ORF307_HW6

March 23, 2022

ORF307 Homework 6

Due: Friday, April 1, 2022 9:00 pm ET

- Please export your code with output as pdf.
- If there is any additional answers, please combine them as **ONE** pdf file before submitting to the Gradescope.

Question 1

Let A be a given matrix. Show that exactly one of the following alternatives must hold.

(a) There exists some $x \neq 0$ such that $Ax = 0, x \geq 0$.

(b) There exists some y such that $A^T y > 0$.

Question 2

An alternative to the phase-I/phase-II method for solving the LP

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax = b, \\ & x \geq 0 \end{array} \tag{1}$$

is the "big-M"-method, in which we solve the auxiliary problem

$$\begin{array}{ll} \text{minimize} & c^T x + M\mathbf{1}^T z \\ \text{subject to} & Ax + z = b \\ & x \geq 0, z \geq 0, \end{array} \tag{2}$$

where $M > 0$ is a parameter and $z \in \mathbf{R}^m$ is an auxiliary variable. Here, $A \in \mathbf{R}^{m \times n}, c \in \mathbf{R}^n$, and $b \in \mathbf{R}^m$. Note that this auxiliary problem has an initial basic feasible solution $(x, z) = (0, b) \geq 0$.

(a) Derive the dual LP of (2).

(b) Prove the following property:

If $M > -y_i^\star$ for $i = 1, \dots, m$, where $y^\star$ is an optimal solution of the dual of (1), then the optimal z in (2) is zero, and therefore the optimal x in (2) is also an optimal solution of (1).

Hint: Use complementary slackness.

Question 3

Consider the following LP:

$$\begin{aligned} &\text{minimize} && 13x_1 + 10x_2 + 6x_3 \\ &\text{subject to} && 5x_1 + x_2 + 3x_3 = 8 \\ &&& 3x_1 + x_2 = 3 \\ &&& x_1, x_2, x_3 \geq 0. \end{aligned} \tag{3}$$

(a) Solve it using the big-M formulation as in Q3 obtaining optimal primal and dual variables (use the provided function).

(b) Derive the dual LP for (3)

(c) Solve the dual using CVXPY and compare the optimal primal-dual variables with the ones from (a).

[2]:
```
'''
This code is provided to help with question 4.
This code returns optimal primal variables x
and dual variables y.
'''
import cvxpy as cp
import numpy as np
import numpy.linalg as la

def simplex_iteration(x, B, problem):
    """Perform one simplex iteration given
    - basic feasible solution x
    - basis B

    It returns new x, new basis, new dual variable,
    and termination flag (true/false)
    """
    A, b, c = problem['A'], problem['b'], problem['c']
    m, n = A.shape
    A_B = A[:, B]

    # Compute reduced cost vector
    p = la.solve(A_B.T, c[B])
    c_bar = c - A.T @ p

    # Check optimality
    if np.all(c_bar >= 0):
        print("Optimal solution found!")
        return x, B, -p, True

    # Choose j such that c_bar < 0 (first one)
    j = np.where(c_bar < 0)[0][0]
```

```python
    # Compute search direction d
    d = np.zeros(n)
    d[j] = 1
    d[B] = la.solve(A_B, -A[:, j])

    # Check for unboundedness
    if np.all(d >= 0):
        print("Unbounded problem!")
        return None, None, True

    # Compute step length theta
    d_i = np.where(d[B] < 0)[0]
    theta = np.min(- x[B[d_i]] / d[B[d_i]])
    i = B[d_i[np.argmin(- x[B[d_i]] / d[B[d_i]])]]

    # Compute next point
    x_next = x + theta * d

    # Compute next basis
    B_next = B
    B_next[np.where(B == i)[0]] = j

    return x_next, B_next, -p, False

def simplex_algorithm(x, B, problem, max_iter=1000):
    """Run simplex algorithm"""

    for k in range(max_iter):

        x, B, y, end = simplex_iteration(x, B, problem)

        if end:
            break
    return x, B, y
```