RM 294 Optimization Project 1

Christian Alfonso, Brent Hensley, Jessie Lee, Sungho Park

Question 1: ROI Table and Load the CSV

The marketing firms fund allocation for their respective medium is defined in this table.

Platform	Print	TV	SEO	AdWords	Facebook	LinkedIn	Instagram	Snapchat	Twitter	Email
ROI	3.1%	4.9%	2.4%	3.9%	1.6%	2.4%	4.6%	2.6%	3.3%	4.4%

The following graph shows the imported ROI_data.csv file

```
target=pd.read_csv('ROI_data.csv')
target
```

	Platform	Print	TV	SEO	AdWords	Facebook	LinkedIn	Instagram	Snapchat	Twitter	Email
0	ROI	0.031	0.049	0.024	0.039	0.016	0.024	0.046	0.026	0.033	0.044
1	Second Firms ROI Estimate	0.049	0.023	0.024	0.039	0.044	0.046	0.026	0.019	0.037	0.026

Question 2: Boss Constraints

The boss now wants us to specify our budget on 3 constraints: **1.** The amount invested in print and TV should be no more than the amount spent on Facebook and Email. **2.** The total amount used in social media (Facebook, LinkedIn, Instagram, Snapchat, and Twitter) should be at least twice of SEO and AdWords. **3.** For each platform, the amount invested should be no more than $3M.

Question 3: Linear Programming Formulation

We formulated the constraints in to the matrix shown below using the following code:

```
# Constrains:

A = np.zeros((13,10)) # initialize constraint matrix

A[0,:] = [1,1,0,0,-1,0,0,0,0,-1]  # print and TV no more than Facebook and Email
A[1,:] = [0,0,2,2,-1,-1,-1,-1,-1,0]  # Facebook, LinkedIn, Instagram, Snapchat, and Twitter > 2X (SEO+AdWords)
A[2:12,:] = np.identity(10)  # each platform <3
A[12,:] = [1,1,1,1,1,1,1,1,1,1] # total <10
b = np.array([0.,0.,3.,3.,3.,3.,3.,3.,3.,3.,3.,10.]) # constrains
sense = np.array(['<']*13) # all constraints are less than or equal constraints
```

```
1 A
```

```
array([[ 1.,  1.,  0.,  0., -1.,  0.,  0.,  0.,  0., -1.],
       [ 0.,  0.,  2.,  2., -1., -1., -1., -1., -1.,  0.],
       [ 1.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.],
       [ 0.,  1.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.],
       [ 0.,  0.,  1.,  0.,  0.,  0.,  0.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  1.,  0.,  0.,  0.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  0.,  1.,  0.,  0.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  0.,  0.,  1.,  0.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  0.,  0.,  0.,  1.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  0.,  0.,  0.,  0.,  1.,  0.,  0.],
       [ 0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  1.,  0.],
       [ 0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  1.],
       [ 1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.]])
```

Gurobi produces the following optimal investment, which can be interpreted as spending $3M on TV, $1M on AdWords, $3M on Instagram, and $3M on email, and nothing on other mediums.

```
array([0., 3., 0., 1., 0., 0., 3., 0., 0., 3.])
```

Question 4: Second Opinion

The boss might be pleased initially, but they still want a second opinion. Below is the fund allocation for the second ROI calculations of the other firm.

Platform	Print	TV	SEO	AdWords	Facebook	LinkedIn	Instagram	Snapchat	Twitter	Email
ROI	4.9%	2.3%	2.4%	3.9%	4.4%	4.6%	2.6%	1.9%	3.7%	2.6%

Question 5: Comparing ROI's

Below, we have the direct comparison table for the first fund allocation optimization and the second fund allocation optimization to accommodate the change in ROI for each medium. The changes in allocation result in the following shift:

- Dropped funds for TV, Instagram and Email
- Invest in Print, Facebook and LinkedIn
- All others remain the same

	platform	amount_allo_1	amount_allo_2
1	Print	0.0	3.0
2	TV	3.0	0.0
3	SEO	0.0	0.0
4	AdWords	1.0	1.0
5	Facebook	0.0	3.0
6	LinkedIn	0.0	3.0
7	Instagram	3.0	0.0
8	Snapchat	0.0	0.0
9	Twitter	0.0	0.0
10	Email	3.0	0.0

The allocations are not quite the same. If we assume ROI 1 is correct, allocation 1 would see a return of $ 0.456 million while allocation 2 sees $0.252 million. This results in a $0.204 million lower ROI by mistakenly using allocation 2. Inversely, if we assume ROI 2 is correct, allocation 2 would see a return of $0.456 million while allocation 1 sees $0.264 million. This results in a lower ROI of $ 0.192 million by mistakenly using allocation 1.

In [146]:
```
print('Assume ROI 1 is correct:')
print('Allocation 1 would have a return of {}.'.format(round(y1,3)))
print('while allocationi 2 would have a return of {}.'.format(x2@obj1.T))
print('So we would have ${}million lower ROI by mistakenly use allocation 2.'.format(round((y1-x2@obj1.T),3)))
```

```
Assume ROI 1 is correct:
Allocation 1 would have a return of 0.456.
while allocationi 2 would have a return of 0.252.
So we would have $0.204million lower ROI by mistakenly use allocation 2.
```

In [147]:
```
print('Assume ROI 2 is correct:')
print('Allocation 2 would have a return of {}'.format(round(y2,3)))
print('while allocation 1 would have a return of {}'.format(round(x1@obj2.T,3)))
print('So we would have ${}million lower ROI by mistakenly use allocation 1'.format(round((y2-x1@obj2.T),3)))
```

```
Assume ROI 2 is correct:
Allocation 2 would have a return of 0.456
while allocation 1 would have a return of 0.264
So we would have $0.192million lower ROI by mistakenly use allocation 1
```

The boss's cap on each of the funds is useful to constrain the allocation because if you lift the cap for each medium, the funds all go to TV and Email, with $5M on each medium.

platform	Print	TV	SEO	AdWords	Facebook	LinkedIn	Instagram	Snapchat	Twitter	Email
allocation	0.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	5.0

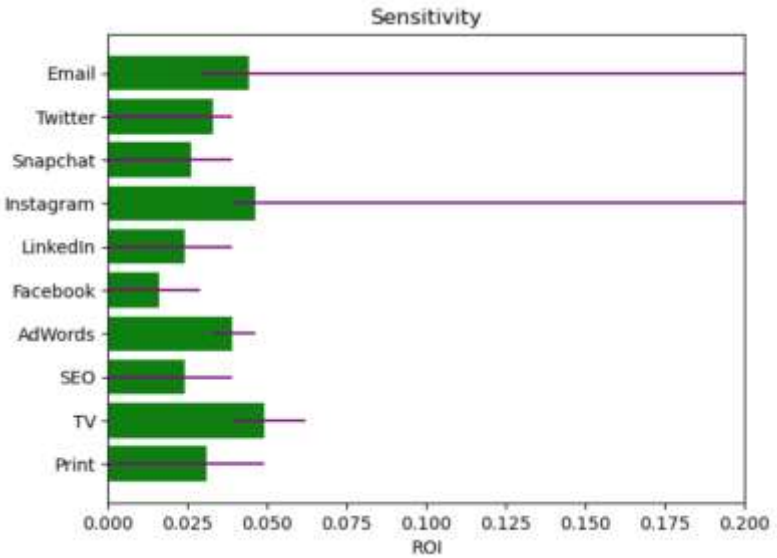
Additionally, if we vary the funds cap for each medium, we get a very different allocation, however, when it's equal or greater than $5 million, the funds all go to TV and Email.

```
allo_change=pd.DataFrame(allo_change)
allo_change.index=counter
allo_change.columns=target_index['platform']
allo_change.index.names=['max for each medium in $M']
allo_change
```

platform max for each medium in $M	Print	TV	SEO	AdWords	Facebook	LinkedIn	Instagram	Snapchat	Twitter	Email
1	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
2	0.0	2.0	0.0	2.0	0.0	0.0	2.0	0.0	2.0	2.0
3	0.0	3.0	0.0	1.0	0.0	0.0	3.0	0.0	0.0	3.0
4	0.0	4.0	0.0	0.0	0.0	0.0	2.0	0.0	0.0	4.0
5	0.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	5.0
6	0.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	5.0
7	0.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	5.0
8	0.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	5.0
9	0.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	5.0
10	0.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	5.0

Question 6: ROI Sensitivity

The following graph shows the sensitivity of return on variation of ROI assumptions on each medium. The green bar is the basecase ROI assumption, and the purple lines show the ROI range for each medium before it changes the fund allocation. When the purple lines touch the edge of the diagram, it indicates that the fund allocation is insensitive to the change of that particular ROI in that direction, i.e., in the extreme case, even when the ROI increases or decreases to infinity, the allocation won't change.



This chart numerically shows what is illustrated in purple above, with **-inf and inf** representing the extreme case of insensitivity (won't change the allocation even going to infinite large or small numbers).

platform	lowerlimit	basecase	upperlimit
Print	-inf	0.031	0.049
TV	0.039	0.049	0.062
SEO	-inf	0.024	0.039
AdWords	0.033	0.039	0.046
Facebook	-inf	0.016	0.029
LinkedIn	-inf	0.024	0.039
Instagram	0.039	0.046	inf
Snapchat	-inf	0.026	0.039
Twitter	-inf	0.033	0.039
Email	0.029	0.044	inf

Question 7: Reinvesting the Returns & Question 8: Budget Stability

Knowing that the same constraints are in place, and that half of the return can be reinvested, we used the following code to create the constraints and solve for the optimal investment solution for each month. If we use a **constant ROI to optimize**, we get a **stable fund allocation** of each month as shown on the table.

