ISyE524: Introduction to Optimization
Problem Set #6
Due: *Tuesday* May 5, at 11:59PM

Notes and Deliverables:

- Submit solutions to all problems in the form of a single PDF file of the IJulia notebook to the course dropbox. (If you do not submit as a single PDF you will lose points.)
- Be sure that your answers (including any pictures you include in the notebook) are in the correct order. (If you do not submit the problems in order, you will lose points.)

1 Buying Barley

You are a beer producer who needs to buy 800 pounds of barley. There are three suppliers who are willing to provide barley, quoting prices for various quantities of the product. Each supplier offers decreasing prices for increased lot size, in the form of incremental discounts. The quoted prices for each quantity are given in the table below.

Supplier 1		**Supplier 2**		**Supplier 3**	
# Pounds	**Price/Pound**	**# Pounds**	**Price/Pound**	**# Pounds**	**Price/Pound**
[0,100]	9.2	[0,50]	9	[0,100]	11
[100,200]	9	[50,250]	8.5	[100,300]	8.5
[200,800]	7	[250,800]	8.3	[300,800]	7.5

The costs are incremental. For example, if you wish to purchase 300 pounds of barley from supplier 1, you would pay

$$100(9.2) + (200 - 100)(9) + (300 - 200)7 = 2520.$$

You would like to buy your barley at the least total cost, yet not buy too much from any one supplier. You can buy at most 40% of the total required from any one supplier.

1-1 Problem Write an integer program to determine the minimum cost purchase of barley. If you write parts of the formulation in general form, you need to specify what are the values of the parameters for the data set given in this problem.

Answer:
Our initial decision variables are as follows:

- x_i: Pounds of barley to purchase from supplier $i \in I$

For the objective function, we model to model three piecewise-linear functions, one for each supplier. Each piecewise-linear function has three intervals. j Let $f_1(x_1)$ be the cost of the amount ordered from supplier 1. We know:

$$f_1(x_1) = 9.2x_1 \quad , x_1 \in [0, 100].$$

For $x_1 \in [100, 200]$ the cost is:

$$f_1(x_1) = 100 * 9.2 + (x_1 - 100) * 9 = 9x + 20$$

For $x_1 \in [200, 800]$,

$$f_1(x_1) = 100 * 9.2 + 200 * 9 + (x_1 - 200) * 7 = 7x_1 + 420.$$

To model this piecewise-linear function, we follow the template from the lectures slides and introduce a variavble t_1 to represent the cost, and binary variables $y_{1,k}$ for $k = 1, 2, 3$ and continuous variables $w_{1,k}$ for $k = 1, 2, 3$, and add the constraints:

$$\begin{aligned} w_{11} + w_{12} + w_{13} &= x_1 \\ y_{11} + y_{12} + y_{13} &= 1 \\ 9.2w_{11} + 9w_{12} + 7w_{13} + 20y_{12} + 420y_{13} &= t_1 \\ 0 \le w_{11} &\le 100y_{11} \\ 100y_{12} \le w_{12} &\le 200y_{12} \\ 200y_{13} \le w_{13} &\le 800y_{13}. \end{aligned}$$

Similarly, for supplier $i = 2$, the supplier cost $f_2(x_2)$ is

$$f_2(x_2) = 9x_2 \quad , x \in [0, 50].$$

For $x_2 \in [50, 250]$,

$$f_2(x_2) = 50 * 9 + (x_2 - 50) * 8.5 = 8.5x_2 + 25$$

and for $x_2 \in [250, 800]$,

$$f_2(x_2) = 50 * 9 + 200 * 8.5 + (x_2 - 250) * 8.3 = 8.3x_2 + 75.$$

To model this piecewise-linear function, we again follow the template from the lectures slides and introduce a variavble t_2 to represent the cost, and binary variables $y_{2,k}$ for $k = 1, 2, 3$ and continuous variables $w_{2,k}$ for $k = 1, 2, 3$, and add the constraints:

$$\begin{aligned} w_{21} + w_{22} + w_{23} &= x_2 \\ y_{21} + y_{22} + y_{23} &= 1 \\ 9w_{21} + 8.5w_{22} + 8.3w_{23} + 25y_{22} + 75y_{23} &= t_2 \\ 0 \le w_{21} &\le 50y_{21} \\ 50y_{22} \le w_{22} &\le 250y_{22} \\ 250y_{23} \le w_{23} &\le 800y_{23}. \end{aligned}$$

