# hw10

October 31, 2018

## 1 HW 10

Given the following formulation:

$$
\begin{aligned}
\min \sum_{j=1}^{n} x_j \\
\text{s.t.} \sum_{j=1}^{n} A_j x_j = b \\
x_j \geq 0 \forall j = 1, ..., n
\end{aligned}
$$

The problem has the following data. Customers need three types of smaller widths: $w_1 = 5, w_2 = 12, w_3 = 16$ with quantities $b_1 = 150, b_2 = 100, b_3 = 80$. The width of a big roll is $W = 200$.

Assume the column generation algorithm starts from the following initial patterns:

$A_1 = [40\ 0\ 0]^T A_2 = [0\ 16\ 0]^T A_3 = [0\ 0\ 12]^T$

**1. Write down the restricted master problem (RMP) using these patterns.**

$$
\begin{aligned}
\min\ & x_1 + x_2 + x_3 \\
\text{s.t.}\ & 40x_1 = 150 \\
& 16x_2 = 100 \\
& 12x_3 = 80 \\
& x_1, x_2, x_3 \geq 0
\end{aligned}
$$

**2. Solve RMP in CVX.**

```
In [1]: import cvxpy as cp
        import numpy as np
        from scipy import linalg

        #setup variables and coeffcients
        x = cp.Variable(3, 1)
        A = np.array([[40.,0.,0.],[0.,16.,0.],[0.,0.,12.]])
        b = np.array([150., 100., 80.])
        W = 200.
        w = np.array([5., 12., 16.])
        c = np.array([1., 1., 1.])
```

```
    #setup objective and constraints
    objective = cp.Minimize(cp.sum(x))
    constraints = [A*x == b, x >= 0.]

    # solve
    prob = cp.Problem(objective, constraints)
    result = prob.solve()

    # display optimal value of variables
    print('The solution status is', prob.status)
    print('The optimal value is', round(result))
    print('The optimal [x1, x2, x3] is', [round(xx,2) for xx in x.value])
    print('\nB is\n', A)

    BI =  linalg.inv(A)
    y = c.dot(BI)
    print('\nThe inverse of B is\n', BI)
    print('\nThe optimal dual solution is\n', y)
```

```
The solution status is optimal
The optimal value is 17.0
The optimal [x1, x2, x3] is [3.75, 6.25, 6.67]

B is
 [[40.  0.  0.]
 [ 0. 16.  0.]
 [ 0.  0. 12.]]

The inverse of B is
 [[ 0.025       0.          -0.        ]
 [ 0.          0.0625     -0.        ]
 [ 0.          0.          0.08333333]]

The optimal dual solution is
 [0.025      0.0625     0.08333333]
```

**3. Write down the pricing problem.** The pricing problem is of the form:

$$\begin{aligned} \max\ & .025a_1 + .0625a_2 + .083a_3 \\ \text{s.t. } & 5a_1 + 12a_2 + 16a_3 \leq 200 \\ & a_1, a_2, a_3 \geq 0, \text{integers} \end{aligned}$$

**4. Solve the pricing problem in CVX.**

```
In [2]: a = cp.Variable(3,1, integer=True)
```

```
    #setup objective and constraints
    objective = cp.Maximize(cp.sum(y*a))
    constraints = [w*a <= W, a >= 0.]

    # solve
    prob = cp.Problem(objective, constraints)
    result = prob.solve()

    # display optimal value of variables
    print('The solution status is', prob.status)
    print('The optimal value is', round(result))
    print('The optimal [a1, a2, a3] is', [round(aa,2) for aa in a.value])
```

```
The solution status is optimal_inaccurate
The optimal value is 1.0
The optimal [a1, a2, a3] is [0.0, 11.0, 4.0]
```

We can see in the above that the optimal solution is non-negative, therefore we have the optimal set of columns and there is no need to continue.

**6. Write down the final optimal solution, the optimal basis, and the optimal objective value.**
The optimal value is 17.0

The optimal $\{x_1, x_2, x_3\}$ is $\{3.75, 6.25, 6.67\}$

The optimal basis is:

$$\begin{bmatrix} 40 & 0 & 0 \\ 0 & 16 & 0 \\ 0 & 0 & 12 \end{bmatrix}$$