In [1]:

```
#-*-coding:utf-8-*-
```

In [29]:

```
import urllib2 as url2
import scipy
import numpy as np
import matplotlib.pyplot as plt
from scipy import stats
import cvxopt as opt
from cvxopt import blas,solvers
import csv
import pandas as pd
%matplotlib inline
```

First, I download the S&P 500 Index from Yahoo and calculate the monthly return rate of it.

In [3]:

```
spurl='http://real-chart.finance.yahoo.com/\
table.csv?s=%5EGSPC&a=00&b=3&c=2005&d=05&e=18&f=2015&g=m&ignore=.csv'
f=url2.urlopen(spurl)
```

In [4]:

```
s=f.read()
f.close()
s=s.split('\n')
```

In [5]:

```
N1=len(s)
sp=np.zeros(N1-2)
for i in range(1,N1-1):
    t=s[i]
    t=t.split(',')
    sp[i-1]=t[6]
sp=sp[::-1]
sp_r=np.diff(np.log(sp))
```

Then, program loads the list of S&P 500 Composite that I stored advancedly in the 'S&P 500 Composite.csv'

In [6]:

```
f=open('S&P 500 Composite.csv','r')
com_list=f.read()
f.close()
com_list=com_list.split('\r')
ticker=[]
name=[]
for k in range(len(com_list)):
    temp=com_list[k]
    temp=temp.split(',')
    ticker.append(temp[0])
    name.append(temp[1])
ticker=ticker[1:]
name=name[1:]
```

Define a function in order to download and calculate the autocorrelation between S&P 500 Index and one specific composite stock. What's more, return an 'alert' in order to remind users to take care of stocks that don't have long enough history data.

In [297]:

```
def Composite_cov(ticker,i):
    url='http://real-chart.finance.yahoo.com/\
    table.csv?s=%s&a=00&b=3&c=2005&d=05&e=18&f=2015&g=m&ignore=.csv'\
    %ticker[i]
    f=url2.urlopen(url)
    stock=f.read()
    f.close()
    stock=stock.split('\n')
    N2=len(stock)
    stock_p=np.zeros(N2-2)
    for j in range(1,N2-1):
        t=stock[j]
        t=t.split(',')
        stock_p[j-1]=t[6]

    stock_p=stock_p[::-1]
    stock_r=np.diff(np.log(stock_p))
    N3=np.min((len(sp_r),len(stock_r)))
    if len(stock_r)<len(sp_r):
        alert=True
    else:
        alert=False
    cov=scipy.cov(stock_r[-N3:],sp_r[-N3:])
    row=cov[0][1]/(np.std(stock_r[-N3:])*np.std(sp_r[-N3:]))
    return row, alert
```

Download and calculate the autocorrelation between S&P 500 Composite stocks and S&P Index.

In [77]:

```
N4=len(ticker)
total_cov=np.zeros((N4,3))
print 'Percentage of downloading data:'

for i in range(N4):
    total_cov[i][1],total_cov[i][2]=Composite_cov(ticker,i)
    total_cov[i][0]=int(i)
    rate=(i+1)/502.*100
    print '\r%d%%' %rate,
```

```
 Percentage of downloading data:
100%
```

In [78]:

```
with open('total_cov.csv','wb') as csvfile:
    writers=csv.writer(csvfile,dialect='excel')
    for line in total_cov:
        writers.writerow(line)
```

In [79]:

```
readers=pd.read_csv('total_cov.csv',index_col=False,\
                    names=['No.',\
                           'Correlationship',\
                           'Is Time Series too short?\
                           (\'1\' means too short)'])
readers.head(10)
```

Out[79]:

	No.	Correlationship	Is Time Series too short?('1' means too short)
0	0	0.676152	0
1	1	0.773215	0
2	2	0.236900	1
3	3	0.543129	0
4	4	0.670860	1
5	5	0.530956	0
6	6	0.341026	0
7	7	0.616556	0
8	8	0.640697	0
9	9	0.351619	0

Recognize the composites that don't have ten-year length of history data.

In [80]:

```
special_id=total_cov[total_cov[:,2]==1][:,0]
special_ticker=[]
special_stock=[]
for i in range(len(special_id)):
    special_ticker.append(ticker[int(special_id[i])])
    special_stock.append(name[int(special_id[i])])
print special_ticker
```

```
['AAL', 'ABBV', 'ADT', 'ALLE', 'AMP', 'AVGO', 'CBS', 'CF', 'CMG',
'DAL', 'DFS', 'DG', 'DISCA', 'DISCK', 'DLPH', 'DPS', 'EXPE', 'F
B', 'FSLR', 'GM', 'GOOG', 'HBI', 'HCA', 'ICE', 'KMI', 'KORS', 'KR
FT', 'LO', 'LYB', 'MA', 'MJN', 'MNK', 'MPC', 'NAVI', 'NLSN', 'NWS
A', 'PM', 'PSX', 'QEP', 'SE', 'SNI', 'TDC', 'TEL', 'TRIP', 'TWC',
'UA', 'V', 'VIAB', 'WU', 'WYN', 'XYL', 'ZTS']
```

As shown above, these stocks don't have enough data for us to analysis. Some of them are lacking in data because they are newly appeared on the stock market, such as Facebook. Others are lacking in data because they experienced some events nearly before. For example, Kraft Foods Group(KRFT) only has a three-year history data, though it firstly appeared on the stock market in 1924. What KRFT experienced is that KRFT was once divided into two separate companies, and KRFT had to be listed again.

After finding out the unfitted data, we just delete them.

In [81]:

```
total_covx=total_cov[total_cov[:,2]==0]
newid=total_covx[:,0]
namex=[]
tickerx=[]
for i in range(len(newid)):
    namex.append(name[int(newid[i])])
    tickerx.append(ticker[int(newid[i])])
```

Next step is to find out stocks that have lowest correlationship with S&P 500 Index. I will only choose first six ones to construct my portfolio.

In [82]:

```
myport=np.argsort(total_covx[:,1])
myport=map(int,myport)
my_port=[]
for i in range(len(myport)):
    my_port.append(namex[myport[i]])
```

In [83]:

```
ticker_sorted=[]
for i in range(0,6):
    ticker_sorted.append(tickerx[myport[i]])
```

Final result of my components in my portfolio.

Final result of my components in my portfolio.

In [84]:

```
print ticker_sorted
```

```
['FDO', 'NEM', 'MNST', 'VRTX', 'NFLX', 'ED']
```

In [85]:

```
print my_port[0:6]
```

```
['Family Dollar Stores', 'Newmont Mining Corp. (Hldg. Co.)', 'Mon
ster Beverage', 'Vertex Pharmaceuticals Inc', 'Netflix Inc.', 'Co
nsolidated Edison']
```

Finally I choose 'Family Dollar Stores', 'Newmont Mining Corp. (Hldg. Co.)', 'Monster Beverage', 'Vertex Pharmaceuticals Inc', 'Netflix Inc.', 'Consolidated Edison' as components of my portfolio.

After determined which stocks to be selected for portfolio, I have to decide the detailed weight of the portfolio. In order to do so, I have to collect the full statistics of these specific stocks.

In [86]:

```
def Download_data(ticker_sorted,i):
    for j in range(i):
        url='http://real-chart.finance.yahoo.com/\
        table.csv?s=%s&a=08&b=7&c=2005\
        &d=04&e=18&f=2015&g=m&ignore=.csv'\
        %ticker_sorted[j]
        f=url2.urlopen(url)
        data=f.read()
        f.close()
        g=file('%s.csv'%ticker_sorted[j],'w')
        g.write(data)
        g.close
```

In [87]:

```
Download_data(ticker_sorted,6)
```

A module for collecting data online and save them into a csv file.

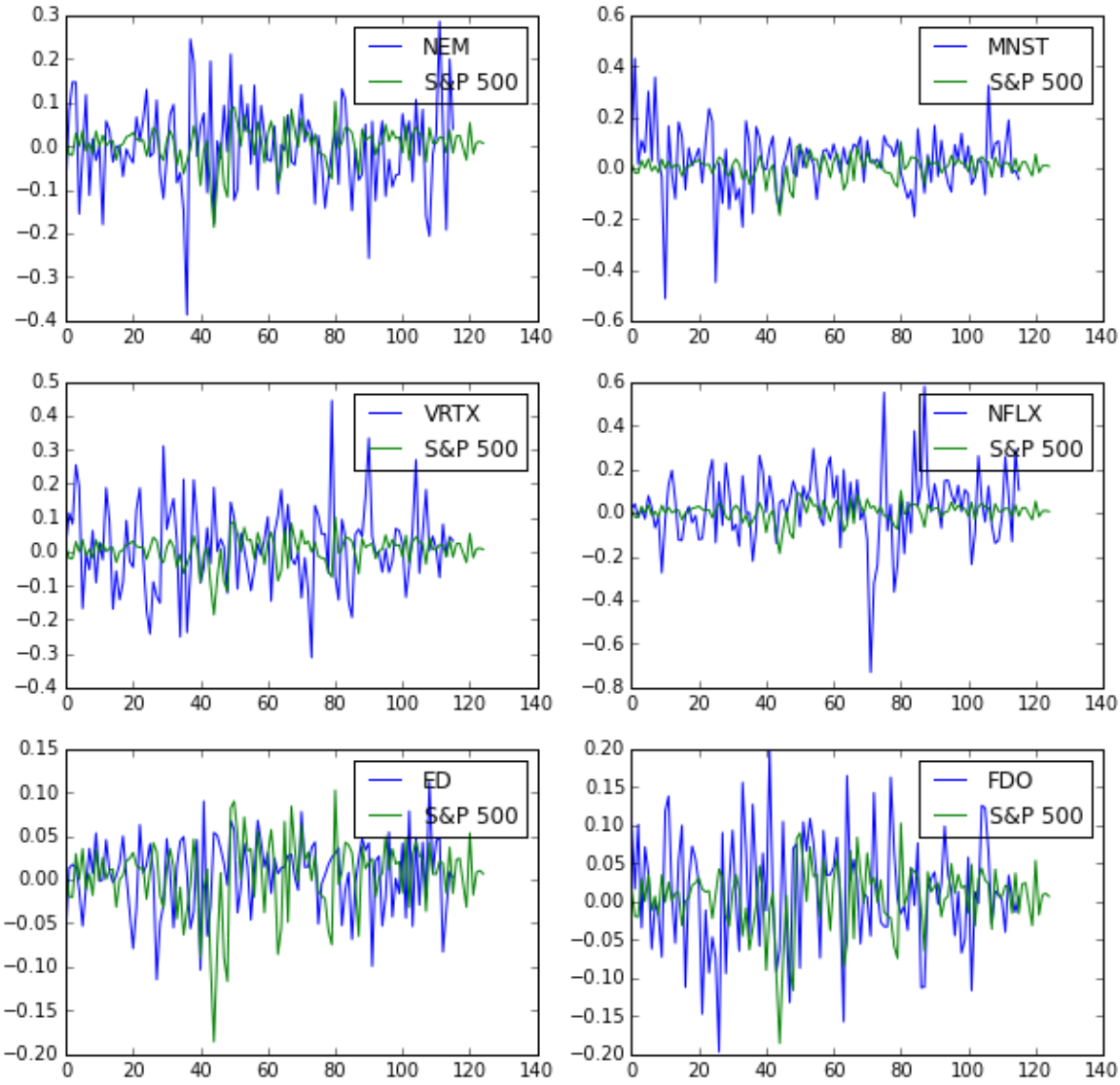
In [89]:

```
def read_data(ticker_name):
    f=file('%s.csv'%ticker_name,'r')
    data=f.read()
    f.close()
    data=data.split('\n')
    N=len(data)
    data=data[1:N-1]


    stock_p=np.zeros(len(data))
    for i in range(len(data)):
        temp=data[i]
        temp=temp.split(',')
        stock_p[i]=temp[6]
    stock_p=stock_p[::-1]
    stock_r=np.diff(np.log(stock_p))
    return stock_r
```

Plot rate of return of selected stocks, And plot relative S&P 500 Index on the same figure.

