Technical University of Crete

School of Electrical and Computer Engineering

Course: **Optimization**

Exercise 4 (100/1000)

Angelopoulos Dimitris, 2020030038

In this exercise we will study the topic of Linear Programming. We will solve problems of the following form.

$$\begin{aligned} \text{minimize} \quad & f(\mathbf{x}) = \mathbf{c}^T\mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

Where $\mathbf{A} \in \mathbb{R}^{p \times n}$ with $\mathbf{rank}(\mathbf{A}) = p$ and $\mathbf{b} \in \mathbb{R}^p$. To create feasible problems we generate random $\mathbf{A}$, random non-negative $\mathbf{x}$ and compute $\mathbf{b} = \mathbf{A}\mathbf{x}$.

1. First of all we solve the problem using CVX.

2. Now we will use the interior point method using the logarithmic barrier starting from a feasible point. Before solving the problem we must check whether the problem is feasible or not. To achieve that we solve a Phase-1 optimization problem of the following form :

   $$\begin{aligned} \text{minimize} \quad & s \\ \text{s.t.} \quad & -x_i \leq s, \quad i = 1 \ldots n \\ & \mathbf{A}\mathbf{x} = \mathbf{b} \end{aligned}$$

   If there exists an $\mathbf{x}$ such that $s < 0$ then the linear problem is feasible and can be solved. After finding a problem that is feasible and bounded we use the interior method algorithm as follows.

Algorithm 1 Barrier Method for Convex Optimization Problems

$\mathbf{x} \in \mathbf{dom}\ f \cap \mathbf{dom}\ \phi$, $\mathbf{Ax} = \mathbf{b}$, $t = 1$, $\mu > 1$, $\epsilon > 0$

repeat

1. Compute $\mathbf{x}_*(t)$, by minimizing $tf + \phi$ subject to $\mathbf{Ax} = \mathbf{b}$, starting at $\mathbf{x}$
2. $\mathbf{x} := \mathbf{x}_*(t)$
3. quit if $\frac{m}{t} < \epsilon$
4. $t := \mu t$

Where ϕ is the barrier function.

$$\phi(\mathbf{x}) = -\sum_{i=1}^{n} \log(x_i)$$

Running the algorithm for $n = 2$ and $p = 1$ we plot the optimal solution, the algorithm trajectory and the feasible set to get the figure below.

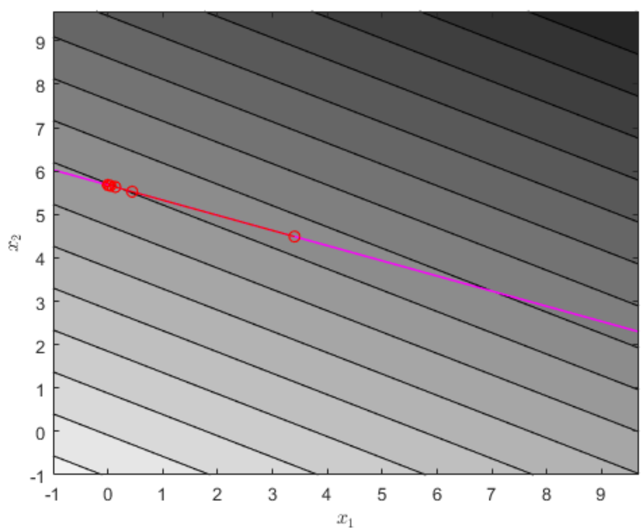

Fig. 1

3. Last but not least we will solve the initial problem using the primal-dual algorithm. The primal dual algorithm goes as follows.

