ESE 605 Homework 3

Xi Cao

Mar. 25, 2022

CVX* warmup:

```
# Import packages.
import cvxpy as cp
import numpy as np

# Generate a random feasible SOCP.
m = 3
n = 10
p = 5
n_i = 5
np.random.seed(2)
f = np.random.randn(n)
A = []
b = []
c = []
d = []
x0 = np.random.randn(n)
for i in range(m):
    A.append(np.random.randn(n_i, n))
    b.append(np.random.randn(n_i))
    c.append(np.random.randn(n))
    d.append(np.linalg.norm(A[i] @ x0 + b, 2) - c[i].T @ x0)
F = np.random.randn(p, n)
g = F @ x0

# Define and solve the CVXPY problem.
x = cp.Variable(n)
# We use cp.SOC(t, x) to create the SOC constraint ||x||_2 <= t.
soc_constraints = [
      cp.SOC(c[i].T @ x + d[i], A[i] @ x + b[i]) for i in range(m)
]
prob = cp.Problem(cp.Minimize(f.T@x),
                  soc_constraints + [F @ x == g])
```

```python
prob.solve()

# Print result.
print("The optimal value is", prob.value)
print("A solution x is")
print(x.value)
for i in range(m):
    print("SOC constraint %i dual variable solution" % i)
    print(soc_constraints[i].dual_value)
```

```
The optimal value is -9.582695716266176
A solution x is
[ 1.40303325  2.4194569   1.69146656 -0.26922215  1.30825472 -0.70834842
  0.19313706  1.64153496  0.47698583  0.66581033]
SOC constraint 0 dual variable solution
[ 0.61662526  0.35370661 -0.02327185  0.04253095  0.06243588  0.49886837]
SOC constraint 1 dual variable solution
[ 0.35283078 -0.14301082  0.16539699 -0.22027817  0.15440264  0.06571645]
SOC constraint 2 dual variable solution
[ 0.86510445 -0.114638   -0.449291    0.37810251 -0.6144058  -0.11377797]
```

Problems from Boyd & Vandenberghe:
4.26.

$$\left\| \begin{bmatrix} 2x \\ y-z \end{bmatrix} \right\|_2^2 - (y+z)^2 = 4x^Tx + y^2 + z^2 + 2yz - (y+z)^2 = 4(x^Tx - yz)$$

Since $\left\| \begin{bmatrix} 2x \\ y-z \end{bmatrix} \right\|_2 \geq 0$, and $y+z \geq 0$, then $\left\| \begin{bmatrix} 2x \\ y-z \end{bmatrix} \right\|_2^2 \leq (y+z)^2$ is equivalent to $\left\| \begin{bmatrix} 2x \\ y-z \end{bmatrix} \right\|_2 \leq y+z$, and is equivalent to $x^Tx \leq yz$

(a) The problem is equivalent to:

$$\begin{aligned} \text{min.} \quad & \sum_{i=1}^{n} t_i \\ \text{s.t.} \quad & t_i \geq \frac{1}{a_i^Tx - b_i} \\ & t_i \geq 0 \end{aligned}$$

Writing the constrains in the form of:

$$x^Tx \leq yz, \quad y \geq 0, \quad z \geq 0$$

where

$$x = 1, y = t_i, z = a_i^Tx - b_i$$

the problem can be casted as SOCP:

$$\begin{array}{rl} \text{min.} & 1^T t \\ \text{s.t.} & \left\| \begin{bmatrix} 2 \\ t_i - a_i^T x + b_i \end{bmatrix} \right\|_2 \leq t_i + a_i^T x - b_i \\ & t_i \geq 0 \\ & a_i^T x - b_i \geq 0 \end{array}$$

(b) solve only for m = 6 case.