Platform	Print	TV	SEO	AdWords	Facebook	LinkedIn	Instagram	Snapchat	Twitter	Email
ROI	3.1%	4.9%	2.4%	3.9%	1.6%	2.4%	4.6%	2.6%	3.3%	4.4%

```
# Initiate the constrains
A = np.zeros((13,10)) # initialize constraint matrix
A[0,:] = [1,1,0,0,-1,0,0,0,0,-1]  # print and TV no more than Facebook and Email
A[1,:] = [0,0,2,2,-1,-1,-1,-1,-1,0]  # Facebook, LinkedIn, Instagram, Snapchat, and Twitter > 2X (SEO+AdWords)
A[2:12,:] = np.identity(10)  # each platform <3
A[12,:] = [1,1,1,1,1,1,1,1,1,1] # total <10
b = np.array([0.,0.,3.,3.,3.,3.,3.,3.,3.,3.,10.]) # All constrains
sense = np.array(['<']*13) # all constraints are less than or equal constraints
# must define the variables before adding constraints because variables go into the constraints

monthly1 = []   #each month's allocation
b_var1 = []  #constrain change through time due to increased funds
b = np.round(b,3) # round the constrains to 3 digits

for i in range(12):  # For each month
    ojModel = gp.Model() # initialize an empty model
    ojModX = ojModel.addMVar(10) # tell the model how many variables there are
    ojModCon = ojModel.addMConstrs(A, ojModX, sense, b) # add the constraints to the model
    ojModel.setMObjective(None,obj1,0,sense=gp.GRB.MAXIMIZE) # add the objective to the model
    ojModel.Params.OutputFlag = 0 # tell gurobi to shut up!!
    ojModel.optimize() # solve the LP
    monthly1.append(ojModX.x)
    y=ojModel.objVal
    b[12]=b[12]+y*0.5     #  add 50% of the return to total fund available for next month
    b_var1.append(b.copy())
df=pd.DataFrame(monthly1)
df.index=['Jan','Feb','Mar','Apr','May','Jun','Jul','Aug','Sep','Oct','Nov','Dec']
df.columns=target_index['platform']
df.round(3)
```

platform	Print	TV	SEO	AdWords	Facebook	LinkedIn	Instagram	Snapchat	Twitter	Email
Jan	0.0	3.0	0.0	1.000	0.0	0.0	3.0	0.0	0.000	3.0
Feb	0.0	3.0	0.0	1.228	0.0	0.0	3.0	0.0	0.000	3.0
Mar	0.0	3.0	0.0	1.460	0.0	0.0	3.0	0.0	0.000	3.0
Apr	0.0	3.0	0.0	1.566	0.0	0.0	3.0	0.0	0.132	3.0
May	0.0	3.0	0.0	1.646	0.0	0.0	3.0	0.0	0.292	3.0
Jun	0.0	3.0	0.0	1.728	0.0	0.0	3.0	0.0	0.456	3.0
Jul	0.0	3.0	0.0	1.811	0.0	0.0	3.0	0.0	0.623	3.0
Aug	0.0	3.0	0.0	1.896	0.0	0.0	3.0	0.0	0.792	3.0
Sep	0.0	3.0	0.0	1.982	0.0	0.0	3.0	0.0	0.964	3.0
Oct	0.0	3.0	0.0	2.070	0.0	0.0	3.0	0.0	1.140	3.0
Nov	0.0	3.0	0.0	2.159	0.0	0.0	3.0	0.0	1.318	3.0
Dec	0.0	3.0	0.0	2.250	0.0	0.0	3.0	0.0	1.500	3.0

However, if we use **the variable ROI** as shown below:

```
monthly_roi=pd.read_csv('roi_mat.csv')     #vaviable ROI
```

