Mathematics 2, Part 4, Homework 2

I. Duals and duals and duals

1)

For the first task, as instructed we can rewrite the (D) problem in the standard form:

$$\begin{aligned} \max\ & -b^\top y \\ -A^\top y &\le -c \\ y &\ge 0 \end{aligned}$$

Then we can dualize according to the given pattern to get:

$$\begin{aligned} \max\ & -c^\top x \\ -A^\top x &\ge -b \\ x &\ge 0 \end{aligned}$$

And finally by multiplying with -1, we get back to the (P) problem:

$$\begin{aligned} \max\ & c^\top x \\ A^\top x &\le b \\ x &\ge 0 \end{aligned}$$

2)

For this problem we took the (P) and (D) problem from [1]:

$$\begin{array}{ll} \text{(P):} & \text{(D):} \\ \min c^\top x & \max b^\top y \\ Ax = b & A^\top y + s = c \\ x \ge 0 & s \ge 0,\ y \in \mathbb{R}^m \end{array}$$

We show that they are dual by starting from the (D) problem. Firstly, we eliminate the slack s to get: $A^T y \le c$. Then we can convert y into nonnegative variables: $y = y_+ - y_-$. By doing that we can rewrite the maximization to be:

$$\max\ \begin{bmatrix} b & -b \end{bmatrix} \begin{bmatrix} y_+ \\ y_- \end{bmatrix}$$

And the constraints:

$$\begin{bmatrix} A^T & -A^T \end{bmatrix} \begin{bmatrix} y_+ \\ y_- \end{bmatrix} \le c, \begin{bmatrix} y_+ \\ y_- \end{bmatrix} \ge 0$$

By doing this we achieve the standard form and can now use the pattern formula to get:

$$\begin{aligned} \min\ & c^T x \\ \begin{bmatrix} A^T & -A^T \end{bmatrix} x\ &\ge\ \begin{bmatrix} b \\ -b \end{bmatrix} \\ x &\ge 0 \end{aligned}$$

If we look closely at the second line we can see that it must hold that: $A^T x \ge b$ and $-A^T x \ge -b$, which translates to $A^T x \le b$ if we multiply by -1. With that, the only option is that $A^T x = b$. With that we arrive at our (P) problem.

II. Interior point algorithm implementation

5)

Here we compute the dual of problem X. We can notice that this is the (P) problem from task 2), so we need to firstly extract the vectors and the matrix:

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix}, \quad c = \begin{bmatrix} -3 \\ -4 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad b = \begin{bmatrix} 4 \\ 3 \\ 4 \end{bmatrix}, \quad A = \begin{bmatrix} 3 & 3 & 3 & 0 & 0 \\ 3 & 1 & 0 & 1 & 0 \\ 1 & 4 & 0 & 0 & 1 \end{bmatrix}$$

Than we can simply use the formula shown in task 2) to calculate the dual which now looks like the (D) problem from task 2). By evaluating the matrix multiplications we get the requested dual problem:

$$\begin{aligned} \max\ & 4y_1 + 3y_2 + 4y_3 \\ \\ 3y_1 + 3y_2 + y_3 + s_1 &= -3 \\ 3y_1 + y_2 + 4y_3 + s_2 &= -4 \\ 3y_1 + s_3 &= 0 \\ y_2 + s_4 &= 0 \\ y_3 + s_5 &= 0 \\ s_1, s_2, s_3, s_4, s_5 &\ge 0 \\ y_1, y_2, y_3 &\in \mathbb{R} \end{aligned}$$

6)

Since the elements of x and s are strictly positive, we only need to check if $Ax = b$ and $A^T y + s = c$. We confirmed that using Pyhthon and with that we can claim that these vectors are strictly feasible solutions to both problem X and its dual.

7)

To show that the vectors are a good starting solution, we need to check the invariants from [1] and the lectures. We already checked the primal and dual feasibility in the previous task, so we only need to satisfy the third one:

$$\sigma^2 = \sum_{i=1}^{m} \left(\frac{x_i s_i}{\mu} - 1 \right)^2 \le \frac{1}{4}$$

By picking μ ad the average of $x_i s_i$ and calculating the left-hand size we get 0.0016, which easily satisfies the condition. Picking $\mu = 1$ is also appropriate since we get 0.0025.

8)

In this section we iterated our algorithm until it stabilized, by checking how much the primal and dual variables changed. We stopped when both changes were less the 10^{-8}. It took 220 itertions to reach that point. The final points were:

$$x = \begin{bmatrix} 0.444 & 0.889 & 1.49 \times 10^{-8} & 0.778 & 1.19 \times 10^{-8} \end{bmatrix}$$

$$y = \begin{bmatrix} -0.889 & -5.11 \times 10^{-8} & -0.333 \end{bmatrix}$$

$$s = \begin{bmatrix} 8.27 \times 10^{-8} & 4.14 \times 10^{-8} & 2.67 & 4.72 \times 10^{-8} & 0.333 \end{bmatrix}$$

9)

We can speed up the convergence by setting the mew to change adaptively. By trial and error we deduct that we can use:

$$\mu = \frac{0.6(x^T s)}{m}$$

This way we reach convergence in 37 iterations, while also still satisfying the invariants.

10)

Here we heuristically decide when to stop the process. When μ drops below 10^{-6}, we set the appropriate components of s and x to 0. We obtain the same results as previously, with the difference that some of the components are exactly 0 because we set them to zero. By doing that we get extra collection of scalar equations:

$$x_3 = 0$$

$$x_5 = 0$$

$$s_1 = 0$$

$$s_2 = 0$$

$$s_4 = 0$$

A. 11)

To test weather our choice was correct, we solved the equations exactly using the Python sympy package (for symbolic solving). We achieve the same results as before, therefore our choice was correct. To test if the solutions are indeed feasible and optimal we checked the criteria in Table I.

Check	**Value**
Primal residual $\|Ax - b\|$	0.0
Dual residual $\|A^T y + s - c\|$	0.0
Complementary slackness $\max_i \|x_i s_i\|$	0.0
Nonnegativity satisfied	True
Objective gap $\|c^T x - b^T y\|$	0.0

Table I
FEASIBILITY AND OPTIMALITY DIAGNOSTICS FOR THE EXACT SOLUTION OF THE LP WITH HEURISTIC ROUNDING.

III. Commercial solver

For this section, we used scipy's commercial solver and used it on problem X, using both the simplex method and interior-point method. We found the same optimum as in our implementations with both methods, however using the commercial solver, we reached convergence a lot faster. Concretly, we needed 3 iterations for the simplex method and 5 iterations for the interior-point method.

References

[1] Kurt Mehlhorn and Sanjeev Saxena. A still simpler way of introducing interior-point method for linear programming. *Computer Science Review*, 22:1–11, 2016.