 \Rightarrow y_1 + y_2 + y_3 = 0 \Rightarrow \begin{bmatrix} y_2 + y_3 \\ y_1 + y_3 \\ y_1 + y_2 \end{bmatrix} = \begin{bmatrix} -y_1 \\ -y_2 \\ -y_3 \end{bmatrix}$$

thererfore, we get

$$\begin{aligned} \mathbf{y}^T \nabla^2_{\mathbf{x}} \mathbf{y} &= \begin{bmatrix} y_2 + y_3 & y_1 + y_3 & y_1 + y_2 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} \\ &= \begin{bmatrix} -y_1 & -y_2 & -y_3 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} \\ &= -y_1^2 - y_2^2 - y_3^2 < 0 \qquad\qquad * \, \mathbf{y} \neq \mathbf{0} \end{aligned}$$

2 General constrained optimization

(a)

(b)

(c)

3 Box-constrained optimization

(a) The scalar box constrained minimization problem can be formulated to general constrained optimiziation problem:

$$\min_{x \in \mathbb{R}} \frac{1}{2} h x^2 - g x \quad s.t. \quad \begin{cases} -x + a \leq 0 \\ x - b \leq 0 \end{cases}$$

The Lagrangian of this method is

$$\mathcal{L}(x, \lambda_1, \lambda_2) = \frac{1}{2} h x^2 - g x + \lambda_1(-x + a) + \lambda_2(x - b)$$

We'll compute the first order necessary conditions: Suppose that x^* is a local solution of the problem, than the following conditions hold:

(1) $\nabla_x \mathcal{L}(x^*, \lambda_1^*, \lambda_2^*) = h x^* - g - \lambda_1^* + \lambda_2^* = 0$

(2) $-x^* + a \leq 0$

(3) $x^* - b \leq 0$

(4) $\lambda_1^* \geq 0$

(5) $\lambda_2^* \geq 0$

(6) $\lambda_1^*(-x^*+a)=0$

(7) $\lambda_2^*(x^*-b)=0$

To know whether a stationary point is a minimum or a maximum we have the second order necessary conditions for general constrained minimization:

$$\nabla_x^2 \mathcal{L}(x^*,\lambda_1^*,\lambda_2^*)=h>0$$

therefore

$$y\nabla_x^2 \mathcal{L}(x^*,\lambda_1^*,\lambda_2^*)y=yhy>0 \quad \forall y\neq 0$$

and the stationary point is a minimum.
We can divide our solution to 3 cases:

(a) $a\leq \frac{g}{h}\leq b$:
Solution: $x^*=\frac{g}{h},\lambda_1^*=0,\lambda_2^*=0$ we'll check that the first order necessary conditions hold:

(1) $\nabla_x \mathcal{L}(x^*,\lambda_1^*,\lambda_2^*)=hx^*-g-\lambda_1^*+\lambda_2^*=h\frac{g}{h}-g-0+0=0$
(2) $-x^*+a=-\frac{g}{h}+a\leq 0$
(3) $x^*-b=\frac{g}{h}-b\leq 0$
(4) $\lambda_1^*=0\geq 0$
(5) $\lambda_2^*=0\geq 0$
(6) $\lambda_1^*(-x^*+a)=0(-\frac{g}{h}+a)=0$
(7) $\lambda_2^*(x^*-b)=0(\frac{g}{h}-b)=0$

(b) $a>\frac{g}{h}$:
Solution: $x^*=a,\lambda_1^*=ha-g,\lambda_2^*=0$ we'll check that the first order necessary conditions hold:

(1) $\nabla_x \mathcal{L}(x^*,\lambda_1^*,\lambda_2^*)=hx^*-g-\lambda_1^*+\lambda_2^*=ha-g-(ha-g)+0=0$
(2) $-x^*+a=-a+a\leq 0$
(3) $x^*-b=a-b\leq 0$
(4) $\lambda_1^*=ha-g>h\frac{g}{h}-g=0$
(5) $\lambda_2^*=0\geq 0$
(6) $\lambda_1^*(-x^*+a)=(ha-g)(-a+a)=0$
(7) $\lambda_2^*(x^*-b)=0(a-b)=0$

(c) $\frac{g}{h}>b$:
Solution: $x^*=b,\lambda_1^*=0,\lambda_2^*=-hb+g$ we'll check that the first order necessary conditions hold:

(1) $\nabla_x \mathcal{L}(x^*,\lambda_1^*,\lambda_2^*)=hx^*-g-\lambda_1^*+\lambda_2^*=hb-g-0-hb+g=0$
(2) $-x^*+a=-b+a\leq 0$
(3) $x^*-b=b-b\leq 0$
(4) $\lambda_1^*=0\geq 0$
(5) $\lambda_2^*=-hb+g>-h\frac{g}{h}+g=0$
(6) $\lambda_1^*(-x^*+a)=0(-b+a)=0$

(7) $\lambda_2^*(x^* - b) = (-hb + g)(b - b) = 0$

(b) Let the following problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} \frac{1}{2}\mathbf{x}^T H\mathbf{x} - \mathbf{x}^T\mathbf{g} \quad s.t. \quad \mathbf{a} \leq \mathbf{x} \leq \mathbf{b}$$