The objective function is equivalent to:

$$\text{max.} \quad \Pi_{i=1}^{6} a_i^T x + b_i$$

According to the hint on Piazza, without loss of generality, we can define $a_7 = a_8 = 0$ and $b_7 = b_8 = 1$ to make this a m = 8 case. We can rewrite the problem as:

$$\begin{array}{rl} \text{max.} & t_1 t_2 t_3 t_4 \\ \text{s.t.} & Ax \succeq b \\ & t_1^2 \leq (a_1^T x - b_1)(a_2^T x - b_2) \\ & t_2^2 \leq (a_3^T x - b_3)(a_4^T x - b_4) \\ & t_3^2 \leq (a_5^T x - b_5)(a_6^T x - b_6) \\ & t_4^2 \leq (a_7^T x - b_7)(a_8^T x - b_8) \end{array}$$

which is equivalent to:

$$\begin{array}{rl} \text{max.} & p_1 p_2 \\ \text{s.t.} & Ax \succeq b \\ & t_1^2 \leq (a_1^T x - b_1)(a_2^T x - b_2) \\ & t_2^2 \leq (a_3^T x - b_3)(a_4^T x - b_4) \\ & t_3^2 \leq (a_5^T x - b_5)(a_6^T x - b_6) \\ & t_4^2 \leq (a_7^T x - b_7)(a_8^T x - b_8) \\ & p_1^2 \leq t_1 t_2 \\ & p_2^2 \leq t_3 t_4 \end{array}$$

which is equivalent to:

$$\begin{aligned} \text{max.} \quad & k \\ \text{s.t.} \quad & Ax \succeq b \\ & t_1^2 \le (a_1^T x - b_1)(a_2^T x - b_2) \\ & t_2^2 \le (a_3^T x - b_3)(a_4^T x - b_4) \\ & t_3^2 \le (a_5^T x - b_5)(a_6^T x - b_6) \\ & t_4^2 \le (a_7^T x - b_7)(a_8^T x - b_8) \\ & p_1^2 \le t_1 t_2 \\ & p_2^2 \le t_3 t_4 \\ & k^2 \le p_1 p_2 \end{aligned}$$

so the problem can be casted as SOCP:

$$\begin{aligned} \text{min.} \quad & -k \\ \text{s.t.} \quad & \left\| \begin{bmatrix} 2t_1 \\ a_1^T x - b_1 - a_2^T x + b_2 \end{bmatrix} \right\|_2 \le a_1^T x - b_1 + a_2^T x - b_2 \\ & \left\| \begin{bmatrix} 2t_2 \\ a_3^T x - b_3 - a_4^T x + b_4 \end{bmatrix} \right\|_2 \le a_3^T x - b_3 + a_4^T x - b_4 \\ & \left\| \begin{bmatrix} 2t_3 \\ a_5^T x - b_5 - a_6^T x + b_6 \end{bmatrix} \right\|_2 \le a_5^T x - b_5 + a_6^T x - b_6 \\ & \left\| \begin{bmatrix} 2t_4 \\ a_7^T x - b_7 - a_8^T x + b_8 \end{bmatrix} \right\|_2 \le a_7^T x - b_7 + a_8^T x - b_8 \\ & \left\| \begin{bmatrix} 2p_1 \\ t_1 - t_2 \end{bmatrix} \right\|_2 \le t_1 + t_2 \\ & \left\| \begin{bmatrix} 2p_2 \\ t_3 - t_4 \end{bmatrix} \right\|_2 \le t_3 + t_4 \\ & \left\| \begin{bmatrix} 2k \\ p_1 - p_2 \end{bmatrix} \right\|_2 \le p_1 + p_2 \\ & ax_i - b_i \ge 0, t_i \ge 0, p_i \ge 0 \end{aligned}$$

4.43.