Finally, for supplier $i = 3$, for $x_3 \in [0, 100]$ we have

$$f_3(x_3) = 11x_3.$$

For $x_3 \in [100, 300]$,

$$f_3(x_3) = 100 * 11 + (x_3 - 100) * 8.5 = 8.5x_3 + 250$$

and for $x_3 \in [300, 800]$,

$$f_3(x) = 100 * 11 + 8.5 * 200 + (x_3 - 300) * 7.5 = 7.5x_3 + 550$$

We again follow the template from the lectures slides and introduce a variavble t_3 to represent the cost, and binary variables $y_{3,k}$ for $k = 1, 2, 3$ and continuous variables $w_{3,k}$ for $k = 1, 2, 3$, and add the constraints:

$$\begin{aligned} w_{31} + w_{32} + w_{33} &= x_3 \\ y_{31} + y_{32} + y_{33} &= 1 \\ 11w_{31} + 8.5w_{32} + 7.5w_{33} + 250y_{32} + 550y_{33} &= t_3 \\ 0 \leq w_{31} &\leq 100y_{31} \\ 100y_{32} \leq w_{32} &\leq 300y_{32} \\ 300y_{33} \leq w_{33} &\leq 800y_{33}. \end{aligned}$$

The objective function then becomes:

$$\min \ t_1 + t_2 + t_3$$

In addition to the above constraints, we just need to enforce that enough supply is purchased, and that not too much is purchased from any one suppliers:

$$\begin{aligned} x_1 + x_2 + x_3 &= 800 \\ x_i &\leq (0.4)(800), \forall i = 1, 2, 3 \\ x_i &\geq 0 \ \forall i = 1, 2, 3 \\ y_{ik} & \text{ binary}, \forall i = 1, 2, 3, \ k = 1, 2, 3 \end{aligned}$$

1-2 Problem Implement and solve the model using the JuMP package in Julia, and explain the solution in terms of the problem.

Answer:

Julia Code

```julia
using JuMP, HiGHS

demand = 800
max_percent = 0.4
suppliers = 1:3
# Piecewise linear functions' parameters
nsegments = 3
slope = [
    9.2 9   7;
    9   8.5 8.3;
    11  8.5 7.5
]
intercept = [
    0 20  420;
    0 25  75;
    0 250 550
```

```
    ]
    endpoint = [
        0 100 200 800;
        0 50  250 800;
        0 100 300 800;
    ]

    m = Model(HiGHS.Optimizer)

    @variable(m, purchase[suppliers] >= 0)
    @variable(m, cost[suppliers]) # t variables in math model
    # Using math model names for other piecewise variables
    @variable(m, y[suppliers,1:nsegments], Bin)
    @variable(m, w[suppliers,1:nsegments])

    @objective(m, Min, sum(cost[s] for s in suppliers))

    @constraint(m, y_sum_to_1[s in suppliers],
        sum(y[s,i] for i in 1:nsegments) == 1)

    @constraint(m, w_sum_to_x[s in suppliers],
        sum(w[s,i] for i in 1:nsegments) == purchase[s])

    @constraint(m, w_lower_bound[s in suppliers, i in 1:nsegments],
        endpoint[s,i]y[s,i] <= w[s,i])

    @constraint(m, w_upper_bound[s in suppliers, i in 1:nsegments],
        w[s,i] <= endpoint[s,i+1]y[s,i])

    @constraint(m, t_defn[s in suppliers],
        cost[s] ==
        sum(slope[s,i]w[s,i] + intercept[s,i]y[s,i]
                    for i in 1:nsegments))

    @constraint(m, meet_demand,
        sum(purchase[s] for s in suppliers) == demand)

    @constraint(m, max_from_one_suppliers[s in suppliers],
        purchase[s] <= max_percent*demand)

    set_silent(m)

    optimize!(m)

    println("\nTotal cost: \$",
      round(objective_value(m),digits=2))
  println("Where to buy things from:")
  for s in suppliers
      println("Buy ",
          round(value(purchase[s]),digits=2),
          " pounds from supplier ",
          s, " for \$",
          round(value(cost[s]),digits=2))
```

```
end
```