In [401]:

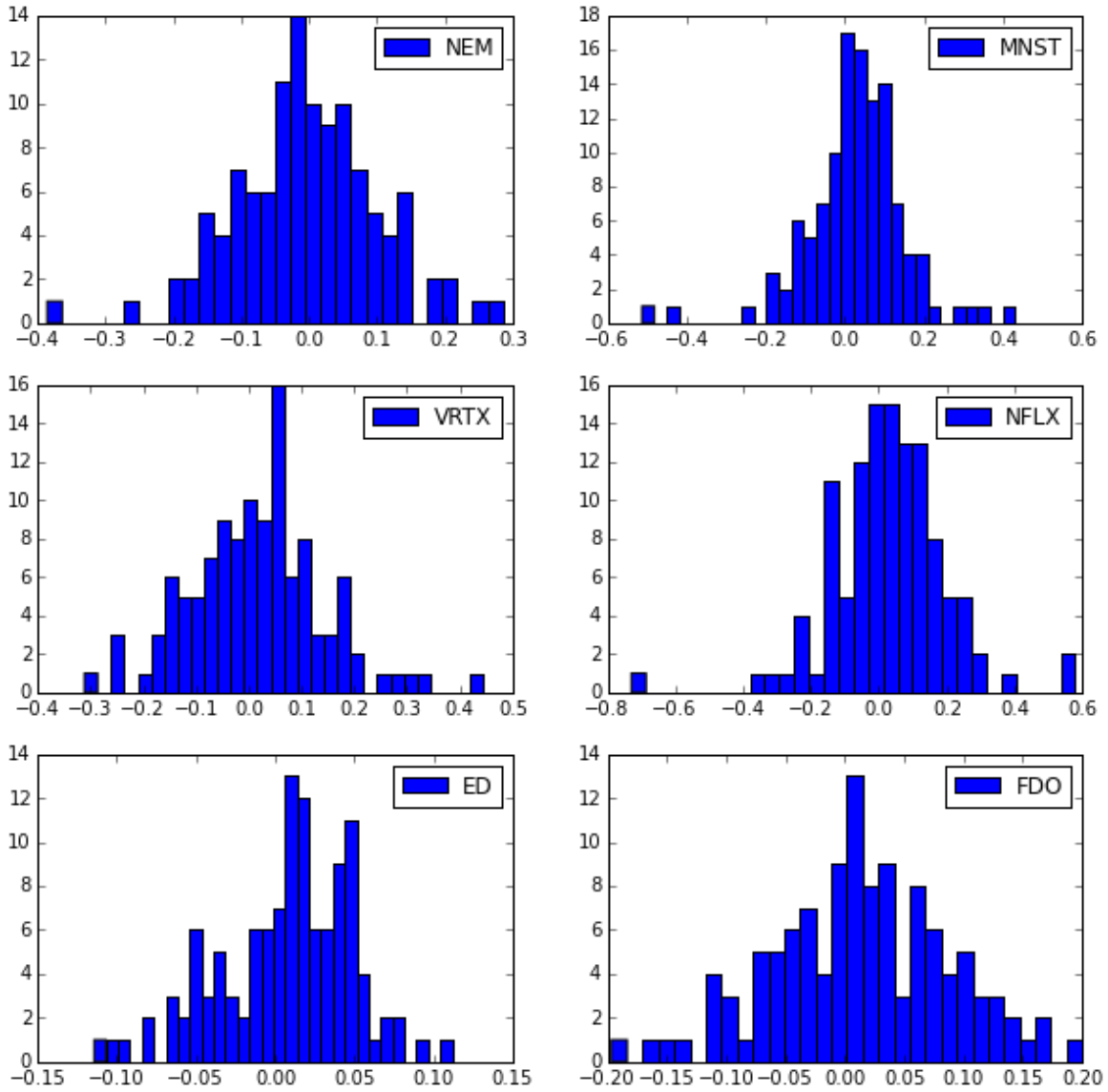
```
plt.figure(figsize=(10,10))
for i in range(6):

    plt.subplot(3,2,i)
    plt.plot(read_data('%s'%ticker_sorted[i]),label='%s'%ticker_sorted[i]);
    plt.plot(sp_r,label='S&P 500');
    plt.legend()
```



Plot histogram of different composites.

In [91]:

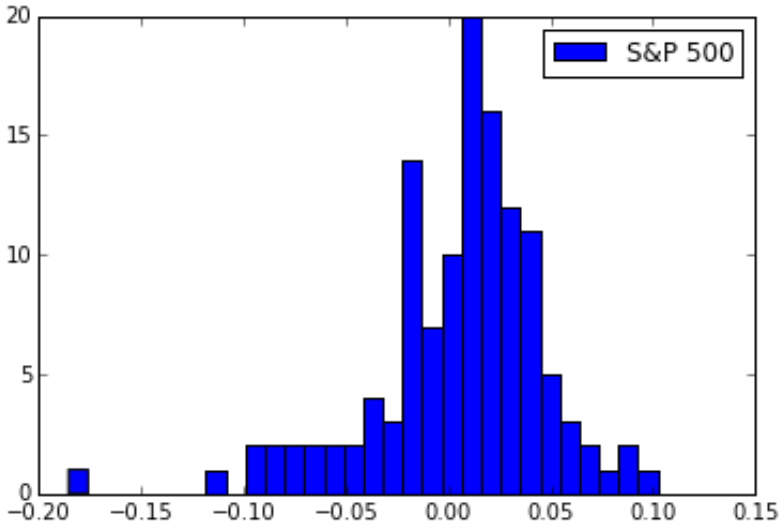
```
plt.figure(figsize=(10,10))
for i in range(6):

    plt.subplot(3,2,i)
    plt.hist(read_data('%s'%ticker_sorted[i]),bins=30,label='%s'\
             %ticker_sorted[i]);
    plt.legend()
```



The histogram of S&P 500 Index.

In [93]:

```
plt.hist(sp_r,bins=30,label='S&P 500');
plt.legend();
```



After collecting enough data, next step is to calculate the weight of composites, so first I should simulate random weights of the portfolio of which the total sum equals one.

In [94]:

```
inv=np.linalg.inv
f=open('%s.csv'%ticker_sorted[0],'r')
temp=f.read()
f.close()
temp=temp.split('\n')
N7=len(temp)-2

N6=len(ticker_sorted)
P=np.zeros((N7,N6))
R=np.zeros((N7-1,N6))
for i in range(N6):
    f=open('%s.csv'%ticker_sorted[i],'r')
    r_i=f.read()
    f.close()
    r_i=r_i.split('\n')
    r_i=r_i[0:-1]

    for j in range(1,N7+1):
        temp_r=r_i[j]
        temp_s=temp_r.split(',')
        temp_ar=temp_s[6]
        P[j-1,i]=temp_ar
P=P[::-1,:]
for i in range(N6):
    for j in range(N7-1):
        R[j,i]=np.log(P[j+1,i])-np.log(P[j,i])

with open('Returns.csv','wb') as csvfile:
    writers=csv.writer(csvfile, dialect='excel')
    for i in range(N7-1):
        writers.writerows([R[i,:]])

cov_matrix=np.cov(R,rowvar=0)
rate_of_return=np.zeros(N6)
for i in range(N6):
    rate_of_return[i]=np.mean(R[:,i])
```

In [95]:

```
with open('S_P.csv','wb') as csvfile:
    writers=csv.writer(csvfile, dialect='excel')
    for line in sp_r[-(N7-1):]:
        writers.writerow([line])
```

Calculate β for every asset.

In [408]:

```
beta=np.zeros(N6)
intercept=np.zeros(N6)
r_value=np.zeros(N6)
p_value=np.zeros(N6)
std_err=np.zeros(N6)
for i in range(N6):
    beta[i],intercept[i],r_value[i],p_value[i],std_err[i]\
    =stats.linregress(R[:,i],sp_r[-(N7-1):])

for j in range(len(beta)):
    print beta[j],'\n'
```

```
0.0103498184527

0.00458633958868

-0.0174150657763

0.0212816692217

0.00594830704975

-0.0365902763738
```

Calculate the efficient frontier by using a convex optimization package called 'cvxopt'. No arbitrage is allowed.

In [285]:

```
import cvxopt as opt
from cvxopt import blas, solvers
```

In [286]:

```
n_assets=6
n_obs=10**3
```

In [293]:

```
def rand_weights(n):
    ''' Produces n random weights that sum to 1 '''
    k = np.random.rand(n)
    return k / sum(k)
```

In [288]:

```
def random_portfolio(returns):

    p = np.asmatrix(np.mean(returns, axis=1))
    w = np.asmatrix(rand_weights(returns.shape[0]))
    C = np.asmatrix(np.cov(returns))

    mu = w * p.T
    sigma = np.sqrt(w * C * w.T)

    if sigma > 2:
        return random_portfolio(returns)
    return mu, sigma
```