(a) The original problem is:

$$\text{min.} \quad \lambda_1(x)$$

We know that $\lambda I - A(x) \succeq 0$ iff $\lambda - \lambda_1(x) \ge 0$. So the problem can be written in the form of SDP as:

$$\begin{aligned} \text{min.} \quad & \lambda \\ \text{s.t.} \quad & A(x) \preceq \lambda I \end{aligned}$$

(b) The original problem is:

$$\text{min.} \quad \lambda_1(x) - \lambda_m(x)$$

We know that $\gamma I - A(x) \preceq 0$ iff $\gamma - \lambda_m(x) \leq 0$. So the problem can be written in the form of SDP as:

$$\begin{aligned} \text{min.} \quad & \lambda - \gamma \\ \text{s.t.} \quad & A(x) \preceq \lambda I \\ & A(x) \succeq \gamma I \end{aligned}$$

(c) From (a) and (b), we know that the problem:

$$\begin{aligned} \text{min.} \quad & \lambda_1(x)/\lambda_m(x) \\ \text{s.t.} \quad & A(x) \succ 0 \end{aligned}$$

is equivalent to:

$$\begin{aligned} \text{min.} \quad & \lambda/\gamma \\ \text{s.t.} \quad & 0 \prec \gamma I \preceq A(x) \preceq \lambda I \end{aligned}$$

Change variables to $y = x/\gamma, t = \lambda/\gamma, s = 1/\gamma$.

If $\gamma > 0$, without loss of generality, the problem becomes:

$$\begin{aligned} \text{min.} \quad & t \\ \text{s.t.} \quad & I \preceq sA(0) + y_1 A(1) + \cdots + y_n A(n) \preceq tI \\ & s \geq 0 \end{aligned}$$

We want to show the above form holds when $\gamma = 0$.

If $\gamma = 0$, the constraint becomes $I \preceq y_1 A(1) + \cdots + y_n A(n) \preceq tI$. We know that the solution of the former problem is always feasible for the SDP problem, so that $p_1^* \geq p_2^*$. We want to show that $p_2^* \geq p_1^*$.

Since $A(x) \succ 0$ for at least one x, we can construct $A(\tau y) \succeq A0 + \tau I \succ 0$. Then, $\lambda_1(\tau y) \leq \lambda_1(0) + t\tau, \lambda_m(\tau y) \geq \lambda_m(0) + \tau$, when τ is sufficiently large,

$$\kappa(A(x)) = \frac{\lambda_1(x)}{\lambda_m(x)} \leq \frac{\lambda_1(0) + t\tau}{\lambda_m(0) + \tau}$$

so that $p_2^* \geq p_1^*$.

Now we have proved the problem can be written as SDP.

4.45.

(a) If p can be expressed as a positive semidefinite quadratic form $p = f^T V f$.

Since $V \in S_+^s$, we can write $V = WW^T$, where W is $s \times r$, and

$$p = (W^T f)^T W^T f = \sum_{i=1}^{r} (W_i f)^2$$

where degree of $W_i f$ is no more than k. So p is SOS.

If p is SOS, then p have the form $p = \sum_{i=1}^{r} q_i(x)^2$. We can separate the monomials and coefficients and have,

$$p = \sum_{i=1}^{r} q_i(x)^2 = \sum_{i=1}^{r} (W_i f)^2 = (W^T f)^T W^T f$$

Set $V = WW^T$, then p can be expressed as a positive semidefinite quadratic form $p = f^T V f$, and $V \in S_+^s$.

(b) We can spread out the condition,

$$p = f^T V f = \sum_{i,j=1}^{s} V_{ij} f_i f_j$$

so that p is a set of linear equality constraints relating the coefficients of p and the matrix V. Since p is SOS requires all the coefficients to be non-negative, $V \succeq 0$.

(c) Set $f = [1 \ x_1 \ x_2 \ x_1^2 \ x_2^2 \ x_1x_2]^T$, then

$$f^T f = \begin{bmatrix} 1 & x_1 & x_2 & x_1^2 & x_2^2 & x_1x_2 \\ x_1 & x_1^2 & x_1x_2 & x_1^3 & x_1x_2^2 & x_1^2x_2 \\ x_2 & x_1x_2 & x_2^2 & x_1^2x_2 & x_2^3 & x_1x_2^2 \\ x_1^2 & x_1^3 & x_1^2x_2 & x_1^4 & x_1^2x_2^2 & x_1^3x_2 \\ x_2^2 & x_1x_2^2 & x_2^3 & x_1^2x_2^2 & x_2^4 & x_1x_2^3 \\ x_1x_2 & x_1^2x_2 & x_1x_2^2 & x_1^3x_2 & x_1x_2^3 & x_1^2x_2^2 \end{bmatrix}$$

