math_464_actual_hw3

February 1, 2024

Math 464. HW3. Lazizbek Sadullaev

1 *Integer Program*

The integer program I've come up with the given problem is as follows:

[12]:
```
"""
All decision variables are from integers:
"""

import pandas as pd
import numpy as np
import scipy.optimize as opt

# The problem we will solve is:
#
# min z = 0 x1 + 0 x2 + 0 x3 + 0 x4 + 0 x5 + delta
# s.t.   35 x1 + 26 x2 + 21  x3 + 15 x4 + 4 x5 - delta <= 235
#        -35 x1 - 26 x2 - 21  x3 - 15 x4 - 4 x5 - delta <= -235
#         6 x1 +  5 x2 + 7 x3 + 4 x4 + 8 x5 <= 57
#           x >= 1
#           x1, x2,x3,x4, x5 in Z

# First build the objective vector.
c=np.array([0, 0, 0, 0, 0, 1])

# Next, create the coefficient array for the inequality constraints.
# Note that the inequalities must be Ax <= b, so some sign
# changes result when converting >= into <=.
A = np.array([[  35,   26,   21,   15,  4,  -1],\
              [ -35,  -26,  -21,  -15, -4,  -1],\
              [   6,    5,    7,    4,  8,   0]] )

# Next the right-hand-side vector for the inequalities
# Sign changes can occur here too.
b = np.array([235 , -235 , 57 ])

#The coefficient matrix for the equality constraints and
```

```
# the right hand side vector.
Ae = None      # Ae = [[1,1,1,1]]
be = None

# Next, we provide any lower and upper bound vectors, one
# value for each decision variable.  In this example all
# lower bound are zero and there are no upper bounds.
bounds=((1,np.inf),(1,np.inf),(1,np.inf),(1,np.inf), (1,np.inf), (1,np.inf))

# Lastly, we can specify which variables are required to be integer.
# If no variables are integer then isint=[];  In our example, only x2
# is integer.
isint=[1,1,1,1,1, 0]

# The call to the mixed integer solver looks like the following.
# Notice that we pass usual "c" when we have a maximization
# problem, we send "-c".  This is because the solver is expecting a␣
 ↪minimization.
res=opt.linprog(c,A,b,Ae,be,bounds,integrality=isint)

# The result is stored in the dictionary variable "res".
# In particular, to show the optimal objective value and the
# optimal decision variable values:
print("min z = ", np.dot(res['x'][:5], A[0][:5]))
print(res['fun'])
print(res['x'])
print(np.dot(res['x'], A[0])) # to verify if the answer is corret
print(np.dot(res['x'], A[1]))
print(np.dot(res['x'], A[2]))

print(res)
# print(model.computeIIS())
print(35*4+26+21+15*3+4)
```

```
min z =  236.0
1.0
[4. 1. 1. 3. 1. 1.]
235.0
-237.0
56.0
        message: Optimization terminated successfully. (HiGHS Status 7: Optimal)
        success: True
         status: 0
            fun: 1.0
              x: [ 4.000e+00  1.000e+00  1.000e+00  3.000e+00  1.000e+00
                   1.000e+00]
            nit: -1
```

```
          lower:  residual: [ 3.000e+00  0.000e+00  0.000e+00  2.000e+00
                              0.000e+00  0.000e+00]
                 marginals: [ 0.000e+00  0.000e+00  0.000e+00  0.000e+00
                              0.000e+00  0.000e+00]
          upper:  residual: [       inf        inf        inf        inf
                                    inf        inf]
                 marginals: [ 0.000e+00  0.000e+00  0.000e+00  0.000e+00
                              0.000e+00  0.000e+00]
          eqlin:  residual: []
                 marginals: []
        ineqlin:  residual: [ 0.000e+00  2.000e+00  1.000e+00]
                 marginals: [ 0.000e+00  0.000e+00  0.000e+00]
 mip_node_count: 1
 mip_dual_bound: 1.0
        mip_gap: 0.0
236
```

I've read an article on the web about **How do I determine why my model is infeasible?**. Although I could not exactly understood I've tried to implement it above to identify the things to modify.

I'll give my ideas on how I came up with the model above on a separate paper!

[ ]: