HW12

May 23, 2020

1 HW12

1.1 Alireza Darvishi 96109674

[1]:
```
from scipy.sparse import coo_matrix
import numpy as np
import cvxpy as cp
import matplotlib.pyplot as plt
```

1.1.1 A6.30.b

[2]:
```
from team_data import *
A1 = coo_matrix((train[:, 2], (range(m), train[:, 0])), shape=(m, n)).toarray()
A2 = coo_matrix((-train[:, 2], (range(m), train[:, 1])), shape=(m, n)).toarray()
A = A1 + A2
```

[3]:
```
a = cp.Variable(n)
objective = cp.Minimize(cp.sum(cp.logistic(-2 / sigma * A @ a)))
constraint = [a <= 1, a >= 0]
prob = cp.Problem(objective, constraint)
prob.solve()
a_hat = a.value - min(a.value)
```

1.1.2 A6.30.c

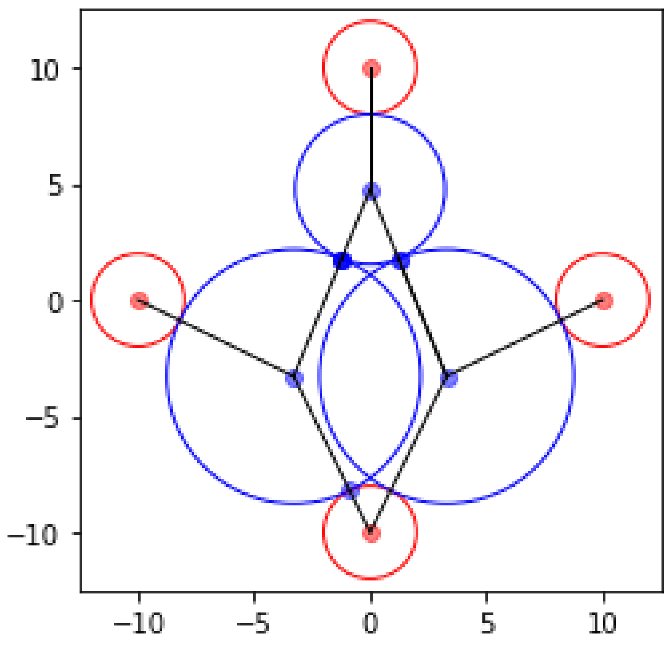
[4]:
```
prediction = np.zeros(shape=(m, 3))
i = 0
for j in range(n):
    for k in range(j + 1, n):
        prediction[i, :] = [j, k, np.sign(a_hat[j] - a_hat[k])]
        i = i + 1
print(
    " prediction accuracy:",
    round(100 * sum(prediction[:, 2] == test[:, 2]) / m, 2),
    "%",
    "\n biased prediction accuracy:",
    100 * round(sum(train[:, 2] == test[:, 2]) / m, 2),
    "%",
)
```

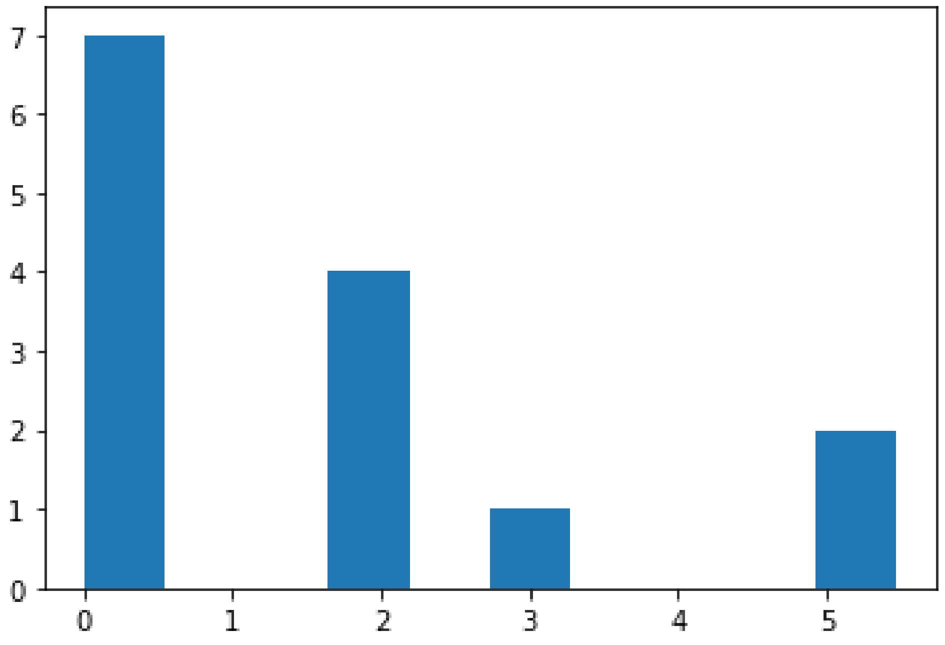
```
prediction accuracy: 75.56 %
biased prediction accuracy: 71.0 %
```