We can write p as:

$$\begin{aligned} p = {} & a_1 + a_2x_1 + a_3x_2 + a_4x_1^2 + a_5x_1x_2 + a_6x_2^2 + a_7x_1^3 + a_8x_1^2x_2 \\ & + a_9x_1x_2^2 + a_{10}x_2^3 + a_{11}x_1^4 + a_{12}x_1^3x_2 + a_{13}x_1^2x_2^2 + a_{14}x_1x_2^3 + a_{15}x_2^4 \end{aligned}$$

Since $p = f^T V f = \sum_{i,j=1}^{s} V_{ij} f_i f_j$,

$$\begin{gathered} a_1 = V_{11}, \quad a_2 = V_{12} + V_{21}, \quad a_3 = V_{13} + V_{31} \\ a_4 = V_{14} + V_{22} + V_{41}, \quad a_5 = V_{16} + V_{23} + V_{32} + V_{61}, \quad a_6 = V_{15} + V_{33} + V_{51} \\ a_7 = V_{24} + V_{42}, \quad a_8 = V_{26} + V_{34} + V_{43} + V_{62}, \quad a_9 = V_{25} + V_{36} + V_{52} + V_{63} \\ a_{10} = V_{35} + V_{53}, \quad a_{11} = V_{44}, \quad a_{12} = V_{46} + V_{64} \\ a_{13} = V_{45} + V_{54} + V_{66}, \quad a_{14} = V_{56} + V_{65}, \quad a_{15} = V_{55} \end{gathered}$$

We have worked out the LMI conditions for SOS explicitly.

4.59.

(a) If f_i are convex in x for each u.

$$\mathbf{E}_u f_i(x,u) = \int f_i(x,u)p(u)du$$

Since $p(u) > 0$, $\mathbf{E}_u f_i(x,u)$ is convex. So this stochastic optimization problem is convex.

(b) If $f_i(x,u)$ is convex in x for each $u \in U$, then

$$\sup_{u \in U} f_i(x,u)$$

is convex. So this stochastic optimization problem is convex.

(c) Set up the stochastic optimization problems:

$$\begin{aligned} \text{min.} \quad & \sum_{j=i}^{N} f_0(x,u_j)p_j \\ \text{s.t.} \quad & \sum_{j=i}^{N} f_i(x,u_j)p_j \leq 0, \quad i = 1, \cdots, m \end{aligned}$$

Set up the worst-case optimization problems:

$$\begin{aligned} \text{min.} \quad & \max_j f_0(x,u_j) \\ \text{s.t.} \quad & \max_j f_i(x,u_j) \leq 0, \quad i = 1, \cdots, m, \; j = 1, \cdots, N \end{aligned}$$

Disciplined Convex Programming:
A4.3.

(a)
```
cp.inv_pos(x) + cp.inv_pos(y) <= 1, x >= 0, y >= 0
```

(b)
```
x >= cp.inv_pos(y), x >= 0, y >= 0
```

(c)
```
cp.quad_over_lin(x+y, cp.sqrt(y)) <= x - y + 5
```

(d)
```
x + z <= 1 + cp.geo_mean(cp.hstack([x - cp.quad_over_lin(z,y), y])),
x >= 0, y >= 0
```

Codes of small problems are listed in Appendix A.

Problems from Boyd & Vandenberghe:
5.3.

Lagrangian is:

$$L(x, \lambda) = c^T x + \lambda f(x)$$

Dual function is:

$$\begin{aligned} g(\lambda) &= \inf_x L(x, \lambda) \\ &= \lambda \inf_x (c/\lambda)^T x + f(x) \\ &= -\lambda \sup_x -(c/\lambda)^T x - f(x) \\ &= -\lambda f^*(-c/\lambda) \end{aligned}$$

Dual problem is:

$$\begin{aligned} \text{max.} \quad & -\lambda f^*(-c/\lambda) \\ \text{s.t.} \quad & \lambda \geq 0 \end{aligned}$$

Since conjugate function f^* is convex, the objective is maximize over a concave function. Also, constraint is convex. So this is a convex problem.