Gives answer

Julia Answer

```
Total cost: 6995.0
Where to buy things from:
Buy 320.0 pounds from supplier 1 for 2660.0
Buy 160.0 pounds from supplier 2 for 1385.0
Buy 320.0 pounds from supplier 3 for 2950.0
```

2 Peeps Production in the Lehigh Valley

There are seven cities in the Lehigh Valley: Allentown, Bethlehem, Catasauqua, Durham, Easton, Freemansburg, and Glendon. or 'A', 'B', 'C', 'D', 'E', 'F', and 'G' for short. You are in charge of desigining an important component of the production of Peeps, everyone's favorite marshmallow candy. Peeps production will require 4 factories, and for tax reasons, at most one factory can be located in each of the 7 cities. The distances between each of the possible locations is given in Table 1.

Table 1: Distances Between Various Locations

	A	B	C	D	E	F	G
Allentown	0.0	6.1	15.2	17.0	16.8	8.4	16.6
Bethlehem	6.1	0.0	7.6	16.0	11.3	2.2	11.9
Catasauqua	15.2	7.6	0.0	22.0	17.3	10.2	16.7
Durham	17.0	16.0	22.0	0.0	12.1	14.8	7.2
Easton	16.8	11.3	17.3	12.1	0.0	9.4	2.9
Freemansburg	8.4	2.2	10.2	14.8	9.4	0.0	9.2
Glendon	16.6	11.9	16.7	7.2	2.9	9.2	0.0

Between each pair of factories, A given number of tons of manufacturing product "fluff" must be transported between each pair of factories each month. This amount is given in Table 2. Suppose that the transportation cost of fluff is $1 per mile per ton.

Table 2: Flow Between Factories

	1	2	3	4
1	0	3	2	9
2	5	0	1	2
3	10	4	0	2
4	1	6	3	0

2-1 Problem Write a *mixed integer linear program* that will determine in which city to locate each factory to minimize the total transportation cost

Answer:
We have a set of cities $J = \{A, B, C, D, E, F, G\}$ and a set of factories $I = \{1, 2, 3, 4\}$. $d_{ab}, a \in J, b \in J$ is the distance between each pair of cities, and $f_{ab}, a \in I, b \in I$ is the flow between each pair of factories.

Let x_{ij} be a binary variable if we assign factory $i \in I$ to city $j \in J$.

We wish to minimize the ***nonlinear*** objective

$$\min \sum_{i \in I} \sum_{j \in J} \sum_{k \in I} \sum_{\ell \in J} f_{ik} d_{j\ell} x_{ij} x_{kl},$$

but in order to model this we use the QAP linearization trick and introduce a new binary variables $z_{ijk\ell}$ be a binary variable if we assign factory $i \in I$ to city $j \in J$ and factory $k \in I$ to city $\ell \in J$ that we have to be $z_{ijk\ell} = x_{ij} x_{k\ell}$.

$$\min \sum_{i \in I} \sum_{j \in J} \sum_{k \in I} \sum_{\ell \in J} f_{ik} d_{j\ell} z_{ijk\ell}$$

Constraint #1: Assign each factory to a city

$$\sum_{j \in J} x_{ij} = 1 \quad \forall i \in I$$

Constraint #2: Assign at most one factory to each city

$$\sum_{i \in I} x_{ij} \leq 1 \quad \forall j \in J$$

Constraint #3: Make $z_{ijk\ell} = x_{ij} x_{k\ell}$.

$$\begin{aligned} x_{ij} &\geq z_{ijk\ell} & \forall i \in I, j \in J, k \in I, \ell \in J \\ x_{k\ell} &\geq z_{ijk\ell} & \forall i \in I, j \in J, k \in I, \ell \in J \\ x_{ij} + x_{k\ell} - 1 &\leq z_{ijk\ell} & \forall i \in I, j \in J, k \in I, \ell \in J \end{aligned}$$

Constraint to make everything binary

$$\begin{aligned} z_{ijk\ell} &\in \{0, 1\} \quad \forall i \in I, j \in J, k \in I, \ell \in J \\ x_{ij} &\in \{0, 1\} \quad \forall i \in I, j \in J \end{aligned}$$

2-2 Problem Implement your model from Problem 2-1 in Julia and JuMP to determine the optimal placement.