	month	Print	TV	SEO	AdWords	Facebook	LinkedIn	Instagram	Snapchat	Twitter	Email
0	January	4.0	3.6	2.4	3.9	3.0	3.5	3.6	2.25	3.5	3.5
1	February	4.0	3.9	2.7	3.8	4.3	3.2	2.7	1.80	3.7	3.5
2	March	3.5	2.9	3.1	3.8	2.4	4.1	3.7	2.60	4.2	2.5
3	April	3.8	3.1	2.4	4.4	2.4	3.8	3.7	2.50	3.6	2.9
4	May	3.5	3.2	1.9	3.4	2.7	2.7	3.9	2.20	4.5	3.9
5	June	4.0	3.2	2.7	3.4	3.4	3.0	4.5	2.10	3.8	4.1
6	July	3.9	3.6	2.0	4.4	3.9	3.7	4.3	1.80	4.0	3.8
7	August	4.2	3.3	2.8	4.2	2.0	3.7	3.6	1.50	4.4	4.3
8	September	4.1	2.8	2.5	4.2	2.9	3.7	2.8	2.50	4.0	3.4
9	October	3.0	3.0	3.1	4.6	3.1	3.3	3.2	2.30	2.5	3.2
10	November	4.8	3.3	2.7	4.1	2.9	3.6	4.2	3.00	3.1	4.1
11	December	4.8	4.0	1.9	3.7	4.2	3.6	2.6	2.90	3.6	3.7

We got the following monthly funds allocation. As we can see from the table, the funds allocation becomes ***'unstable'*** by definition of change of over 1 million dollars from month to month for almost all mediums other than SEO and SnapChat, which stays 0 throughout the year.

platform	Print	TV	SEO	AdWords	Facebook	LinkedIn	Instagram	Snapchat	Twitter	Email
Jan	3.000	0.0	0.0	1.333	0.000	0.000	2.667	0.0	0.000	3.000
Feb	3.000	0.0	0.0	2.396	3.000	0.000	0.000	0.0	1.791	0.000
Mar	0.000	0.0	0.0	3.000	0.000	3.000	1.390	0.0	3.000	0.000
Apr	0.000	0.0	0.0	3.000	0.000	3.000	3.000	0.0	1.597	0.000
May	1.804	0.0	0.0	0.000	0.000	0.000	3.000	0.0	3.000	3.000
Jun	3.000	0.0	0.0	0.000	0.000	0.000	3.000	0.0	2.020	3.000
Jul	1.124	0.0	0.0	3.000	1.124	0.000	3.000	0.0	3.000	0.000
Aug	3.000	0.0	0.0	1.827	0.000	0.655	0.000	0.0	3.000	3.000
Sep	1.363	0.0	0.0	3.000	0.000	3.000	0.000	0.0	3.000	1.363
Oct	0.000	0.0	0.0	3.000	0.000	3.000	3.000	0.0	0.000	2.955
Nov	3.000	0.0	0.0	2.056	0.000	1.113	3.000	0.0	0.000	3.000
Dec	3.000	3.0	0.0	0.428	3.000	0.000	0.000	0.0	0.000	3.000

This chart summarizes the month to month change in funds with ‘unstable’ changes of over 1 million highlighted in orange:

platform	Print	TV	SEO	AdWords	Facebook	LinkedIn	Instagram	Snapchat	Twitter	Email
Feb-Jan	0.000000	0.000000	0.000000	1.062167	3.000000	0.000000	-2.666667	0.000000	1.791000	-3.000000
Mar-Feb	-3.000000	0.000000	0.000000	0.604500	-3.000000	3.000000	1.389648	0.000000	1.209000	0.000000
Apr-Mar	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.610352	0.000000	-1.403144	0.000000
May-Apr	1.804100	0.000000	0.000000	-3.000000	0.000000	-3.000000	0.000000	0.000000	1.403144	3.000000
Jun-May	1.195900	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	-0.979828	0.000000
Jul-Jun	-1.876223	0.000000	0.000000	3.000000	1.123777	0.000000	0.000000	0.000000	0.979828	-3.000000
Aug-Jul	1.876223	0.000000	0.000000	-1.172706	-1.123777	0.654588	-3.000000	0.000000	0.000000	3.000000
Sep-Aug	-1.637067	0.000000	0.000000	1.172706	0.000000	2.345412	0.000000	0.000000	0.000000	-1.637067
Oct-Sep	-1.362933	0.000000	0.000000	0.000000	0.000000	0.000000	3.000000	0.000000	-3.000000	1.592543
Nov-Oct	3.000000	0.000000	0.000000	-0.943579	0.000000	-1.887158	0.000000	0.000000	0.000000	0.044525
Dec-Nov	0.000000	3.000000	0.000000	-1.628470	3.000000	-1.112842	-3.000000	0.000000	0.000000	0.000000

Additionally, we can see the overall funds growth based on the optimized allocation based on the 2 ROI assumptions, where the constant ROI assumption would give us an overall higher return at the end of the 12 month period.

	constant ROI	variable ROI
Jan	10.228000	10.186500
Feb	10.460446	10.389648
Mar	10.697425	10.596856
Apr	10.938630	10.804100
May	11.184056	11.020172
Jun	11.433777	11.247555
Jul	11.687868	11.481882
Aug	11.946405	11.725865
Sep	12.209467	11.955475
Oct	12.477133	12.169263
Nov	12.749483	12.427951
Dec	13.026599	12.686368