In [404]:

```
n_portfolios = 1000
means, stds = np.column_stack([
    random_portfolio(R.T)
    for _ in xrange(n_portfolios)
])

def optimal_portfolio(returns):
    n = len(returns)
    returns = np.asmatrix(returns)

    N = 100
    mus = [10**(6.0 * t/N - 1.0) for t in range(N)]


    S = opt.matrix(np.cov(returns))
    pbar = opt.matrix(np.mean(returns, axis=1))


    G = -opt.matrix(np.eye(n))
    h = opt.matrix(0.0, (n ,1))
    A = opt.matrix(1.0, (1, n))
    b = opt.matrix(1.0)


    portfolios = [solvers.qp(mu*S, -pbar, G, h, A, b)['x']
                  for mu in mus]

    returns = [blas.dot(pbar, x) for x in portfolios]
    risks = [np.sqrt(blas.dot(x, S*x)) for x in portfolios]

    m1 = np.polyfit(returns, risks, 2)
    x1 = np.sqrt(m1[2] / m1[0])

    wt = solvers.qp(opt.matrix(x1 * S), -pbar, G, h, A, b)['x']
    return np.asarray(wt), returns, risks

weights, returns, risks = optimal_portfolio(R.T);
```

```
     pcost       dcost       gap    pres   dres
 0: -1.4686e-02 -1.0269e+00  1e+00  2e-16  3e+00
 1: -1.4905e-02 -3.6848e-02  2e-02  6e-17  6e-02
```

```
 2: -2.3279e-02 -2.6584e-02  3e-03  3e-16  5e-03
 3: -2.6408e-02 -2.6469e-02  6e-05  2e-16  2e-05
 4: -2.6443e-02 -2.6443e-02  6e-07  7e-17  2e-07
 5: -2.6443e-02 -2.6443e-02  6e-09  2e-16  2e-09
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -1.4665e-02 -1.0268e+00  1e+00  1e-16  3e+00
 1: -1.4882e-02 -3.6779e-02  2e-02  6e-17  6e-02
 2: -2.3174e-02 -2.6508e-02  3e-03  2e-16  5e-03
 3: -2.6320e-02 -2.6390e-02  7e-05  1e-16  9e-06
 4: -2.6356e-02 -2.6357e-02  7e-07  1e-16  9e-08
 5: -2.6356e-02 -2.6356e-02  7e-09  6e-17  9e-10
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -1.4641e-02 -1.0267e+00  1e+00  1e-16  3e+00
 1: -1.4857e-02 -3.6699e-02  2e-02  9e-17  6e-02
 2: -2.3035e-02 -2.6448e-02  3e-03  1e-16  6e-03
 3: -2.6217e-02 -2.6301e-02  8e-05  4e-16  6e-18
 4: -2.6258e-02 -2.6259e-02  8e-07  1e-16  1e-17
 5: -2.6258e-02 -2.6258e-02  8e-09  1e-16  5e-18
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -1.4612e-02 -1.0266e+00  1e+00  6e-17  3e+00
 1: -1.4827e-02 -3.6608e-02  2e-02  1e-16  6e-02
 2: -2.2880e-02 -2.6377e-02  3e-03  2e-16  6e-03
 3: -2.6095e-02 -2.6198e-02  1e-04  3e-16  1e-17
 4: -2.6146e-02 -2.6147e-02  1e-06  1e-16  7e-18
 5: -2.6147e-02 -2.6147e-02  1e-08  1e-16  9e-18
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -1.4580e-02 -1.0265e+00  1e+00  8e-17  3e+00
 1: -1.4794e-02 -3.6504e-02  2e-02  1e-16  6e-02
 2: -2.2717e-02 -2.6294e-02  4e-03  1e-16  6e-03
 3: -2.5957e-02 -2.6079e-02  1e-04  2e-17  7e-18
 4: -2.6019e-02 -2.6020e-02  1e-06  2e-16  1e-17
 5: -2.6019e-02 -2.6019e-02  1e-08  1e-16  1e-17
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -1.4543e-02 -1.0264e+00  1e+00  8e-17  3e+00
 1: -1.4755e-02 -3.6384e-02  2e-02  6e-17  6e-02
 2: -2.2556e-02 -2.6196e-02  4e-03  1e-16  6e-03
 3: -2.5802e-02 -2.5941e-02  1e-04  2e-16  2e-18
 4: -2.5873e-02 -2.5875e-02  1e-06  2e-16  1e-17
 5: -2.5874e-02 -2.5874e-02  1e-08  1e-16  1e-17
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -1.4500e-02 -1.0263e+00  1e+00  6e-17  3e+00
 1: -1.4711e-02 -3.6266e-02  2e-02  5e-17  6e-02
 2: -2.2368e-02 -2.6080e-02  4e-03  2e-16  6e-03
 3: -2.5625e-02 -2.5781e-02  2e-04  3e-16  6e-18
 4: -2.5707e-02 -2.5709e-02  2e-06  2e-16  9e-18
 5: -2.5708e-02 -2.5708e-02  2e-08  1e-16  5e-18
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -1.4451e-02 -1.0262e+00  1e+00  2e-16  3e+00
 1: -1.4660e-02 -3.6191e-02  2e-02  1e-16  6e-02
 2: -2.2128e-02 -2.5945e-02  4e-03  9e-17  6e-03
 3: -2.5420e-02 -2.5599e-02  2e-04  6e-17  7e-18
```

```
 4: -2.5517e-02 -2.5519e-02  2e-06  2e-16  6e-18
 5: -2.5518e-02 -2.5518e-02  2e-08  1e-16  1e-17
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -1.4395e-02 -1.0261e+00  1e+00  3e-16  3e+00
 1: -1.4602e-02 -3.6106e-02  2e-02  1e-16  6e-02
 2: -2.1866e-02 -2.5787e-02  4e-03  2e-16  7e-03
 3: -2.5186e-02 -2.5389e-02  2e-04  1e-16  7e-18
 4: -2.5300e-02 -2.5302e-02  2e-06  2e-16  1e-17
 5: -2.5301e-02 -2.5301e-02  2e-08  1e-16  9e-18
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -1.4331e-02 -1.0260e+00  1e+00  8e-17  3e+00
 1: -1.4536e-02 -3.6009e-02  2e-02  8e-17  6e-02
 2: -2.1581e-02 -2.5602e-02  4e-03  1e-16  7e-03
 3: -2.4918e-02 -2.5146e-02  2e-04  2e-16  9e-18
 4: -2.5050e-02 -2.5053e-02  2e-06  4e-18  1e-17
 5: -2.5052e-02 -2.5052e-02  2e-08  6e-17  5e-18
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -1.4258e-02 -1.0259e+00  1e+00  3e-17  3e+00
 1: -1.4459e-02 -3.5896e-02  2e-02  9e-17  6e-02
 2: -2.1272e-02 -2.5386e-02  4e-03  7e-17  7e-03
 3: -2.4610e-02 -2.4867e-02  3e-04  1e-16  7e-18
 4: -2.4765e-02 -2.4767e-02  3e-06  3e-17  7e-18
 5: -2.4766e-02 -2.4766e-02  3e-08  1e-16  8e-18
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -1.4173e-02 -1.0258e+00  1e+00  2e-16  3e+00
 1: -1.4372e-02 -3.5768e-02  2e-02  6e-17  6e-02
 2: -2.0937e-02 -2.5135e-02  4e-03  1e-16  7e-03
 3: -2.4255e-02 -2.4545e-02  3e-04  2e-16  7e-18
 4: -2.4437e-02 -2.4440e-02  3e-06  1e-16  2e-17
 5: -2.4439e-02 -2.4439e-02  3e-08  2e-16  6e-18
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -1.4077e-02 -1.0256e+00  1e+00  8e-17  3e+00
 1: -1.4272e-02 -3.5620e-02  2e-02  1e-16  6e-02
 2: -2.0576e-02 -2.4842e-02  4e-03  2e-16  7e-03
 3: -2.3848e-02 -2.4174e-02  3e-04  2e-16  4e-18
 4: -2.4060e-02 -2.4064e-02  4e-06  2e-16  6e-18
 5: -2.4063e-02 -2.4063e-02  4e-08  1e-16  9e-18
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -1.3966e-02 -1.0255e+00  1e+00  1e-16  3e+00
 1: -1.4157e-02 -3.5450e-02  2e-02  6e-17  6e-02
 2: -2.0187e-02 -2.4504e-02  4e-03  1e-16  7e-03
 3: -2.3381e-02 -2.3747e-02  4e-04  1e-16  5e-18
 4: -2.3628e-02 -2.3633e-02  4e-06  1e-16  6e-18
 5: -2.3631e-02 -2.3632e-02  4e-08  1e-16  7e-18
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -1.3839e-02 -1.0253e+00  1e+00  1e-16  3e+00
 1: -1.4026e-02 -3.5255e-02  2e-02  7e-17  6e-02
 2: -1.9769e-02 -2.4114e-02  4e-03  2e-16  7e-03
 3: -2.2847e-02 -2.3257e-02  4e-04  2e-16  1e-17
 4: -2.3132e-02 -2.3138e-02  6e-06  6e-17  1e-17
 5: -2.3137e-02 -2.3137e-02  6e-08  1e-16  6e-18
```

```
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -1.3694e-02 -1.0251e+00  1e+00  3e-17  3e+00
 1: -1.3876e-02 -3.5030e-02  2e-02  8e-17  6e-02
 2: -1.9318e-02 -2.3666e-02  4e-03  1e-16  7e-03
 3: -2.2242e-02 -2.2696e-02  5e-04  2e-16  1e-17
 4: -2.2560e-02 -2.2570e-02  1e-05  2e-16  9e-18
 5: -2.2569e-02 -2.2569e-02  1e-07  1e-16  1e-17
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -1.3528e-02 -1.0248e+00  1e+00  8e-17  3e+00
 1: -1.3705e-02 -3.4773e-02  2e-02  1e-16  6e-02
 2: -1.8835e-02 -2.3155e-02  4e-03  7e-17  7e-03
 3: -2.1564e-02 -2.2056e-02  5e-04  1e-16  5e-18
 4: -2.1900e-02 -2.1918e-02  2e-05  1e-16  6e-18
 5: -2.1917e-02 -2.1917e-02  2e-07  1e-17  9e-18
 6: -2.1917e-02 -2.1917e-02  2e-09  2e-16  3e-18
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -1.3338e-02 -1.0245e+00  1e+00  3e-16  3e+00
 1: -1.3509e-02 -3.4477e-02  2e-02  3e-16  6e-02
 2: -1.8316e-02 -2.2576e-02  4e-03  1e-16  7e-03
 3: -2.0816e-02 -2.1332e-02  5e-04  2e-16  1e-17
 4: -2.1137e-02 -2.1171e-02  3e-05  6e-17  6e-18
 5: -2.1168e-02 -2.1169e-02  6e-07  6e-17  5e-18
 6: -2.1169e-02 -2.1169e-02  6e-09  2e-16  4e-18
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -1.3121e-02 -1.0242e+00  1e+00  2e-16  3e+00
 1: -1.3285e-02 -3.4138e-02  2e-02  9e-17  6e-02
 2: -1.7761e-02 -2.1924e-02  4e-03  1e-16  7e-03
 3: -2.0009e-02 -2.0524e-02  5e-04  3e-17  8e-18
 4: -2.0276e-02 -2.0322e-02  5e-05  1e-16  6e-18
 5: -2.0305e-02 -2.0310e-02  5e-06  1e-16  5e-18
 6: -2.0309e-02 -2.0310e-02  3e-07  9e-17  5e-18
 7: -2.0310e-02 -2.0310e-02  3e-09  1e-16  4e-18
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -1.2873e-02 -1.0238e+00  1e+00  2e-16  3e+00
 1: -1.3030e-02 -3.3748e-02  2e-02  1e-16  6e-02
 2: -1.7168e-02 -2.1195e-02  4e-03  8e-17  7e-03
 3: -1.9164e-02 -1.9639e-02  5e-04  1e-16  7e-18
 4: -1.9371e-02 -1.9403e-02  3e-05  1e-16  6e-18
 5: -1.9382e-02 -1.9385e-02  3e-06  8e-17  4e-18
 6: -1.9383e-02 -1.9383e-02  4e-07  1e-16  5e-18
 7: -1.9383e-02 -1.9383e-02  1e-08  2e-16  1e-17
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -1.2589e-02 -1.0234e+00  1e+00  1e-16  3e+00
 1: -1.2740e-02 -3.3300e-02  2e-02  2e-16  6e-02
 2: -1.6538e-02 -2.0388e-02  4e-03  2e-16  6e-03
 3: -1.8307e-02 -1.8702e-02  4e-04  6e-17  5e-18
 4: -1.8484e-02 -1.8511e-02  3e-05  1e-16  8e-18
 5: -1.8494e-02 -1.8496e-02  1e-06  6e-17  4e-18
 6: -1.8494e-02 -1.8494e-02  2e-08  1e-16  5e-18
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -1.2266e-02 -1.0229e+00  1e+00  8e-17  3e+00
```

```
 1: -1.2409e-02 -3.2785e-02  2e-02  4e-17  6e-02
 2: -1.5871e-02 -1.9504e-02  4e-03  7e-17  6e-03
 3: -1.7440e-02 -1.7774e-02  3e-04  2e-16  8e-05
 4: -1.7632e-02 -1.7645e-02  1e-05  2e-16  1e-07
 5: -1.7635e-02 -1.7635e-02  2e-07  2e-16  1e-09
 6: -1.7635e-02 -1.7635e-02  2e-09  1e-16  1e-11
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -1.1898e-02 -1.0223e+00  1e+00  8e-17  3e+00
 1: -1.2033e-02 -3.2195e-02  2e-02  1e-16  6e-02
 2: -1.5170e-02 -1.8546e-02  3e-03  7e-17  5e-03
 3: -1.6530e-02 -1.6953e-02  4e-04  1e-16  4e-04
 4: -1.6783e-02 -1.6795e-02  1e-05  2e-16  1e-06
 5: -1.6787e-02 -1.6787e-02  1e-07  1e-16  1e-08
 6: -1.6787e-02 -1.6787e-02  1e-09  7e-17  1e-10
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -1.1479e-02 -1.0216e+00  1e+00  0e+00  3e+00
 1: -1.1607e-02 -3.1517e-02  2e-02  1e-16  6e-02
 2: -1.4438e-02 -1.7523e-02  3e-03  6e-17  5e-03
 3: -1.5701e-02 -1.6080e-02  4e-04  1e-16  4e-04
