ProblemSet 3 – Dual linear programs - solutions

George McNinch

2024-02-09

1. Consider the primal linear program

`Maximize`

$$11x + 5y$$

`subject to`

$$\begin{bmatrix} 1 & 1 \\ 10 & 4 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \le \begin{bmatrix} 7 \\ 40 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} x \\ y \end{bmatrix} \ge \mathbf{0}$$

a. Write the dual linear program.

The dual linear program is:

`minimize`

$$7u + 40v$$

subject to

$$\begin{bmatrix} 1 & 10 \\ 1 & 4 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \ge \begin{bmatrix} 11 \\ 5 \end{bmatrix}$$

Here the entries (variables) u and v are the *dual prices*.

b. Find the optimal solution to both the primal and the dual linear programs. You may do this using `python`'s `linprog`, or by plotting the feasible sets. Confirm that both the strong duality theorem and complementary slackness are satisfied. What are the dual prices of each of the constraints?

```
import numpy as np

from scipy.optimize import linprog

c = np.array([11,5])

A = np.array([[1,1],[10,4]])
```

```
b = np.array([7,40])

primal = linprog((-1)*c,A_ub = A, b_ub = b)

dual = linprog(b,A_ub = (-1)*A.T, b_ub = (-1)*c)

print("primal: \n",primal,"\n")
print("dual: \n",dual)
```

Resulting output:

```
primal:
         message: Optimization terminated successfully. (HiGHS
        Status 7: Optimal)
        success: True
         status: 0
            fun: -47.0
              x: [ 2.000e+00  5.000e+00]
            nit: 2
          lower:  residual: [ 2.000e+00  5.000e+00]
                 marginals: [ 0.000e+00  0.000e+00]
          upper:  residual: [       inf        inf]
                 marginals: [ 0.000e+00  0.000e+00]
          eqlin:  residual: []
                 marginals: []
        ineqlin:  residual: [ 0.000e+00  0.000e+00]
                 marginals: [-1.000e+00 -1.000e+00]
 mip_node_count: 0
 mip_dual_bound: 0.0
        mip_gap: 0.0

dual:
         message: Optimization terminated successfully. (HiGHS
        Status 7: Optimal)
        success: True
         status: 0
            fun: 47.0
              x: [ 1.000e+00  1.000e+00]
            nit: 2
          lower:  residual: [ 1.000e+00  1.000e+00]
                 marginals: [ 0.000e+00  0.000e+00]
          upper:  residual: [       inf        inf]
                 marginals: [ 0.000e+00  0.000e+00]
          eqlin:  residual: []
                 marginals: []
        ineqlin:  residual: [ 0.000e+00  0.000e+00]
                 marginals: [-2.000e+00 -5.000e+00]
 mip_node_count: 0
 mip_dual_bound: 0.0
        mip_gap: 0.0
```

Notice that the two linear programs have the same optimal value (recall that `primal.fun` is negative because we `maximize` in this program).

Indeed:

```
abs(primal.fun) == abs(dual.fun)

==> True
```

This confirms the *Strong Duality Theorem* in this case.

Let `x0 = primal.x` be the optimal solution found by the primal linear program. The primal slack vectors is given by `b - A @ x0`

```
b - A @ primal.x
=> array([0., 0.])

## or simply
primal.slack
array([0., 0.])
```

Let `u0 = dual.x` be the optimal solution found by the dual linear program. The dual slack vector is given by `A.T @ y0 - c`

```
>>> A.T @ dual.x - c
=> array([ 0.0000000e+00, -8.8817842e-16])

>>> dual.slack
=> array([0., 0.])
```

So the dual slack vector is also 0 (note that `A.T @ dual.x - c` is very close to 0; this is just an artifact of floating point arithmetic.).

At any rate, we can confirm the *Complementary Slackness Theorem*:

```
>>> primal.slack @ dual.x == dual.slack @ primal.x
==> True
```

This is of course easy to see directly since both `primal.slack` and `dual.slack` are just the zero vectors.

The dual price of the constraint `x + y <= 7` is u=1 – i.e. `dual.x[0]`–, and the dual price of the constraint `10x + 4y <= 40` is v=1 – i.e. `dual.x[1]`.

c. Does the dual price provide an accurate prediction of the increase in the primal objective function when the right-hand side of the first constraint is increased from 7 to 8? From 7 to 9? From 7 to 11?

We consider the change in the objective function for the primal linear program when the first constraint is changed.

Thus we consider perturbing the constraints by the vector $\mathbf{\Delta b} = \begin{bmatrix} c \\ 0 \end{bmatrix}$.