Answer:

Julia Code

```
using JuMP, HiGHS, NamedArrays
J = [:A, :B, :C, :D, :E, :F, :G]
I = [Symbol("F",i) for i in 1:4]
```

```julia

d_numbers = [
0 6.1 15.2 17.0 16.8 8.4 16.6
6.1 0 7.6 16.0 11.3 2.2 11.9
15.2 7.6 0.0 22.0 17.3 10.2 16.7
17.0 16.0 22.0 0.0 12.1 14.8 7.2
16.8 11.3 17.3 12.1 0.0 9.4 2.9
8.4 2.2 10.2 14.8 9.4 0.0 9.2
16.6 11.9 16.7 7.2 2.9 9.2 0.0
]
d = NamedArray(d_numbers, (J,J))

f_numbers = [
    0 3 2 9
    5 0 1 2
    10 4 0 2
    1 6 3 0
]
f = NamedArray(f_numbers, (I,I))

m = Model(HiGHS.Optimizer)
set_silent(m)
@variable(m, x[I,J], Bin)
@variable(m, z[I,J,I,J], Bin)

#Minimize totla cost
@objective(m,Min,sum(f[i,k]*d[j,l]*z[i,j,k,l] for i in I for j in J for k in I for l in J))

# Assign all factories
@constraint(m, [i in I], sum(x[i,j] for j in J) == 1)

# Don't assign more than 1 factory to a city
@constraint(m, [j in J], sum(x[i,j] for i in I) <= 1)

# Do QAP linearization trick
for i in I
    for j in J
        for k in I
            for l in J
                @constraint(m, z[i,j,k,l] <= x[i,j])
                @constraint(m, z[i,j,k,l] <= x[k,l])
                @constraint(m, z[i,j,k,l] >= x[i,j] + x[k,l] - 1)
            end
        end
    end
end

optimize!(m)
for i in I
    for j in J
        if value(x[i,j]) > 0.5
            println("Assign factory ", i, " to city ", j)
        end
```

```
        end
    end
    println("Total cost: ", objective_value(m))
```

Gives answer

Julia Answer

```
    Assign factory F1 to city F
    Assign factory F2 to city E
    Assign factory F3 to city B
    Assign factory F4 to city G
    Total cost: 166.4
```

3 Lifestyle Malls in the Lehigh Valley

There are seven important shopping regions in the Lehigh Valley: Allentown, Bethlehem, Catasauqua, Durham, Easton, Freemansburg, and Glendon. or 'A', 'B', 'C', 'D', 'E', 'F', and 'G' for short. Your partner *really* likes shopping, so they want to visit all the malls in one day. The distances between the various locations is given in Table 3:

Table 3: Distances Between Various Locations

	A	B	C	D	E	F	G
Allentown	0.0	6.1	15.2	17.0	16.8	8.4	16.6
Bethlehem	6.1	0.0	7.6	16.0	11.3	2.2	11.9
Catasauqua	15.2	7.6	0.0	22.0	17.3	10.2	16.7
Durham	17.0	16.0	22.0	0.0	12.1	14.8	7.2
Easton	16.8	11.3	17.3	12.1	0.0	9.4	2.9
Freemansburg	8.4	2.2	10.2	14.8	9.4	0.0	9.2
Glendon	16.6	11.9	16.7	7.2	2.9	9.2	0.0

3-1 Problem You live in Bethlehem. What is the minimum distance round trip to each of the sites that you can make? That is – you start in Bethlehem, visit each mall site exactly once, and return to Bethlehem. What is the sequence of cities to visit? Implement your model in Julia and JuMP to find the sequence of cities and shortest tour.