 4: -1.5931e-02 -1.5939e-02  8e-06  1e-16  2e-06
 5: -1.5935e-02 -1.5935e-02  8e-08  8e-17  2e-08
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -1.1002e-02 -1.0209e+00  1e+00  2e-16  3e+00
 1: -1.1123e-02 -3.0738e-02  2e-02  5e-17  6e-02
 2: -1.3679e-02 -1.6448e-02  3e-03  6e-17  4e-03
 3: -1.4922e-02 -1.5164e-02  2e-04  1e-16  4e-18
 4: -1.5060e-02 -1.5064e-02  4e-06  1e-16  5e-18
 5: -1.5063e-02 -1.5063e-02  4e-08  2e-16  5e-18
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -1.0461e-02 -1.0200e+00  1e+00  2e-16  3e+00
 1: -1.0575e-02 -2.9844e-02  2e-02  8e-17  6e-02
 2: -1.2898e-02 -1.5337e-02  2e-03  3e-17  4e-03
 3: -1.4011e-02 -1.4282e-02  3e-04  7e-17  3e-18
 4: -1.4146e-02 -1.4155e-02  9e-06  8e-17  5e-18
 5: -1.4153e-02 -1.4153e-02  9e-08  6e-17  4e-18
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -9.8468e-03 -1.0190e+00  1e+00  2e-16  3e+00
 1: -9.9561e-03 -2.8819e-02  2e-02  7e-17  5e-02
 2: -1.2044e-02 -1.4682e-02  3e-03  1e-16  5e-03
 3: -1.3044e-02 -1.3329e-02  3e-04  1e-16  4e-18
 4: -1.3175e-02 -1.3193e-02  2e-05  7e-17  6e-18
 5: -1.3189e-02 -1.3189e-02  7e-07  1e-16  5e-18
 6: -1.3189e-02 -1.3189e-02  7e-09  7e-17  4e-18
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -9.1512e-03 -1.0178e+00  1e+00  1e-16  3e+00
 1: -9.2582e-03 -2.7644e-02  2e-02  1e-16  5e-02
 2: -1.1185e-02 -1.3974e-02  3e-03  1e-16  6e-03
 3: -1.2056e-02 -1.2320e-02  3e-04  1e-16  7e-18
 4: -1.2164e-02 -1.2178e-02  1e-05  7e-17  6e-18
 5: -1.2168e-02 -1.2169e-02  4e-07  1e-16  1e-17
 6: -1.2168e-02 -1.2168e-02  4e-09  2e-16  4e-18
Optimal solution found.
```

```
     pcost       dcost       gap    pres   dres
 0: -8.3647e-03 -1.0165e+00  1e+00  8e-17  3e+00
 1: -8.4727e-03 -2.6297e-02  2e-02  1e-16  5e-02
 2: -1.0339e-02 -1.3068e-02  3e-03  6e-17  6e-03
 3: -1.1068e-02 -1.1275e-02  2e-04  3e-17  1e-05
 4: -1.1169e-02 -1.1173e-02  4e-06  1e-16  2e-07
 5: -1.1172e-02 -1.1172e-02  4e-08  5e-17  2e-09
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -7.4769e-03 -1.0160e+00  1e+00  3e-16  3e+00
 1: -7.5906e-03 -2.5821e-02  2e-02  8e-17  5e-02
 2: -9.4626e-03 -1.2046e-02  3e-03  1e-16  5e-03
 3: -1.0083e-02 -1.0289e-02  2e-04  1e-16  1e-04
 4: -1.0195e-02 -1.0200e-02  5e-06  1e-16  8e-07
 5: -1.0199e-02 -1.0199e-02  5e-08  2e-16  8e-09
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -6.4769e-03 -1.0163e+00  1e+00  2e-16  3e+00
 1: -6.6024e-03 -2.6035e-02  2e-02  8e-17  6e-02
 2: -8.5391e-03 -1.0982e-02  2e-03  2e-16  4e-03
 3: -9.1078e-03 -9.3482e-03  2e-04  8e-17  2e-04
 4: -9.2233e-03 -9.2318e-03  8e-06  7e-17  1e-06
 5: -9.2302e-03 -9.2303e-03  9e-08  1e-16  1e-08
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -5.3526e-03 -1.0165e+00  1e+00  4e-16  3e+00
 1: -5.4984e-03 -2.6233e-02  2e-02  2e-16  6e-02
 2: -7.5908e-03 -9.7912e-03  2e-03  1e-16  2e-03
 3: -8.1497e-03 -8.3452e-03  2e-04  1e-16  7e-18
 4: -8.2394e-03 -8.2486e-03  9e-06  1e-16  2e-18
 5: -8.2474e-03 -8.2475e-03  1e-07  2e-16  7e-18
 6: -8.2475e-03 -8.2475e-03  1e-09  2e-16  8e-18
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -4.0913e-03 -1.0168e+00  1e+00  1e-16  3e+00
 1: -4.2683e-03 -2.6401e-02  2e-02  8e-17  6e-02
 2: -6.5476e-03 -9.3212e-03  3e-03  2e-16  3e-03
 3: -7.1297e-03 -7.3895e-03  3e-04  3e-17  3e-18
 4: -7.2176e-03 -7.2347e-03  2e-05  1e-16  4e-18
 5: -7.2313e-03 -7.2318e-03  5e-07  1e-16  4e-18
 6: -7.2318e-03 -7.2318e-03  5e-09  7e-17  4e-18
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -2.6796e-03 -1.0171e+00  1e+00  3e-17  3e+00
 1: -2.9013e-03 -2.6521e-02  2e-02  6e-17  7e-02
 2: -5.4556e-03 -8.8951e-03  3e-03  2e-16  5e-03
 3: -6.0706e-03 -6.3906e-03  3e-04  6e-17  2e-18
 4: -6.1476e-03 -6.1717e-03  2e-05  1e-16  5e-18
 5: -6.1615e-03 -6.1641e-03  3e-06  1e-16  5e-18
 6: -6.1637e-03 -6.1638e-03  8e-08  8e-17  5e-18
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -1.1033e-03 -1.0173e+00  1e+00  1e-16  3e+00
 1: -1.3863e-03 -2.6570e-02  3e-02  1e-16  7e-02
 2: -4.3183e-03 -8.3172e-03  4e-03  3e-16  6e-03
 3: -4.9587e-03 -5.3198e-03  4e-04  4e-17  9e-18
 4: -5.0187e-03 -5.0447e-03  3e-05  3e-16  8e-18
 5: -5.0254e-03 -5.0285e-03  3e-06  6e-17  6e-18
```

```
 6: -5.0258e-03 -5.0260e-03  2e-07  1e-16  5e-18
 7: -5.0258e-03 -5.0258e-03  2e-09  1e-16  3e-18
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  6.5290e-04 -1.0175e+00  1e+00  6e-17  3e+00
 1:  2.8800e-04 -2.6518e-02  3e-02  1e-16  8e-02
 2: -3.1639e-03 -7.4168e-03  4e-03  1e-16  5e-03
 3: -3.7777e-03 -4.1308e-03  4e-04  6e-17  7e-18
 4: -3.8177e-03 -3.8392e-03  2e-05  8e-17  7e-18
 5: -3.8194e-03 -3.8203e-03  9e-07  1e-16  5e-18
 6: -3.8195e-03 -3.8195e-03  9e-09  9e-17  5e-18
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  2.6049e-03 -1.0176e+00  1e+00  1e-16  3e+00
 1:  2.1338e-03 -2.6330e-02  3e-02  6e-17  8e-02
 2: -1.9919e-03 -6.0382e-03  4e-03  1e-16  2e-03
 3: -2.5052e-03 -2.7924e-03  3e-04  2e-17  4e-18
 4: -2.5266e-03 -2.5392e-03  1e-05  1e-16  7e-18
 5: -2.5269e-03 -2.5271e-03  2e-07  6e-17  7e-18
 6: -2.5269e-03 -2.5269e-03  2e-09  1e-16  7e-18
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  4.7698e-03 -1.0176e+00  1e+00  1e-16  3e+00
 1:  4.1637e-03 -2.5995e-02  3e-02  1e-16  9e-02
 2: -6.6739e-04 -5.1808e-03  5e-03  2e-16  1e-17
 3: -1.1113e-03 -1.3635e-03  3e-04  2e-16  5e-18
 4: -1.1234e-03 -1.1306e-03  7e-06  2e-16  1e-17
 5: -1.1235e-03 -1.1235e-03  7e-08  1e-16  8e-18
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  7.1659e-03 -1.0180e+00  1e+00  2e-16  3e+00
 1:  6.3929e-03 -2.6731e-02  3e-02  1e-16  9e-02
 2:  8.7409e-04 -4.5293e-03  5e-03  2e-16  2e-17
 3:  4.2663e-04  1.5470e-04  3e-04  2e-16  7e-18
 4:  4.1772e-04  4.1225e-04  5e-06  6e-17  7e-18
 5:  4.1771e-04  4.1765e-04  5e-08  1e-16  7e-18
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  9.8131e-03 -1.0190e+00  1e+00  8e-17  3e+00
 1:  8.8368e-03 -2.7433e-02  4e-02  1e-16  1e-01
 2:  2.5905e-03 -3.7401e-03  6e-03  2e-16  2e-17
 3:  2.1340e-03  1.7994e-03  3e-04  1e-16  2e-17
 4:  2.1261e-03  2.1207e-03  5e-06  2e-16  8e-18
 5:  2.1261e-03  2.1260e-03  5e-08  1e-16  3e-18
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  1.2733e-02 -1.0201e+00  1e+00  2e-16  3e+00
 1:  1.1513e-02 -2.8065e-02  4e-02  6e-17  1e-01
 2:  4.5137e-03 -2.7447e-03  7e-03  2e-16  3e-17
 3:  4.0417e-03  3.6614e-03  4e-04  1e-16  2e-17
 4:  4.0343e-03  4.0288e-03  5e-06  1e-16  1e-17
 5:  4.0343e-03  4.0342e-03  5e-08  2e-16  1e-17
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  1.5951e-02 -1.0211e+00  1e+00  8e-17  3e+00
 1:  1.4442e-02 -2.8592e-02  4e-02  5e-17  1e-01
 2:  6.6786e-03 -1.4703e-03  8e-03  6e-17  4e-17
 3:  6.1861e-03  5.7704e-03  4e-04  2e-16  2e-17
```

```
 4:  6.1788e-03  6.1732e-03  6e-06  3e-17  8e-18
 5:  6.1788e-03  6.1788e-03  6e-08  1e-16  1e-17
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  1.9494e-02 -1.0221e+00  1e+00  2e-16  3e+00
 1:  1.7648e-02 -2.8971e-02  5e-02  7e-17  1e-01
 2:  9.1241e-03  1.5691e-04  9e-03  1e-16  4e-17
 3:  8.6080e-03  8.1652e-03  4e-04  1e-16  1e-17
 4:  8.6007e-03  8.5948e-03  6e-06  0e+00  1e-17
 5:  8.6007e-03  8.6006e-03  6e-08  1e-16  1e-17
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  2.3394e-02 -1.0229e+00  1e+00  3e-16  3e+00
 1:  2.1161e-02 -2.9152e-02  5e-02  1e-16  1e-01
 2:  1.1894e-02  2.2096e-03  1e-02  9e-17  3e-17
 3:  1.1354e-02  1.0891e-02  5e-04  1e-16  1e-17
 4:  1.1346e-02  1.1340e-02  6e-06  1e-16  1e-17
 5:  1.1346e-02  1.1346e-02  6e-08  1e-16  1e-17
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  2.7690e-02 -1.0237e+00  1e+00  6e-17  3e+00
 1:  2.5016e-02 -2.9075e-02  5e-02  1e-16  2e-01
 2:  1.5039e-02  4.7592e-03  1e-02  2e-16  4e-17
 3:  1.4475e-02  1.4002e-02  5e-04  1e-16  2e-17
 4:  1.4468e-02  1.4462e-02  6e-06  1e-16  2e-17
 5:  1.4468e-02  1.4468e-02  6e-08  1e-16  2e-17
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  3.2424e-02 -1.0242e+00  1e+00  2e-16  3e+00
 1:  2.9256e-02 -2.8667e-02  6e-02  7e-17  2e-01
 2:  1.8614e-02  7.8764e-03  1e-02  1e-16  7e-17
 3:  1.8033e-02  1.7556e-02  5e-04  1e-16  3e-17
 4:  1.8025e-02  1.8019e-02  6e-06  1e-16  2e-17
 5:  1.8025e-02  1.8025e-02  6e-08  1e-16  1e-17
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  3.7649e-02 -1.0245e+00  1e+00  1e-16  3e+00
 1:  3.3933e-02 -2.7843e-02  6e-02  1e-16  2e-01
 2:  2.2687e-02  1.1633e-02  1e-02  2e-16  5e-17
 3:  2.2093e-02  2.1622e-02  5e-04  2e-16  2e-17
 4:  2.2086e-02  2.2080e-02  6e-06  1e-16  2e-17
 5:  2.2086e-02  2.2086e-02  6e-08  3e-17  2e-17
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  4.3425e-02 -1.0245e+00  1e+00  0e+00  3e+00
 1:  3.9107e-02 -2.6502e-02  7e-02  1e-16  2e-01
 2:  2.7332e-02  1.6107e-02  1e-02  2e-16  5e-17
 3:  2.6735e-02  2.6278e-02  5e-04  1e-16  3e-17
 4:  2.6729e-02  2.6723e-02  6e-06  1e-16  3e-17
 5:  2.6729e-02  2.6729e-02  6e-08  1e-16  5e-17
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  4.9826e-02 -1.0240e+00  1e+00  2e-16  3e+00
 1:  4.4852e-02 -2.4524e-02  7e-02  6e-17  2e-01
 2:  3.2636e-02  2.1380e-02  1e-02  1e-16  5e-17
 3:  3.2048e-02  3.1611e-02  4e-04  2e-17  3e-17
 4:  3.2042e-02  3.2037e-02  5e-06  1e-16  3e-17
 5:  3.2042e-02  3.2042e-02  5e-08  2e-16  5e-17
```

```
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  5.6936e-02 -1.0230e+00  1e+00  2e-16  3e+00
 1:  5.1255e-02 -2.1767e-02  7e-02  1e-16  2e-01
 2:  3.8702e-02  2.7548e-02  1e-02  5e-17  6e-17
 3:  3.8131e-02  3.7723e-02  4e-04  1e-16  5e-17
 4:  3.8126e-02  3.8122e-02  5e-06  1e-16  5e-17
 5:  3.8126e-02  3.8126e-02  5e-08  3e-17  5e-17
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  6.4856e-02 -1.0214e+00  1e+00  0e+00  3e+00