Therefore, the minimization for each scalar x_i is given by

$$\begin{aligned} argmin_{x_i\in\mathbb{R}} \frac{1}{2}\mathbf{x}^T H\mathbf{x} - \mathbf{x}^T\mathbf{g} &= argmin_{x_i\in\mathbb{R}} \frac{1}{2}\Big(\sum_i\sum_j x_i x_j h_{i,j}\Big) - \sum_i x_i g_i \\ &= argmin_{x_i\in\mathbb{R}} \frac{1}{2}x_i^2 h_{i,i} + \frac{1}{2}\Big(\sum_{j\neq i} x_i x_j h_{i,j} + x_i x_j h_{j,i}\Big) - x_i g_i \quad * \text{removing cons} \\ &= argmin_{x_i\in\mathbb{R}} \frac{1}{2}x_i^2 h_{i,i} + \frac{1}{2}2\Big(\sum_{j\neq i} x_i x_j h_{i,j}\Big) - x_i g_i \quad * H \text{ symmetric} \\ &= argmin_{x_i\in\mathbb{R}} \frac{1}{2}x_i^2 h_{i,i} + \Big(\Big(\sum_{j\neq i} x_j h_{i,j}\Big) - g_i\Big)x_i \end{aligned}$$

In addition, given that the rest are known:

$$\mathbf{a} \leq \mathbf{x} \leq \mathbf{b} \Rightarrow a_i \leq x_i \leq b_i$$

therefore, we get that the minimization for each scalar x_i is given by

$$\min_{x_i\in\mathbb{R}} \frac{1}{2}x_i^2 h_{i,i} + \Big(\Big(\sum_{i\neq j} x_j h_{i,j}\Big) - g_i\Big)x_i \quad s.t. \quad a_i \leq x_i \leq b_i$$

In order to show the expression for the update of projected coordinate descent, we need to define the projection operation with respect to some norm and then set

$$x_i^{(k+1)} = \Pi_\Omega(x_i^{(k)} - \alpha\nabla f(x_i^{(k)}))$$

We will choose the squared l_2 norm. The lagrangian is given by

$$\mathcal{L}(x_i, \lambda_1, \lambda_2) = \frac{1}{2}||x_i - y_i||_2^2 + \lambda_1(-x + a) + \lambda_2(x - b)$$

and its gradient is given by

$$\nabla_{x_i}\mathcal{L}(x_i, \lambda_1, \lambda_2) = x_i - y_i - \lambda_1 + \lambda_2$$

The problem is separable, so if $a_i \leq x_i \leq b_i$, then we can set $x_i^* = y_i$ without breaking the constraint, and hence $\lambda_1^* = \lambda_2^* = 0$, because the constraints are inactive. If $y_i < a_i$, then the lower bound constraint is active and the upper bound is not. We set $x_i^* = a_i$ and

$$x_i^* - y_i - \lambda_1 = 0 \Rightarrow \lambda_1 = a_i - y_i > 0$$

We get a positive Lagrange multiplier, which is what needs to be. If $y_i > b_i$, then the upper bound constraint is active and the lower bound is not. We set $x_i^* = b_i$ and

$$x_i^* - y_i + \lambda_2 = 0 \Rightarrow \lambda_2 = y_i - b_i > 0$$

The gradient is defined by

$$(\nabla f(x^{(k)}))_{(i)} = h_{i,i}x_i + \sum_{j \neq i} h_{i,j}x_j - g_i$$

and the step

$$z_i = x_i^{(k)} - \alpha(\nabla f(x^{(k)}))_{(i)}$$

Overall, the projected steepest descent step is given by:

$$x_i^{(k+1)} = \begin{cases} a_i & z_i < a_i \\ b_i & z_i > b_i \\ z_i & otherwise \end{cases}$$

(c) Following is an implementation for projected coordinate descent algorithm

```
from copy import copy
from numpy.linalg import norm
from numpy import matmul


def projected_coordinate_descent(H, g, a, b, x_0, alpha, max_iter, epsilon):
    x = copy(x_0)
    for _ in range(max_iter):
        prev_x = copy(x)
        grad = matmul(H, x) - g
        z = x - alpha * grad
        for i in range(x.shape[0]):
            if z[i] < a[i]:
                x[i] = a[i]
            elif z[i] > b[i]:
                x[i] = b[i]
            else:
                x[i] = z[i]
        if norm(x - prev_x) < epsilon:
            break
    return x


def objective(H, x, g):
    return 0.5 * matmul(matmul(x.T, H), x) - matmul(x.T, g)
```

(d) Running the implementation over the given parameters

```
from numpy import array
from py_files.part_3_c import projected_coordinate_descent, objective
from numpy.random import uniform
```

```python
# parameters setting
H = array([[5, -1, -1, -1, -1],
           [-1, 5, -1, -1, -1],
           [-1, -1, 5, -1, -1],
           [-1, -1, -1, 5, -1],
           [-1, -1, -1, -1, 5]])
g = array([18, 6, -12, -6, 18])
a = array([0, 0, 0, 0, 0])
b = array([5, 5, 5, 5, 5])

# algorithm parameters setting
epsilon = 1e-4
alpha = 0.1
max_iter = 1000
x_0 = uniform(a, b, a.shape[0])

# running the algorithm
x = projected_coordinate_descent(H, g, a, b, x_0, alpha, max_iter, epsilon)
print(x, r"\\")
print(objective(H, x, g), r"\\")
```

[5. 3.66676844 0.66676844 1.66676844 5.]
-111.999999953

4 Projected Gradient Descent for the LASSO regression

(a)

(b)

(c)

(d) (non-mandatory)

(e) (non-mandatory)