Answer:

Julia Code

```
using JuMP, HiGHS
J = [:A, :B, :C, :D, :E, :F, :G]
n = length(J)

d = [
```

```
0 6.1 15.2 17.0 16.8 8.4 16.6
6.1 0 7.6 16.0 11.3 2.2 11.9
15.2 7.6 0.0 22.0 17.3 10.2 16.7
17.0 16.0 22.0 0.0 12.1 14.8 7.2
16.8 11.3 17.3 12.1 0.0 9.4 2.9
8.4 2.2 10.2 14.8 9.4 0.0 9.2
16.6 11.9 16.7 7.2 2.9 9.2 0.0
]

m = Model(HiGHS.Optimizer)
set_silent(m)

@variable(m, x[1:n,1:n], Bin)

# can't flow from a node to itself
@constraint(m, nzdiag[i=1:n], x[i,i] == 0 )

# flow constraints (exactly one flow out of each )
@constraint(m,  inflows[i=1:n], sum( x[i,j] for j=1:n ) == 1 )
@constraint(m, outflows[j=1:n], sum( x[i,j] for i=1:n ) == 1 )

# additional variables and constraints for MTZ formulation
@variable(m, u[1:n] >= 0)
@constraint(m, MTZ[i=1:n, j=2:n], u[i] - u[j] + n*x[i,j] <= n-1)

@objective(m, Min, sum( d[i,j]*x[i,j] for i=1:n, j=1:n ))

optimize!(m)

X = value.(x);
for i = 1:n
    for j=1:n
        if (X[i,j] >= .99)
            println(i," is followed by ",j)
        end
    end
end
println("Total length: ", objective_value(m))
```

4 Jerry Mander

Governor Jerry Mander of the state of Texis is attempting to get the state legislator to gerrymander Texis's congressional districts. The state consists of ten cities, and the numbers of registered Republicans and Democrats (in thousands) in each city are shown below

city	republicans	democrats
1	80	34
2	60	44
3	40	44

4	20	24
5	40	114
6	40	64
7	70	14
8	50	44
9	70	54
10	70	64

Texis has five congressional representatives. To form congressional districts, cities must be grouped together according to the following restrictions:

- All voters in a city must be in the same district.
- Each district must contain between 150,000 and 250,000 voters (there are no independent voters).

Governor Mander is a Republican. Assuming 100% voter turnout and that each voter always votes for their registered party.

4-1 Problem Formulate and solve an optimization problem to help Governor Mander maximize the number of Republicans who will win congressional seats.

Hints:

- Binary assignment variables if city i goes in district j.
- Variables for number of republicans and democrats in each district.
- Binary variables forcing more republicans than democrats in a district. Use slide of tricks to say that if the "republican" binary variable for a district is 1, then there must be morerepublicans than democrats in the district.

Answer:

Let C be the set of cities and D be the set of districts. Define the decision variables

- $x_{ij} \in \{0, 1\}$ to indicate if city $i \in C$ was assigned to district $j \in D$
- r_j to be the total number of republicans assigned to district $j \in D$
- d_j to be the total number of democrats assigned to district $j \in D$
- The variable $z_j \in \{0, 1\}$ taking the value 1 will imply that more republicans than democrats are in district $j \in D$

To make the problem easy to write in its general form, I define the parameters ρ_i to be the number of republicans (in thousands) in city $i \in C$ and δ_i to be the number of democrats (in thousands) in city $i \in C$.

$$\begin{aligned}
\max \quad & \sum_{j \in D} z_j \\
\text{s.t.} \quad & \sum_{i \in C} \rho_i x_{ij} = r_j & \forall j \in D \\
& \sum_{i \in C} \delta_i x_{ij} = d_j & \forall j \in D \\
& \sum_{j \in D} x_{ij} = 1 & \forall i \in C \\
& r_j + d_j \geq 150 & \forall j \in D \\
& r_j + d_j \leq 250 & \forall j \in D \\
& r_j - d_j + (-250 - 1) z_j \geq -250 & \forall j \in D \\
& r_j, d_j \geq 0 & \forall j \in D \\
& z_j \in \{0, 1\} & \forall j \in D \\
& x_{ij} \in \{0, 1\} & \forall i \in C, j \in D
\end{aligned}$$

The constraint $r_j - d_j + (-250 - 1)z_j \geq -250$ is the appropriate slide of trix application of the logical implication:

$$z_j = 1 \Rightarrow r_j - d_j \geq 1.$$

The value of $m = -251$. (You could also implement that ties go to republicans)

4-2 Problem Implement your model from Problem 4-1 in JuMP and Julia and solve the model. What is the maximum number of seats Republicans can win?