 1:  5.8417e-02 -1.8066e-02  8e-02  9e-17  2e-01
 2:  4.5644e-02  3.4720e-02  1e-02  1e-16  6e-17
 3:  4.5103e-02  4.4727e-02  4e-04  3e-17  3e-17
 4:  4.5099e-02  4.5095e-02  4e-06  1e-16  3e-17
 5:  4.5099e-02  4.5099e-02  4e-08  1e-16  3e-17
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  7.3704e-02 -1.0189e+00  1e+00  6e-16  3e+00
 1:  6.6461e-02 -1.3231e-02  8e-02  7e-17  2e-01
 2:  5.3599e-02  4.3022e-02  1e-02  1e-16  4e-17
 3:  5.3096e-02  5.2757e-02  3e-04  2e-16  4e-17
 4:  5.3093e-02  5.3089e-02  4e-06  2e-16  4e-17
 5:  5.3093e-02  5.3093e-02  4e-08  1e-16  3e-17
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  8.3617e-02 -1.0155e+00  1e+00  6e-17  3e+00
 1:  7.5530e-02 -7.0412e-03  8e-02  2e-16  2e-01
 2:  6.2719e-02  5.2599e-02  1e-02  3e-16  7e-17
 3:  6.2264e-02  6.1963e-02  3e-04  2e-16  6e-17
 4:  6.2261e-02  6.2258e-02  3e-06  1e-16  9e-17
 5:  6.2261e-02  6.2261e-02  3e-08  2e-16  5e-17
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  9.4756e-02 -1.0109e+00  1e+00  3e-16  3e+00
 1:  8.5791e-02  7.5092e-04  9e-02  2e-16  2e-01
 2:  7.3183e-02  6.3620e-02  1e-02  1e-16  7e-17
 3:  7.2781e-02  7.2519e-02  3e-04  1e-16  8e-17
 4:  7.2779e-02  7.2776e-02  3e-06  2e-17  9e-17
 5:  7.2779e-02  7.2779e-02  3e-08  2e-16  9e-17
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  1.0731e-01 -1.0049e+00  1e+00  6e-17  3e+00
 1:  9.7445e-02  1.0423e-02  9e-02  1e-16  2e-01
 2:  8.5195e-02  7.6277e-02  9e-03  1e-16  6e-17
 3:  8.4848e-02  8.4625e-02  2e-04  1e-16  5e-17
 4:  8.4847e-02  8.4845e-02  2e-06  2e-17  7e-17
 5:  8.4847e-02  8.4847e-02  2e-08  1e-16  7e-17
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  1.2150e-01 -9.9717e-01  1e+00  3e-16  3e+00
 1:  1.1073e-01  2.2280e-02  9e-02  1e-16  3e-01
 2:  9.8987e-02  9.0792e-02  8e-03  1e-16  7e-17
 3:  9.8698e-02  9.8511e-02  2e-04  1e-16  1e-16
 4:  9.8697e-02  9.8695e-02  2e-06  2e-16  9e-17
 5:  9.8697e-02  9.8697e-02  2e-08  2e-16  4e-17
Optimal solution found.
     pcost       dcost       gap    pres   dres
```

```
 0:  1.3757e-01 -9.8746e-01  1e+00  0e+00  3e+00
 1:  1.2591e-01  3.6658e-02  9e-02  7e-17  3e-01
 2:  1.1483e-01  1.0741e-01  7e-03  2e-16  1e-16
 3:  1.1459e-01  1.1443e-01  2e-04  2e-16  4e-17
 4:  1.1459e-01  1.1459e-01  2e-06  1e-16  8e-17
 5:  1.1459e-01  1.1459e-01  2e-08  1e-16  1e-16
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  1.5583e-01 -9.7539e-01  1e+00  1e-16  3e+00
 1:  1.4333e-01  5.3920e-02  9e-02  1e-16  3e-01
 2:  1.3302e-01  1.2643e-01  7e-03  1e-16  9e-17
 3:  1.3284e-01  1.3271e-01  1e-04  1e-16  7e-17
 4:  1.3284e-01  1.3284e-01  1e-06  1e-16  1e-16
 5:  1.3284e-01  1.3284e-01  1e-08  1e-16  1e-16
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  1.7662e-01 -9.6056e-01  1e+00  6e-17  3e+00
 1:  1.6334e-01  7.4462e-02  9e-02  8e-17  3e-01
 2:  1.5392e-01  1.4818e-01  6e-03  1e-16  1e-16
 3:  1.5378e-01  1.5368e-01  1e-04  2e-16  1e-16
 4:  1.5378e-01  1.5378e-01  1e-06  1e-16  5e-17
 5:  1.5378e-01  1.5378e-01  1e-08  1e-16  1e-16
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  2.0034e-01 -9.4252e-01  1e+00  0e+00  3e+00
 1:  1.8639e-01  9.8711e-02  9e-02  2e-16  2e-01
 2:  1.7793e-01  1.7302e-01  5e-03  1e-16  2e-16
 3:  1.7783e-01  1.7775e-01  8e-05  1e-16  2e-16
 4:  1.7783e-01  1.7782e-01  8e-07  1e-16  1e-16
 5:  1.7783e-01  1.7783e-01  8e-09  1e-16  2e-16
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  2.2745e-01 -9.2075e-01  1e+00  1e-16  3e+00
 1:  2.1297e-01  1.2714e-01  9e-02  2e-16  2e-01
 2:  2.0550e-01  2.0140e-01  4e-03  3e-17  1e-16
 3:  2.0543e-01  2.0537e-01  6e-05  1e-16  1e-16
 4:  2.0543e-01  2.0543e-01  6e-07  2e-16  7e-17
 5:  2.0543e-01  2.0543e-01  6e-09  1e-16  9e-17
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  2.5850e-01 -8.9468e-01  1e+00  3e-16  3e+00
 1:  2.4363e-01  1.6026e-01  8e-02  6e-17  2e-01
 2:  2.3716e-01  2.3382e-01  3e-03  1e-16  2e-16
 3:  2.3712e-01  2.3707e-01  4e-05  3e-16  1e-16
 4:  2.3712e-01  2.3712e-01  4e-07  5e-17  2e-16
 5:  2.3712e-01  2.3712e-01  4e-09  1e-16  1e-16
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  2.9411e-01 -8.6366e-01  1e+00  2e-16  4e+00
 1:  2.7902e-01  1.9865e-01  8e-02  1e-16  2e-01
 2:  2.7353e-01  2.7087e-01  3e-03  1e-16  4e-16
 3:  2.7350e-01  2.7346e-01  3e-05  1e-16  2e-16
 4:  2.7350e-01  2.7350e-01  3e-07  3e-17  2e-16
 5:  2.7350e-01  2.7350e-01  3e-09  1e-16  1e-16
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  3.3498e-01 -8.2696e-01  1e+00  1e-16  4e+00
 1:  3.1986e-01  2.4296e-01  8e-02  1e-16  2e-01
```

```
 2:  3.1528e-01  3.1320e-01  2e-03  1e-16  8e-05
 3:  3.1526e-01  3.1524e-01  2e-05  3e-17  8e-07
 4:  3.1526e-01  3.1526e-01  2e-07  1e-16  8e-09
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  3.8194e-01 -7.8375e-01  1e+00  1e-16  4e+00
 1:  3.6697e-01  2.9392e-01  7e-02  1e-16  2e-01
 2:  3.6323e-01  3.6137e-01  2e-03  8e-17  9e-04
 3:  3.6322e-01  3.6320e-01  2e-05  1e-16  9e-06
 4:  3.6322e-01  3.6322e-01  2e-07  1e-16  9e-08
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  4.3592e-01 -7.3311e-01  1e+00  4e-16  4e+00
 1:  4.2128e-01  3.5238e-01  7e-02  2e-16  2e-01
 2:  4.1828e-01  4.1668e-01  2e-03  1e-16  1e-03
 3:  4.1828e-01  4.1826e-01  2e-05  1e-16  1e-05
 4:  4.1828e-01  4.1828e-01  2e-07  2e-16  1e-07
 5:  4.1828e-01  4.1828e-01  2e-09  1e-16  1e-09
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  4.9801e-01 -6.7398e-01  1e+00  6e-17  4e+00
 1:  4.8385e-01  4.1930e-01  6e-02  1e-16  2e-01
 2:  4.8149e-01  4.8009e-01  1e-03  2e-16  2e-03
 3:  4.8149e-01  4.8147e-01  1e-05  7e-18  2e-05
 4:  4.8149e-01  4.8149e-01  1e-07  6e-17  2e-07
 5:  4.8149e-01  4.8149e-01  1e-09  2e-16  2e-09
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  5.6943e-01 -6.0514e-01  1e+00  1e-16  4e+00
 1:  5.5588e-01  4.9581e-01  6e-02  1e-16  2e-01
 2:  5.5407e-01  5.5288e-01  1e-03  1e-16  2e-03
 3:  5.5407e-01  5.5405e-01  1e-05  1e-16  2e-05
 4:  5.5407e-01  5.5407e-01  1e-07  2e-16  2e-07
 5:  5.5407e-01  5.5407e-01  1e-09  2e-16  2e-09
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  6.5160e-01 -5.2522e-01  1e+00  6e-17  4e+00
 1:  6.3876e-01  5.8319e-01  6e-02  1e-16  1e-01
 2:  6.3739e-01  6.3640e-01  1e-03  1e-16  2e-03
 3:  6.3739e-01  6.3738e-01  1e-05  1e-16  2e-05
 4:  6.3739e-01  6.3739e-01  1e-07  1e-16  2e-07
 5:  6.3739e-01  6.3739e-01  1e-09  1e-16  2e-09
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  7.4612e-01 -4.3263e-01  1e+00  2e-16  4e+00
 1:  7.3408e-01  6.8296e-01  5e-02  1e-16  1e-01
 2:  7.3307e-01  7.3224e-01  8e-04  2e-16  2e-03
 3:  7.3307e-01  7.3306e-01  8e-06  1e-16  2e-05
 4:  7.3307e-01  7.3307e-01  8e-08  2e-16  2e-07
 5:  7.3307e-01  7.3307e-01  8e-10  2e-16  2e-09
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  8.5484e-01 -3.2557e-01  1e+00  1e-16  4e+00
 1:  8.4365e-01  7.9686e-01  5e-02  1e-16  1e-01
 2:  8.4291e-01  8.4223e-01  7e-04  1e-16  1e-03
 3:  8.4291e-01  8.4291e-01  7e-06  2e-16  1e-05
 4:  8.4291e-01  8.4291e-01  7e-08  1e-16  1e-07
 5:  8.4291e-01  8.4291e-01  7e-10  1e-16  1e-09
```

```
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  9.7988e-01 -2.0194e-01  1e+00  3e-16  4e+00
 1:  9.6956e-01  9.2691e-01  4e-02  1e-16  1e-01
 2:  9.6903e-01  9.6847e-01  6e-04  2e-16  1e-03
 3:  9.6903e-01  9.6903e-01  6e-06  1e-16  1e-05
 4:  9.6903e-01  9.6903e-01  6e-08  1e-16  1e-07
 5:  9.6903e-01  9.6903e-01  6e-10  2e-16  1e-09
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  1.1237e+00 -5.9365e-02  1e+00  0e+00  4e+00
 1:  1.1142e+00  1.0755e+00  4e-02  1e-16  1e-01
 2:  1.1138e+00  1.1134e+00  5e-04  1e-16  1e-03
 3:  1.1138e+00  1.1138e+00  5e-06  4e-17  1e-05
 4:  1.1138e+00  1.1138e+00  5e-08  1e-16  1e-07
 5:  1.1138e+00  1.1138e+00  5e-10  1e-16  1e-09
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  1.2889e+00  1.0491e-01  1e+00  3e-16  4e+00
 1:  1.2803e+00  1.2452e+00  4e-02  1e-16  9e-02
 2:  1.2801e+00  1.2797e+00  4e-04  1e-16  1e-03
 3:  1.2801e+00  1.2801e+00  4e-06  1e-16  1e-05
 4:  1.2801e+00  1.2801e+00  4e-08  4e-17  1e-07
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  1.4789e+00  2.9404e-01  1e+00  1e-16  4e+00
 1:  1.4712e+00  1.4394e+00  3e-02  1e-16  9e-02
 2:  1.4710e+00  1.4706e+00  3e-04  2e-16  9e-04
 3:  1.4710e+00  1.4710e+00  3e-06  1e-16  9e-06
 4:  1.4710e+00  1.4710e+00  3e-08  2e-16  9e-08
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  1.6973e+00  5.1165e-01  1e+00  1e-16  4e+00
 1:  1.6903e+00  1.6615e+00  3e-02  1e-16  8e-02
 2:  1.6901e+00  1.6898e+00  3e-04  2e-17  8e-04
 3:  1.6901e+00  1.6901e+00  3e-06  1e-16  8e-06
 4:  1.6901e+00  1.6901e+00  3e-08  1e-16  8e-08
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  1.9481e+00  7.6191e-01  1e+00  0e+00  4e+00
 1:  1.9419e+00  1.9157e+00  3e-02  1e-16  7e-02
 2:  1.9418e+00  1.9415e+00  3e-04  2e-16  7e-04
 3:  1.9418e+00  1.9418e+00  3e-06  1e-16  7e-06
 4:  1.9418e+00  1.9418e+00  3e-08  1e-16  7e-08
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  2.2363e+00  1.0496e+00  1e+00  1e-16  4e+00
 1:  2.2308e+00  2.2070e+00  2e-02  1e-16  7e-02
 2:  2.2307e+00  2.2305e+00  2e-04  1e-16  7e-04
 3:  2.2307e+00  2.2307e+00  2e-06  1e-16  7e-06
 4:  2.2307e+00  2.2307e+00  2e-08  1e-16  7e-08
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  2.5674e+00  1.3803e+00  1e+00  1e-16  4e+00
 1:  2.5625e+00  2.5407e+00  2e-02  2e-17  6e-02
 2:  2.5624e+00  2.5622e+00  2e-04  1e-16  6e-04
 3:  2.5624e+00  2.5624e+00  2e-06  1e-16  6e-06
 4:  2.5624e+00  2.5624e+00  2e-08  6e-17  6e-08
```

