Convex Optimization

Tutorial 9

Tanmay Garg CS20BTECH11063

In [ ]:

```
#Importing required Libraries
import numpy as np
import matplotlib.pyplot as plt
import cvxpy as cp
import math
```

In [ ]:

```
# Gain matrix
G = np.matrix(( [1, 0.1, 0.2, 0.1, 0],
                [0.1, 1, 0.1, 0.1, 0],
                [0.2, 0.1, 2, 0.2, 0.2],
                [0.1, 0.1, 0.2, 1, 0.1],
                [0, 0, 0.2, 0.1, 1]))

# Number of transmitters
n = 5

#Gain matrix for Calculating Signal Power
G_for_Signal_Mat = np.multiply(G, np.identity(n))

#Gain Matrix for Calculating Interference Power
G_for_Inter_Mat = G - G_for_Signal_Mat

#Matrix for getting groups of transmitters
groups = np.matrix(([1, 1, 0, 0, 0],
                    [0, 0, 1, 1, 1]))

#Maximum power of each group
groups_max_val = np.matrix(([4,6]))
groups_max_val = np.reshape(groups_max_val, (2,1))

#Maximum Power for each transmitter
P_max = 3 * np.ones((n,1))
P_max = np.reshape(P_max, (n,1))

#Maximum Power for each receiver
P_rc = 5 * np.ones((n,1))

alpha = cp.Parameter(1)

#Self Noise Power
sigma = 0.5 * np.ones((n,1))
```

In [ ]:

```
p = cp.Variable((n,1))
# Upper Bound
u = 1e4
# Lower Bound
l = 0
```

In [ ]:

```
MyObjective = cp.Minimize(alpha)

MyConstraints = [
    p >= 0,
    p <= P_max,
```

```
    G@p <= P_rc,
    (groups)@p <= groups_max_val,
    G_for_Inter_Mat@p + sigma <= alpha*G_for_Signal_Mat@p
]
```

In [ ]:

```
# alpha.value = [u]
# MyProblem = cp.Problem(MyObjective, MyConstraints)
# MyProblem.solve()

# alpha.value = [l]
# MyProblem = cp.Problem(MyObjective, MyConstraints)
# MyProblem.solve()

for i in range(1, 10000):
    alpha.value = np.atleast_1d(((u + l)/2.0))
    # tmp = np.ones((1,))
    # tmp[0] = (u+l/2.0)
    # print(tmp.shape)
    # print(tmp)
    # alpha.value = tmp
    print(MyProblem.status)
    if u - l <= 0.005:
        break
    MyProblem = cp.Problem(MyObjective, MyConstraints)
    MyProblem.solve()

    if MyProblem.status == 'optimal':
        u = alpha.value
    else:
        l = alpha.value
```

```
optimal
optimal
optimal
optimal
optimal
optimal
optimal
optimal
optimal
optimal
optimal
optimal
optimal
optimal
optimal
infeasible
infeasible
infeasible
infeasible
infeasible
optimal
optimal
```

In [ ]:

```
print("SINR Value: {}".format(1/alpha.value))
print("Value of powers for transmitters: ")
print(p.value)
```

```
SINR Value: [1.68445944]
Value of powers for transmitters:
[[2.10756611]
 [1.88019374]
 [1.64000272]
 [2.37664543]
 [1.84235581]]
```