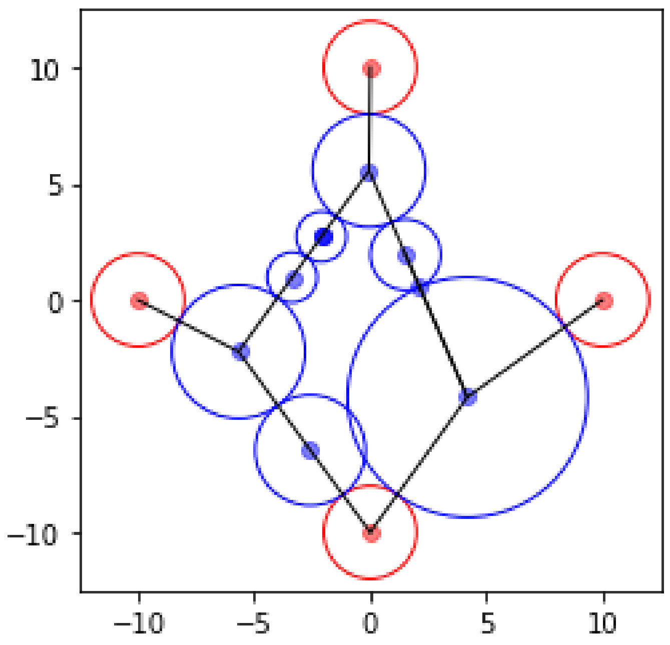
1.1.3 A7.23

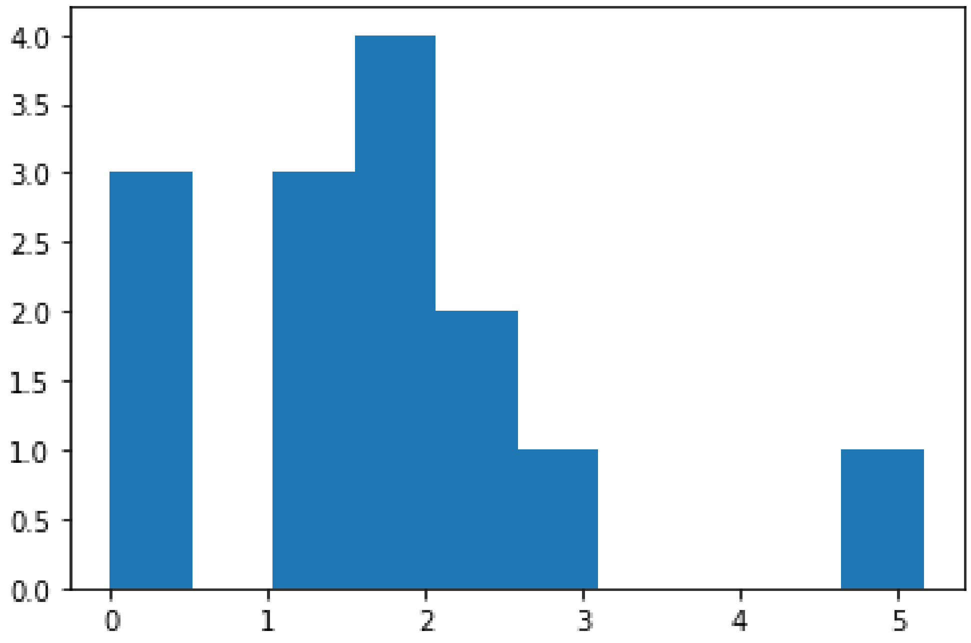
[5]:
```
from disks_data import *

C = cp.Variable(shape=(n, 2))
R = cp.Variable(n)
constraint = [C[:k] == Cgiven, R[:k] == Rgiven, R >= 0]
constraint += [
    cp.norm(C[row[0, 0]] - C[row[0, 1]]) <= R[row[0, 0]] + R[row[0, 1]]
    for row in Gindexes
]
objective1 = cp.Minimize(cp.sum(R))
objective2 = cp.Minimize(cp.sum(R ** 2))
problem1 = cp.Problem(objective1, constraint)
problem2 = cp.Problem(objective2, constraint)
problem1.solve()
plot_disks(C.value, R.value, Gindexes.astype(int))
plt.hist(R.value)
problem2.solve();
plot_disks(C.value, R.value, Gindexes.astype(int))
plt.hist(R.value);
```


10
5
0
−5
−10
−10
−5
0
5
10


7
6
5
4
3
2
1
0
0
1
2
3
4
5





As we can see in histograms, fewer radiuses are near zero in the second histogram but instead, the maximum radius is less than the first solution. In the first solution there are more radiuses equal to zero.

1.1.4 A14.30

```
[6]: from currency_exchange_data import *

X = cp.Variable(shape=(n, n))
values = np.zeros(n)
for i in range(n):
    values[i] = np.sqrt(F[i, 0] / F[0, i])

c_f = c_init - cp.sum(X, 0) + cp.diag(X @ (1 / F.T))
constraint = [cp.diag(X) == 0, c_init >= cp.sum(X, 0), c_req <= c_f , X>=0]
objective = cp.Maximize(values @ c_f)
problem = cp.Problem(objective, constraint)
print("optimal cost is:",round(c_init @ values - problem.solve(),2))
```

```
optimal cost is: 7.72
```