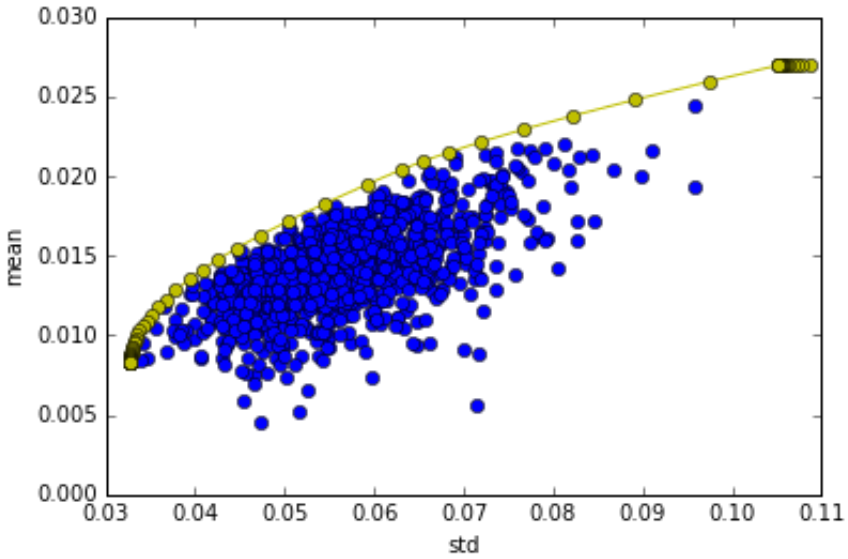
```
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  2.9477e+00  1.7602e+00  1e+00  1e-16  4e+00
 1:  2.9433e+00  2.9233e+00  2e-02  2e-16  6e-02
 2:  2.9433e+00  2.9431e+00  2e-04  1e-16  6e-04
 3:  2.9433e+00  2.9433e+00  2e-06  2e-16  6e-06
 4:  2.9433e+00  2.9433e+00  2e-08  1e-16  6e-08
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  3.3845e+00  2.1967e+00  1e+00  0e+00  4e+00
 1:  3.3806e+00  3.3621e+00  2e-02  1e-16  5e-02
 2:  3.3806e+00  3.3804e+00  2e-04  9e-17  5e-04
 3:  3.3806e+00  3.3806e+00  2e-06  1e-16  5e-06
 4:  3.3806e+00  3.3806e+00  2e-08  2e-16  5e-08
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  3.8861e+00  2.6980e+00  1e+00  1e-16  4e+00
 1:  3.8827e+00  3.8655e+00  2e-02  2e-16  5e-02
 2:  3.8827e+00  3.8825e+00  2e-04  1e-16  5e-04
 3:  3.8827e+00  3.8827e+00  2e-06  1e-16  5e-06
 4:  3.8827e+00  3.8827e+00  2e-08  1e-16  5e-08
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  4.4622e+00  3.2739e+00  1e+00  2e-16  4e+00
 1:  4.4591e+00  4.4430e+00  2e-02  4e-17  5e-02
 2:  4.4591e+00  4.4590e+00  2e-04  2e-16  5e-04
 3:  4.4591e+00  4.4591e+00  2e-06  4e-17  5e-06
 4:  4.4591e+00  4.4591e+00  2e-08  3e-17  5e-08
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  5.1237e+00  3.9352e+00  1e+00  1e-16  4e+00
 1:  5.1210e+00  5.1057e+00  2e-02  1e-16  5e-02
 2:  5.1210e+00  5.1209e+00  2e-04  2e-16  5e-04
 3:  5.1210e+00  5.1210e+00  2e-06  4e-17  5e-06
 4:  5.1210e+00  5.1210e+00  2e-08  1e-16  5e-08
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  5.8833e+00  4.6946e+00  1e+00  1e-16  4e+00
 1:  5.8809e+00  5.8663e+00  1e-02  1e-16  4e-02
 2:  5.8809e+00  5.8808e+00  1e-04  2e-17  4e-04
 3:  5.8809e+00  5.8809e+00  1e-06  1e-16  4e-06
 4:  5.8809e+00  5.8809e+00  1e-08  1e-16  4e-08
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  6.7555e+00  5.5667e+00  1e+00  2e-16  4e+00
 1:  6.7535e+00  6.7394e+00  1e-02  1e-16  4e-02
 2:  6.7535e+00  6.7533e+00  1e-04  1e-16  4e-04
 3:  6.7535e+00  6.7534e+00  1e-06  1e-16  4e-06
 4:  6.7535e+00  6.7535e+00  1e-08  6e-17  4e-08
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  7.7570e+00  6.5681e+00  1e+00  0e+00  4e+00
 1:  7.7552e+00  7.7416e+00  1e-02  1e-16  4e-02
 2:  7.7552e+00  7.7551e+00  1e-04  1e-16  4e-04
 3:  7.7552e+00  7.7552e+00  1e-06  3e-17  4e-06
 4:  7.7552e+00  7.7552e+00  1e-08  3e-16  4e-08
Optimal solution found.
     pcost       dcost       gap    pres   dres
```

```
 0:  8.9070e+00  7.7180e+00  1e+00  2e-16  4e+00
 1:  8.9054e+00  8.8922e+00  1e-02  1e-16  4e-02
 2:  8.9054e+00  8.9053e+00  1e-04  1e-16  4e-04
 3:  8.9054e+00  8.9054e+00  1e-06  2e-16  4e-06
 4:  8.9054e+00  8.9054e+00  1e-08  1e-16  4e-08
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  1.0227e+01  9.0383e+00  1e+00  1e-16  4e+00
 1:  1.0226e+01  1.0213e+01  1e-02  1e-16  4e-02
 2:  1.0226e+01  1.0226e+01  1e-04  2e-16  4e-04
 3:  1.0226e+01  1.0226e+01  1e-06  1e-16  4e-06
 4:  1.0226e+01  1.0226e+01  1e-08  1e-16  4e-08
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  1.1744e+01  1.0554e+01  1e+00  1e-16  4e+00
 1:  1.1742e+01  1.1730e+01  1e-02  1e-17  4e-02
 2:  1.1742e+01  1.1742e+01  1e-04  3e-17  4e-04
 3:  1.1742e+01  1.1742e+01  1e-06  1e-16  4e-06
 4:  1.1742e+01  1.1742e+01  1e-08  1e-16  4e-08
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  1.3484e+01  1.2295e+01  1e+00  0e+00  4e+00
 1:  1.3483e+01  1.3471e+01  1e-02  1e-16  4e-02
 2:  1.3483e+01  1.3483e+01  1e-04  1e-16  4e-04
 3:  1.3483e+01  1.3483e+01  1e-06  1e-16  4e-06
 4:  1.3483e+01  1.3483e+01  1e-08  1e-16  4e-08
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  1.5483e+01  1.4294e+01  1e+00  0e+00  4e+00
 1:  1.5482e+01  1.5470e+01  1e-02  1e-16  4e-02
 2:  1.5482e+01  1.5482e+01  1e-04  1e-16  4e-04
 3:  1.5482e+01  1.5482e+01  1e-06  1e-16  4e-06
 4:  1.5482e+01  1.5482e+01  1e-08  1e-16  4e-08
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  1.7778e+01  1.6588e+01  1e+00  2e-16  4e+00
 1:  1.7777e+01  1.7765e+01  1e-02  3e-16  4e-02
 2:  1.7777e+01  1.7777e+01  1e-04  1e-16  4e-04
 3:  1.7777e+01  1.7777e+01  1e-06  1e-16  4e-06
 4:  1.7777e+01  1.7777e+01  1e-08  1e-16  4e-08
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  2.0413e+01  1.9223e+01  1e+00  1e-16  4e+00
 1:  2.0412e+01  2.0400e+01  1e-02  2e-16  4e-02
 2:  2.0412e+01  2.0412e+01  1e-04  1e-16  4e-04
 3:  2.0412e+01  2.0412e+01  1e-06  1e-16  4e-06
 4:  2.0412e+01  2.0412e+01  1e-08  6e-17  4e-08
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  2.3438e+01  2.2248e+01  1e+00  2e-16  4e+00
 1:  2.3437e+01  2.3425e+01  1e-02  2e-16  4e-02
 2:  2.3437e+01  2.3437e+01  1e-04  1e-16  4e-04
 3:  2.3437e+01  2.3437e+01  1e-06  1e-16  4e-06
 4:  2.3437e+01  2.3437e+01  1e-08  1e-16  4e-08
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  2.6911e+01  2.5722e+01  1e+00  1e-16  4e+00
 1:  2.6911e+01  2.6899e+01  1e-02  3e-17  4e-02
```

```
 2:  2.6911e+01  2.6911e+01  1e-04  1e-16  4e-04
 3:  2.6911e+01  2.6911e+01  1e-06  1e-16  4e-06
 4:  2.6911e+01  2.6911e+01  1e-08  1e-16  4e-08
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  3.0899e+01  2.9710e+01  1e+00  1e-16  4e+00
 1:  3.0899e+01  3.0887e+01  1e-02  1e-16  4e-02
 2:  3.0899e+01  3.0899e+01  1e-04  1e-16  4e-04
 3:  3.0899e+01  3.0899e+01  1e-06  1e-16  4e-06
 4:  3.0899e+01  3.0899e+01  1e-08  2e-17  4e-08
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  3.5478e+01  3.4289e+01  1e+00  1e-16  4e+00
 1:  3.5478e+01  3.5466e+01  1e-02  2e-17  4e-02
 2:  3.5478e+01  3.5478e+01  1e-04  1e-16  4e-04
 3:  3.5478e+01  3.5478e+01  1e-06  1e-16  4e-06
 4:  3.5478e+01  3.5478e+01  1e-08  6e-17  4e-08
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  4.0736e+01  3.9546e+01  1e+00  2e-16  4e+00
 1:  4.0735e+01  4.0723e+01  1e-02  1e-16  4e-02
 2:  4.0735e+01  4.0735e+01  1e-04  3e-16  4e-04
 3:  4.0735e+01  4.0735e+01  1e-06  3e-17  4e-06
 4:  4.0735e+01  4.0735e+01  1e-08  1e-16  4e-08
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0:  4.6772e+01  4.5583e+01  1e+00  0e+00  4e+00
 1:  4.6772e+01  4.6760e+01  1e-02  1e-16  4e-02
 2:  4.6772e+01  4.6772e+01  1e-04  1e-16  4e-04
 3:  4.6772e+01  4.6772e+01  1e-06  3e-17  4e-06
 4:  4.6772e+01  4.6772e+01  1e-08  1e-16  4e-08
Optimal solution found.
     pcost       dcost       gap    pres   dres
 0: -1.4809e-02 -1.0272e+00  1e+00  1e-16  3e+00
 1: -1.5032e-02 -3.7245e-02  2e-02  6e-17  6e-02
 2: -2.3691e-02 -2.7286e-02  4e-03  8e-17  6e-03
 3: -2.6902e-02 -2.7277e-02  4e-04  1e-15  1e-17
 4: -2.7088e-02 -2.7104e-02  2e-05  2e-16  9e-18
 5: -2.7101e-02 -2.7101e-02  4e-07  1e-16  1e-17
 6: -2.7101e-02 -2.7101e-02  4e-09  2e-17  1e-17
Optimal solution found.
```

