hw1

August 26, 2018

1 GTX Deterministic Optimization - Week 1 Homework

1.0.1 Question 1

Consider the following optimization problem:

$$min\{x^2 + y^2 : x + 3y \geq 6, x + y \geq 4, x \geq 0, y \geq 0\}$$

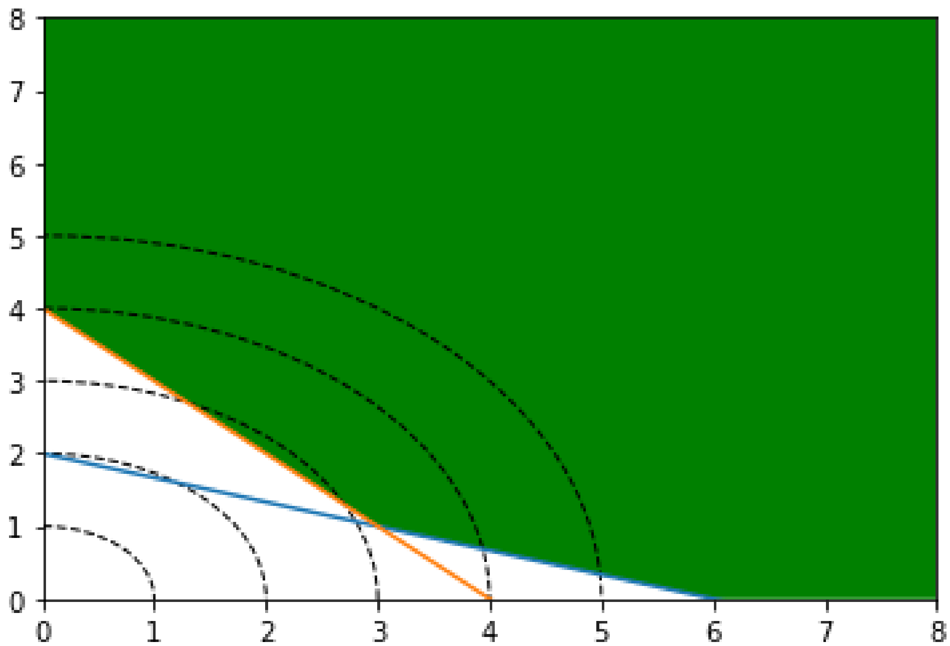
Plot the feasible region of this problem with the feasible area shaded. Draw (in dashed lines) the contours of the objective function. Based on your drawing, guess an optimal solution and the optimal objective value of this problem.

Answer We can see from the constraints that the solution is bound in the quadrant where both x and y are positive. The solution is further bound below by 2 lines that intersect at $(x, y) = (3, 1)$. The resulting feasible reagion is shaded in green in the chart below.

Regarding a feasible solution, it will exist where the objective function (which is a circle) intersects the feasible region constructed from the constraints. This will occur where constraints intersect (i.e. at one of the corners of the feasible region). In this case, it appears to be at $(3, 1)$ which yields an objective function value of 10.

In [18]:
```
#plot the solution using matplotlib
import matplotlib.pyplot as plt
import numpy as np

# x-values for our plot
x = np.arange(0, 8, 0.1)

# the constraints to plot
y1 = 2 - x / 3.
y2 = 4 - x

# plot the constraints
plt.xlim(0, 8)
plt.ylim(0, 8)
plt.plot(x, y1, x, y2)

# fill in the feasable region (using a polygon)
xp = [3, 6, 8, 8, 0, 0]
```

```
yp = [1, 0, 0, 8, 8, 4]
plt.fill(xp ,yp, 'g')

# plot objective function values
for i in range(1,6):
    circle = plt.Circle((0, 0), i, fill=False, ls='dashed')
    plt.gca().add_patch(circle)
```



1.0.2 Question 2

Recall the maximum volume box from Module 1, Lesson 1. Solve the following problem using basic calculus:

$$max\{x(12x)^2 : 0 \leq x \leq \frac{1}{2}\}$$

What is the optimal solution and the optimal objective value?

Answer By plugging in values, we can see that the objective function is 0 at the endpoints of the constraint (at 0 and at $\frac{1}{2}$). Using calculus we take the derivative of the objective function and determine where that equals 0. The first derivative of the objective function yields the polynomial $12x^2 - 8x + 1$. The roots of this polynomial are $x = \frac{1}{6}$ and $x = \frac{1}{2}$.

As already stated, the objective function has value 0 at $x = \frac{1}{2}$. At $x = \frac{1}{6}$ the objective function has value $\frac{2}{27}$. We can now use the second derivative test to determine minimums and maxiumums.