Algorithm 2 Primal-Dual Algorithm

$\mathbf{x} \in \mathbf{dom}\ f \cap \mathbf{dom}\ \phi$, $\boldsymbol{\lambda} > \mathbf{0}$, $t = 1$, $\mu > 1$, $\epsilon_{feas} > 0$, $\epsilon > 0$

repeat

1. $t := \mu \frac{m}{\hat{\eta}}$
2. Compute $\Delta \mathbf{y}_{pd}$
3. Perform Backtracking to choose step s
4. $\mathbf{y} := \mathbf{y} + s\Delta \mathbf{y}_{pd}$

until $(\|r_p\|_2 < \epsilon_{feas}, \|r_d\|_2 < \epsilon_{feas}, \hat{\eta} < \epsilon)$

Where $\hat{\eta}$ is the surrogate duality gap as follows.

$$\hat{\eta}(\mathbf{x}, \boldsymbol{\lambda}) = \mathbf{x}^T \boldsymbol{\lambda}$$

For the second step, after some calculations, we have to solve the following linear system of equations.

$$\begin{bmatrix} \mathbf{O}_{n\times n} & \mathbb{I}_{n\times n} & \mathbf{A}^T \\ \mathbf{diag}(\boldsymbol{\lambda}) & \mathbf{diag}(\mathbf{x}) & \mathbf{O}_{n\times p} \\ \mathbf{A} & \mathbf{O}_{p\times n} & \mathbf{O}_{p\times p} \end{bmatrix} \begin{bmatrix} \Delta \mathbf{x}_{pd} \\ \Delta \boldsymbol{\lambda}_{pd} \\ \Delta \mathbf{v}_{pd} \end{bmatrix} = - \begin{bmatrix} r_{t,d}(\mathbf{y}) \\ r_{t,c}(\mathbf{y}) \\ r_{t,p}(\mathbf{y}) \end{bmatrix}$$

With search direction $\Delta \mathbf{y}_{pd} = (\Delta \mathbf{x}_{pd}, \Delta \boldsymbol{\lambda}_{pd}, \Delta \mathbf{v}_{pd})$ and with the residual functions being :

(1) Dual residual

$$r_{t,d}(\mathbf{y}) = \mathbf{c} - \boldsymbol{\lambda} + \mathbf{A}^T \mathbf{v}$$

(2) Centrality residual

$$r_{t,c}(\mathbf{y}) = \boldsymbol{\lambda} \odot \mathbf{x} - \frac{1}{t}\mathbf{1}$$

(3) Primal residual

$$r_{t,p}(\mathbf{y}) = \mathbf{A}\mathbf{x} - \mathbf{b}$$

The trajectory generated from the primal dual is shown in Fig. 2. Note that we add a random vector to the initial point as the algorithm can converge even for not feasible initial points.

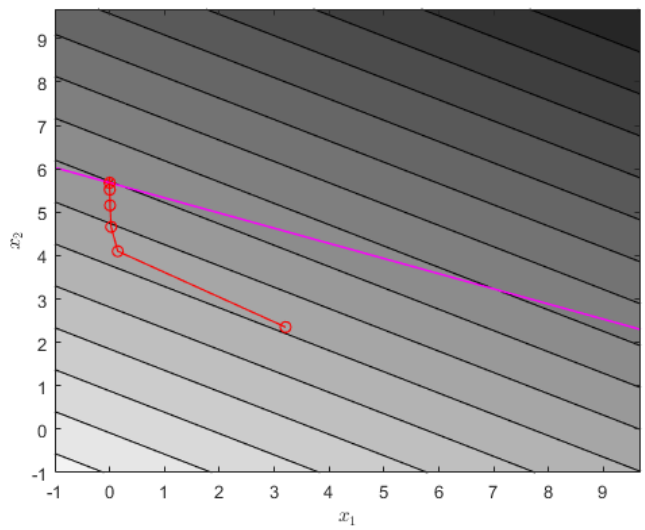

Fig. 2

After running the algorithms various times and increasing the dimensions, we see that the number of nonzero elements of the optimal point is approximately equal to p.

We can also observe that the primal-dual algorithm converges faster than the interior point method. We also see that for the same tolerance the primal-dual algorithm is more accurate.