In [405]:

```
plt.plot(stds,means , 'o');
plt.ylabel('mean');
plt.xlabel('std');
plt.plot(risks, returns, 'y-o');
plt.show()
```



Following is a trial I wrote in order to calculate the weight when arbitrage is allowed. The code uses Markov Chain Monte Carlo algorithm, but as we can see, the result of it isn't so good as convex optimization (The efficiency is low, and accuracy isn't satisfactory). This may be caused by the reason that sample population isn't large enough.

In [334]:

```
def Max_theta_Arb(cov,rate_of_return,weight,c,Nsample=10**5):
    #np.random.seed(100)
    trace=np.zeros((Nsample,len(weight)))
    Accepted=0
    max_fun=lambda cov, rate_of_return,weight_raw,c:\
    (np.dot(weight_raw,rate_of_return)-c)/\
    np.sqrt(np.dot(weight_raw/float(np.sum(weight_raw)),\
                   np.dot(cov,weight_raw/float(np.sum(weight_raw)))))
    weight_raw=weight/float(np.sum(weight))
    weight_star=weight_raw/float(np.sum(weight_raw))
    for j in range(Nsample):
#-------------------------------------------------------------------------
        weight_star=weight_raw
        weight_star[0]=np.random.normal(weight_raw[0],1)
        weight_star=weight_star/float(np.sum(weight_star))
        fun_raw=max_fun(cov,rate_of_return,weight_raw,c)
        fun_star=max_fun(cov,rate_of_return,weight_star,c)
        U=np.random.uniform(0,1)
        R=float(fun_star/fun_raw)
        if R>U:
            trace[j,0]=weight_star[0]
            Accepted+=1
            weight_raw[0]=weight_star[0]
        else:
            trace[j,0]=weight_raw[0]
```

```
            weight_raw[0]=weight_raw[0]
#-------------------------------------------------------------------
        weight_star=weight_raw
        weight_star[1]=np.random.normal(weight_raw[1],1)
        weight_star=weight_star/float(np.sum(weight_star))
        fun_raw=max_fun(cov,rate_of_return,weight_raw,c)
        fun_star=max_fun(cov,rate_of_return,weight_star,c)
        U=np.random.uniform(0,1)
        R=float(fun_star/fun_raw)
        if R>U:
            trace[j,1]=weight_star[1]
            Accepted+=1
            weight_raw[1]=weight_star[1]
        else:
            trace[j,1]=weight_raw[1]
            weight_raw[1]=weight_raw[1]
#-------------------------------------------------------------------
        weight_star=weight_raw
        weight_star[2]=np.random.normal(weight_raw[2],1)
        weight_star=weight_star/float(np.sum(weight_star))
        fun_raw=max_fun(cov,rate_of_return,weight_raw,c)
        fun_star=max_fun(cov,rate_of_return,weight_star,c)
        U=np.random.uniform(0,1)
        R=float(fun_star/fun_raw)
        if R>U:
            trace[j,2]=weight_star[2]
            Accepted+=1
            weight_raw[2]=weight_star[2]
        else:
            trace[j,2]=weight_raw[2]
            weight_raw[2]=weight_raw[2]
#-------------------------------------------------------------------
        weight_star=weight_raw
        weight_star[3]=np.random.normal(weight_raw[3],1)
        weight_star=weight_star/float(np.sum(weight_star))
        fun_raw=max_fun(cov,rate_of_return,weight_raw,c)
        fun_star=max_fun(cov,rate_of_return,weight_star,c)
        U=np.random.uniform(0,1)
        R=float(fun_star/fun_raw)
        if R>U:
            trace[j,3]=weight_star[3]
            Accepted+=1
            weight_raw[3]=weight_star[3]
        else:
            trace[j,3]=weight_raw[3]
            weight_raw[3]=weight_raw[3]
#-------------------------------------------------------------------
        weight_star=weight_raw
        weight_star[4]=np.random.normal(weight_raw[4],1)
        weight_star=weight_star/float(np.sum(weight_star))
        fun_raw=max_fun(cov,rate_of_return,weight_raw,c)
        fun_star=max_fun(cov,rate_of_return,weight_star,c)
        U=np.random.uniform(0,1)
        R=float(fun_star/fun_raw)
        if R>U:
            trace[j,4]=weight_star[4]
            Accepted+=1
            weight_raw[4]=weight_star[4]
```

```
        else:
            trace[j,4]=weight_raw[4]
            weight_raw[4]=weight_raw[4]
#-----------------------------------------------------------------
        weight_star=weight_raw
        weight_star[5]=np.random.normal(weight_raw[5],1)
        weight_star=weight_star/float(np.sum(weight_star))
        fun_raw=max_fun(cov,rate_of_return,weight_raw,c)
        fun_star=max_fun(cov,rate_of_return,weight_star,c)
        U=np.random.uniform(0,1)
        R=float(fun_star/fun_raw)
        if R>U:
            trace[j,5]=weight_star[5]
            Accepted+=1
            weight_raw[5]=weight_star[5]
        else:
            trace[j,5]=weight_raw[5]
            weight_raw[5]=weight_raw[5]
    return trace
```