The second derivative of the objective function is $24x - 8$. The second derivative is positive $x = \frac{1}{2}$ indicating a minimum. The second derivative is negative at $x = \frac{1}{6}$ indicating a maximum.

Therefore we conclude that the objective as a maximum at $x = \frac{1}{6}$ with value $\frac{2}{27}$.

1.0.3 Question 3

Recall the portfolio optimization problem solved in Module 2, Lesson 3. Collect the prices of MSFT, V, and WMT from the last 24 months. Use the code used in the lesson (this will be provided) to solve the exact same portfolio problem using the new data. Compare and contrast your solution to the one in the lesson.

Answer Below we use the provided code with the data for the last 24 months (data was obtained from Yahoo Finance using "Adjusted Close." Note that the code was modified slightly to work with the most recent version of CVXPY.

The following values were yielded in the lecture.

- MSFT: Exp ret = 0.02461117173526473, Risk = 0.05804007272464832
- V: Exp ret = 0.018237255482436765, Risk = 0.04280681304387474
- WMT: Exp ret = 0.009066433484915428, Risk = 0.04446075444703942

Optimal Portfolio - MSFT = 0.5828274876507884 - V = 0.20430730337676578 - WMT = 0.21286520897225722

- Exp ret = 0.020000000137615822
- Risk = 0.03825552318041397

In [1]:
```
"""
Portfolio optimization with CVXPY
See examples at http://cvxpy.org
Author: Shabbir Ahmed
"""

import pandas as pd
import numpy as np
from cvxpy import *

# read monthly_prices.csv
mp = pd.read_csv("monthly_prices_last_24.csv",index_col=0)
mr = pd.DataFrame()

# compute monthly returns
for s in mp.columns:
    date = mp.index[0]
    pr0 = mp[s][date]
    for t in range(1,len(mp.index)):
        date = mp.index[t]
        pr1 = mp[s][date]
        ret = (pr1-pr0)/pr0
        mr.set_value(date,s,ret)
        pr0 = pr1

# get symbol names
symbols = mr.columns
```

```python
    # convert monthly return data frame to a numpy matrix
    return_data = mr.as_matrix().T

    # compute mean return
    r = np.asarray(np.mean(return_data, axis=1))

    # covariance
    C = np.asmatrix(np.cov(return_data))

    # print out expected return and std deviation
    print("----------------------")
    for j in range(len(symbols)):
        print('{}: Exp ret = {}, Risk = {}'.format(symbols[j], r[j], C[j,j]**0.5))

    # set up optimization model
    n = len(symbols)
    x = Variable(n)
    req_return = 0.02
    ret = r.T*x
    risk = quad_form(x, C)
    prob = Problem(Minimize(risk),
                   [sum(x) == 1, ret >= req_return,
                    x >= 0])

    # solve problem and write solution
    prob.solve()
    print("----------------------")
    print("Optimal portfolio")
    print("----------------------")
    for s in range(len(symbols)):
        print('x[{}] = {}'.format(symbols[s], x.value[s])) #, x.value[s,0]
    print("----------------------")
    print('Exp ret = {}'.format(ret.value))
    print('risk    = {}'.format((risk.value)**0.5))
    print("----------------------")
```

```
----------------------
MSFT: Exp ret = 0.030149235355715575, Risk = 0.03581763482246733
V: Exp ret = 0.02557783152238455, Risk = 0.03421173516891634
WMT: Exp ret = 0.015791522346555297, Risk = 0.060577265811611894
----------------------
Optimal portfolio
----------------------
x[MSFT] = 0.44690890851054205
x[V] = 0.5433380521576049
x[WMT] = 0.009753039331784225
----------------------
```

```
Exp ret = 0.0275253863615928
risk    = 0.030608563044001582
----------------------
```

```
/home/robert/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:23: FutureWarning: set_
/home/robert/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:30: FutureWarning: Meth
```

Answer Continued Using the data for the last 24 months, we see the following results for the optimal portfolio.

Stock / Measure	Lecture	Last 24 months
MSFT	58.28%	44.69%
V	20.43%	54.33%
WMT	21.29%	.975%
Exp ret	2.0%	2.75%
Risk	3.82%	3.06%

The result is that the optimized portfolio using the last 24 months of data yields a portfolio with a higher return and lower risk. Furthermore, the optimized portfolio using the last 24 months of data effectively excludes WMT.

Regarding the difference between the optimized portfolios, this suggests that the data is time dependent and hence the often stated "past performance may not indicate future returns."

This model may be sufficient if we are concerned with using current data to assess our current portfolio. In other words, we can use this model to help ensure our rate of return is optimized for our current risk level given the current proportions of stock in our portfolio.

However, if we are concerned with making predictions about future performance, then it may be necessary to build a model that incorporates more information (such as time).