Answer:

Here is my model file:

Julia Code

```
using JuMP, HiGHS

D = [:A, :B, :C, :D, :E];  # five districts
C = collect(1:10)

# first column is republican, second column is democrat
# columns are cities
data = [ 80 34
60 44
40 44
20 24
40 114
40 64
70 14
50 44
70 54
```

```
    70 64 ];

    rho = data[:,1]
    delta = data[:,2]


    m = Model(HiGHS.Optimizer)
    set_silent(m)

    # does district i contain city j?
    @variable(m, x[C,D], Bin)
    @variable(m, r[D] >= 0)
    @variable(m, d[D] >= 0)

    # do the republicans win this district?
    @variable(m, z[D], Bin)

    # Define number of republican votes
    @constraint(m, def_r[j in D], sum(rho[i]*x[i,j] for i in C) == r[j])
    # Define number of democrat votes in district
    @constraint(m, def_d[j in D], sum(delta[i]*x[i,j] for i in C) == d[j])
    #Assign all cities to exactly one district
    @constraint(m, assign[i in C], sum(x[i,j] for j in D) == 1)
    for j in D
        # Bounds
        @constraint(m, 150 <= r[j] + d[j] <= 250)
        # if z_j = 1 => r_j >= d_j
        @constraint(m, r[j] - d[j] >= -250*(1-z[j]))
    end

    @objective(m, Max, sum(z))

    optimize!(m)
    println("Republicans can win a maximum of ", objective_value(m), " districts.")
```

Julia Answer

```
Republicans can win a maximum of 4.0 districts.
```

4-3 Problem Modify your answer to maximize the number of *Democrats* who win seats.

Answer:

There are different ways to do this one, but I just created the variable $y_j = 1, j \in D$ to be a switch that ensured that a democrat won the district. Then maximized the number of y_j that took the value 1.

$$
\begin{aligned}
\max \quad & \sum_{j \in D} y_j \\
\text{s.t.} \quad & \sum_{i \in C} \rho_i x_{ij} = r_j && \forall j \in D \\
& \sum_{i \in C} \delta_i x_{ij} = d_j && \forall j \in D \\
& \sum_{j \in D} x_{ij} = 1 && \forall i \in C \\
& r_j + d_j \geq 150 && \forall j \in D \\
& r_j + d_j \leq 250 && \forall j \in D \\
& d_j - r_j + (-250 - 1) y_j \geq -250 && \forall j \in D \\
& r_j, d_j \geq 0 && \forall j \in D \\
& y_j \in \{0, 1\} && \forall j \in D \\
& x_{ij} \in \{0, 1\} && \forall i \in C, j \in D
\end{aligned}
$$

4-4 Problem Implement your model from Problem 4-3 in JuMP and Julia and solve the model. What is the maximum number of seats Democrats can win?

Answer:

It is pretty simple to "flip" the model around to maximize the number of democrats:

Julia Code

```julia
m = Model(HiGHS.Optimizer)
set_silent(m)

# does district i contain city j?
@variable(m, x[C,D], Bin)
@variable(m, r[D] >= 0)
@variable(m, d[D] >= 0)

# do the republicans win this district?
@variable(m, z[D], Bin)

# Define number of republican votes
@constraint(m, def_r[j in D], sum(rho[i]*x[i,j] for i in C) == r[j])
# Define number of democrat votes in district
@constraint(m, def_d[j in D], sum(delta[i]*x[i,j] for i in C) == d[j])
#Assign all cities to exactly one district
@constraint(m, assign[i in C], sum(x[i,j] for j in D) == 1)
for j in D
    # Bounds
    @constraint(m, 150 <= r[j] + d[j] <= 250)
    # if z_j = 1 => d_j >= r_j
    @constraint(m, d[j] - r[j] >= -250*(1-z[j]))
end
```

```

    @objective(m, Max, sum(z))

    optimize!(m)
    println("Democrats can win a maximum of ", objective_value(m), " districts.")
```

Solving this shows that at most 3 democrats can win districts.

Julia Answer

```
Democrats can win a maximum of 3.0 districts.
```