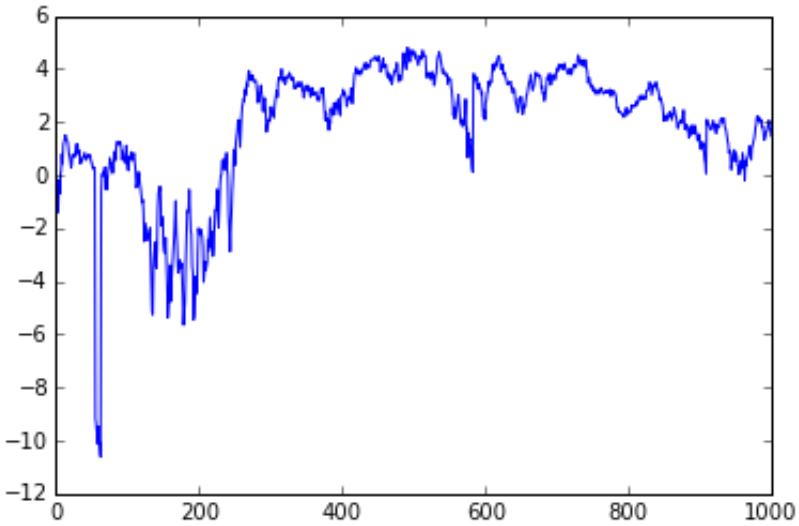
In [396]:

```
weight=np.zeros(N6)
for i in range(N6):
    weight[i]=1./N6


trace=Max_theta_Arb(cov_matrix, rate_of_return, weight, 0.00, \
                    Nsample=10**4)
max_fun=lambda cov, rate_of_return,weight_raw,c:\
(np.dot(weight_raw,rate_of_return)-c)/\
np.sqrt(np.dot(weight_raw/float(np.sum(weight_raw)),\
               np.dot(cov,weight_raw/float(np.sum(weight_raw)))))

trace_p=np.zeros(10**3)
for i in range(10**3):
    trace_p[i]=max_fun(cov_matrix, rate_of_return, trace[i,:], 0.00)
```

This is a trace plot of the slope (expected return/standard deviation). In this figure, there are several peaks. This explains why when we set different start value in Excel's solver, we will get totally different weight of the portfolio.

In [402]:

```
plt.plot(trace_p);
```



Simulate 4000 samples for 4000 times iterations.

In [373]:

```
samples=np.zeros((4000,2))
for i in range(4000):
    #np.random.seed(80)
    c=np.random.uniform(-10,1)
    c=c/100.

    trace=Max_theta_Arb(cov_matrix, rate_of_return, weight, c, \
                        Nsample=4000)
    max_fun=lambda cov, rate_of_return,weight_raw,c:\
    (np.dot((weight_raw)/\
            float(np.sum(weight_raw)),rate_of_return)-c)/\
    np.sqrt(np.dot(weight_raw/float(np.sum(weight_raw)),\
                   np.dot(cov,weight_raw/float(np.sum(weight_raw)))))
    trace_p=np.zeros(4000)
    for j in range(4000):
        trace_p[j]=max_fun(cov_matrix, rate_of_return, trace[j,:], c)
    weight_ad=trace[np.argmax(trace_p),:]
    weight_ad=weight_ad/float(np.sum(weight_ad))
    returns=np.dot(rate_of_return,weight_ad)
    stds=np.sqrt(np.dot(weight_ad,np.dot(cov_matrix,weight_ad)))

    samples[i,0]=returns
    samples[i,1]=stds
    rate=(i+1)/4000.*100
    print '\r%d%%'%rate,
```

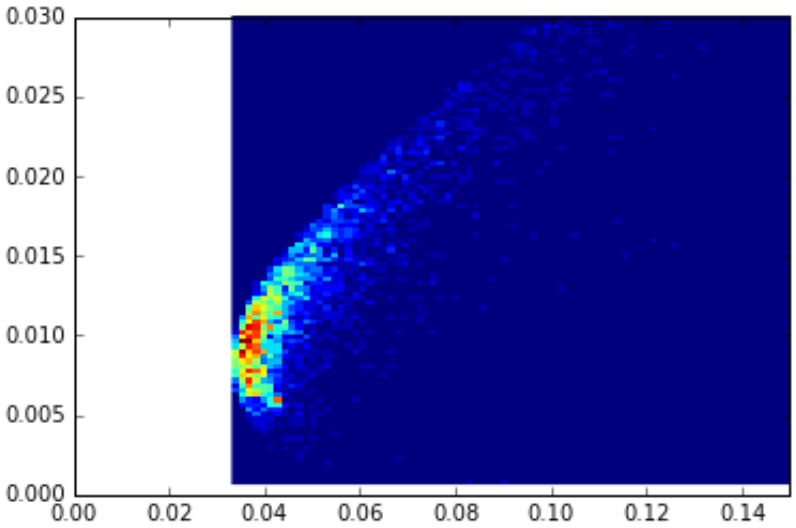
```
100%
```

In [388]:

```
plt.hist2d(samples[:,1],samples[:,0],bins=4000);
plt.xlim(xmin=0,xmax=0.15)
plt.ylim(ymin=0,ymax=0.03)
```

Out[388]:

(0, 0.03)

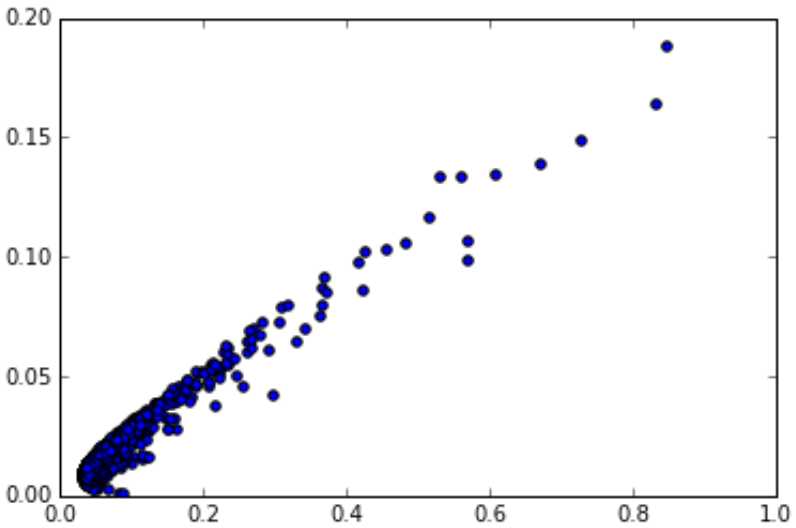


In [392]:

```
plt.scatter(samples[:,1],samples[:,0])
plt.xlim(xmin=0,xmax=1)
plt.ylim(ymin=0,ymax=0.2)
```

Out[392]:

(0, 0.2)



From the figure above, we can find that the shape has the same pattern as the efficient frontier calculated by cvxopt. But the accuracy is a little bit lower than cvxopt. The point that has the least standard deviation is close to point (0.03,0.01), which is the point with least standard deviation in the figure in 'cvxopt' part. When arbitrage is allowed, the frontier will expand to upper side without limitation, which is the effect of arbitrage.