The *dual price lemma* says that if x^* is an ,optimal solution to the original linear program and if x' is an optimal solution to the linear program

maximize $11x + 5y$ subject to

$$\begin{bmatrix} 1 & 1 \\ 10 & 4 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \le \begin{bmatrix} 7+c \\ 40 \end{bmatrix} = \begin{bmatrix} 7 \\ 40 \end{bmatrix} + \mathbf{\Delta b} \quad \text{and} \quad \begin{bmatrix} x \\ y \end{bmatrix} \ge \mathbf{0}$$

then

$$cx' \le cx^* + \mathbf{\Delta b}^T u^* = 47 + [c \quad 0] \begin{bmatrix} 7 \\ 40 \end{bmatrix} = 47 + 7c$$

and that *equality holds when c is small enough*.

We investigate the solutions to the perturbed linear program for the indicated values of c:

```
def deltaB(d):
  return np.array([d,0])



def get_optimal(d):
    # run the linear program with the perturbed upper bounds
    result = linprog((-1)*c, A_ub=A, b_ub = b + deltaB(d))

    # now compare the results
    s1 = f"d = {d}: x + y <= {7+d}, x' = {abs(result.fun):.2f}"
    s2 = f"x* + Δb @ dual.x = {abs(primal.fun) + deltaB(d) @
        dual.x}"
    return s1 + ", " + s2

from pprint import pprint
pprint([ get_optimal(d) for d in [1,2,4]])
=>
["d = 1: x + y <= 8, x' = 48.00, x* + Δb @ dual.x = 48.0",
 "d = 2: x + y <= 9, x' = 49.00, x* + Δb @ dual.x = 49.0",
 "d = 4: x + y <= 11, x' = 50.00, x* + Δb @ dual.x = 51.0"]
```

Thus when d=1,2 the equality prediction of the dual price Lemma holds, but when d=4, the value of the objective function is only 50, while the dual price lemma predicted it would be 51.

2. Consider the linear program

maximize $y + z$

subject to $\mathbf{x} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \geq 0$

and $A\mathbf{x} \leq \mathbf{b}$

where $A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -1 \end{bmatrix}$ and $\mathbf{b} = \begin{bmatrix} 10 \\ 1 \end{bmatrix}$.

a. What is the value of the objective function at points of the form

$$\mathbf{p}(c,t) = \begin{bmatrix} c \\ 0 \\ 0 \end{bmatrix} + t \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} c \\ t \\ t \end{bmatrix} \quad c, t \in \mathbb{R}?$$

The value of the objective function at $\mathbf{p}(c,t)$ is $2t$

b. Under what conditions on c, t is the point $\mathbf{p}(c,t)$ in the *feasible region* of the linear program?

Notice that

$$A \cdot \mathbf{p}(c,t) = \begin{bmatrix} c \\ 0 \end{bmatrix}.$$

Thus the inequality $A \cdot \mathbf{p}(c,t) \leq \mathbf{b}$ holds for all t provided that $c \leq 10$.

c. Does the linear program have an optimal solution?

The linear program has no optimal solution. Indeed, the value of the objective function at the feasible point $\mathbf{p}(1,t)$ is $2t$ and $2t \to \infty$ as $t \to \infty$. Thus, there is no maximum value for the objective function on feasible points.

d. What is the dual linear program? Does the dual linear program have any feasible points? Does it have an optimal solution?

The dual program is

`minimize` $10u + v$ `subject to` $A^T \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \geq \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$.

The inequality condition amounts to the inequalities: $u \geq 0$, $v \geq 1$, and $-v \geq 1$.

But $v \geq 1$ and $v \leq -1$ are incompatible, so there are *no feasible points* (and hence no optimal solution).

---

3. An author of a dystopian novel wants to write a scene in which a characters plans and builds a *doomsday shelter* under his home.

   In the novel, the character will store food supplies in a large underground storage container, which has 50 liters of storage in which he will store `dried` beans and `rice`.

   It seems at least somewhat realistic to expect that a liter of beans provides nutrition for approx. 9 days, while a liter of rice provides nutrition for approx. 5 day.

   Each liter of beans costs \$12.0 and each liter of rice costs \$5.00.

   The character will spend \$60.

   a. Write the primal and dual linear programs.

      In each case, indicate the variables, the objective, and the constraints.

The primal linear program is

`maximize`: $9x_1 + 5x_2$
subject to $A \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq \begin{bmatrix} 50 \\ 60 \end{bmatrix}$ where $A = \begin{bmatrix} 1 & 1 \\ 12 & 5 \end{bmatrix}$ (and where $\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \geq \mathbf{0}$).)

Here x_1 represents the liters of beans purchased and stored, and x_2 represents liters of rice purchased and stored; thus the first entry of $A \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ represents the liters of the storage container used by the supplies, and and the second entry represents the purchase price in dollars of the supplies.

On the other hand, the dual program is

`minimize`: $50y_1 + 60y_2$
subject to $A^T \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \geq \begin{bmatrix} 9 \\ 5 \end{bmatrix}$

Here the y_1 represents the `dual price` of a liter of storage in the container, and y_2 represents the `dual price` of a dollar in the purchase budget.

b. Find solutions to both the primal and dual linear programs. Confirm that both the `strong duality theorem` and `complementary slackness` hold.

```
import numpy as np
from scipy.optimize import linprog

import pprint

c = np.array([9,5])

A = np.array([[1,1],[12,5]])

b = np.array([50 , 60])

primal  = linprog((-1)*c,A_ub = A, b_ub = b)

dual = linprog(b,A_ub = (-1)*A.T, b_ub = (-1)*c)
```

To confirm strong duality, notice

```
print(f"primal optimal value {primal.fun}")
print(f"dual optimal value {dual.fun}")
=>
primal optimal value -60.0
dual optimal value 60.0
```

Now let's check complementary slackness. This amounts to two assertions:

```
primal.slack @ dual.x == 0
=> True

dual.slack @ primal.x == 0
=> True
```

c. Indicate and explain the *dual prices* for each of the `primal` constraints.

We've already identified the dual prices – i.e. the variables y_1, y_2 for the dual linear program – above.

It is maybe useful to analyze the *units* in order to understand why these represent "dual prices".

Well, we should measure y_1 in price/liter and y_2 in price/budget-dollar.

The value of the objective function for the dual program is then

`50 storage-liters × y_1 price /storage-liter + 60 dollars × y_2 price /budget-dollar`

so that the objective function has the (mysterious-seeming) units of “price”.

Note that $A^T \cdot \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \geq \begin{bmatrix} 9 \\ 5 \end{bmatrix}$.

The units of both entries of $A^T \cdot \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$ are measured in

`1 storage-liter/purchase-liter × y_1 price/storage-liter + 12 dollars/purchase-liter × y_2 price/budget-dollar`

This represents the `price/purchase-liter`. In the first row – corresponding to beans – it must exceed 9, and in the second row – corresponding to rice - it must exceed 5; this tells us that `price` is valuing `days of nutrition`.

d. Suppose the author had instead envisioned a storage container holding an additional c liters of food. Does the dual price for this modified constraint provide an accurate prediction for the increase in the primal objective function (i.e. for the number of days of nutrition provided?)

Notice that the optimal dual price is $y_1 = 0$ and $y_2 = 1$

```
>>> dual.x
=> array([0., 1.])
```

This shows that increasing the liters of storage in the container will not increase the value of the primal objective function.

Indeed, the dual price lemma shows if `x'` is the optimal solution to the updated linear program with constraints `b + deltaB` then

`c @ x' <= c @ primal.x + deltaB @ dual.x = c @ primal.x` since `deltaB @ dual.x == 0`.

We can confirm this by checking a few cases

```
def deltaB(d):
    return np.array([d,0])

def compare(c,A,b,d):
    primal  = linprog((-1)*c,A_ub = A, b_ub = b)
    dual    = linprog(b,A_ub = (-1)*A.T, b_ub = (-1)*c)
    tweaked = linprog((-1)*c, A_ub=A, b_ub = b + deltaB(d))
    print(b+deltaB(d))
```

```
    s1 = f"d = {d}: x_1 + x_2 <= {50+d}, c @ x' =
{abs(tweaked.fun):.2f}"
    s2 = f"c @ x* + Δb @ dual.x = {abs(primal.fun) +
deltaB(d) @ dual.x}"
    return s1 + ", " + s2

results = [ compare(c,A,b,d) for d in range(0,10,2) ]

pprint(results)

=>
["d = 0: x_1 + x_2 <= 50, c @ x' = 60.00, c @ x* + Δb @
dual.x = 60.0",
 "d = 2: x_1 + x_2 <= 52, c @ x' = 60.00, c @ x* + Δb @
dual.x = 60.0",
 "d = 4: x_1 + x_2 <= 54, c @ x' = 60.00, c @ x* + Δb @
dual.x = 60.0",
 "d = 6: x_1 + x_2 <= 56, c @ x' = 60.00, c @ x* + Δb @
dual.x = 60.0",
 "d = 8: x_1 + x_2 <= 58, c @ x' = 60.00, c @ x* + Δb @
dual.x = 60.0",
```