5.4.

(a) Lagrangian is:

$$L(x, \lambda) = c^T x + \lambda(w^T A x - w^T b)$$

Dual function is:

$$\begin{aligned} g(\lambda) &= \inf_x L(x, \lambda) \\ &= \inf_x -\lambda w^T b + (c + \lambda A^T w)^T x \\ &= \begin{cases} -\lambda w^T b & c + \lambda A^T w = 0 \\ -\infty & \text{otherwise} \end{cases} \end{aligned}$$

The minimum value of $c^T x$ is:

$$\min(c^T x) = \begin{cases} -\lambda w^T b & c + \lambda A^T w = 0 \\ -\infty & \text{otherwise} \end{cases}$$

(b) The dual problem is:

$$\begin{aligned} \text{max.} \quad & -\lambda w^T b \\ \text{s.t.} \quad & c + \lambda A^T w = 0 \\ & \lambda \geq 0 \\ & w \succeq 0 \end{aligned}$$

(c) Set $\lambda w = \lambda$. In this situation, qualify the new $\lambda \succeq 0$ is equivalent to qualify old $\lambda \geq 0,\ w \succeq 0$. Then the problem can be written as:

$$\begin{aligned} \text{max.} \quad & -b^T\lambda \\ \text{s.t.} \quad & A^T\lambda + c = 0 \\ & \lambda \succeq 0 \end{aligned}$$

5.9.

(a) Consider the following matrix.

$$\begin{aligned} & \begin{bmatrix} \sum_{k=1}^m a_k a_k^T & a_i \\ a_i^T & 1 \end{bmatrix} \\ = & \begin{bmatrix} I & a_i \\ 0 & I \end{bmatrix} \begin{bmatrix} \sum_{k=1,k\neq i}^m a_k a_k^T & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} I & a_i \\ 0 & I \end{bmatrix}^T \\ \succeq & \ 0 \end{aligned}$$

The Schur complements of a positive semi-definite matrix is positive semi-definite,

$$\begin{aligned} S &= 1 - a_i^T(\sum_{k=1}^m a_k a_k^T)^{-1} a_i \\ &= 1 - a_i^T X^{-1} a_i \\ &\geq 0 \end{aligned}$$

So that $a_i^T X^{-1} a_i \leq 1$. We have shown the problem is feasible.

(b) When $\lambda = t\mathbf{1} \in R^m$, the dual problem is:

$$\begin{array}{ll} \text{maximize} & \log\det\left(\sum_{i=1}^m a_i a_i^T\right) + \log t^n - mt + n \\ \text{subject to} & t > 0 \end{array}$$

Since,

$$\frac{d}{dt}(\log t^n - mt) = n/t - m$$

The optimal value of t is n/m. Put this back into the dual problem,

$$\begin{aligned} & \log\det\left(\sum_{i=1}^m a_i a_i^T\right) + \log t^n - mt + n \\ &= \log\det\left(\sum_{i=1}^m a_i a_i^T\right) + n\log(n/m) \\ &= \log\det\left(\sum_{i=1}^m a_i a_i^T\right) - n\log(m/n) \end{aligned}$$

The value of the primal problem is:

$$\log\det\left(\sum_{i=1}^{m} a_i a_i^T\right)$$

The volume of a ellipsoid is $(\det X^{-1})^{1/2}$.

The values of the primal and the dual problem are:

$$Y_1 = \log(\det(X^{-1}))$$
$$Y_2 = \log(\det(X^{-1})) - n\log(m/n)$$

We know the gap between the dual and primal is no more than $n\log(m/n)$, and the volume of a ellipsoid is $\exp(Y/2)$. So the the difference between the ellipsoid $\{u|u^T X_{sim} u \leq 1\}$ and the volume of the minimum volume ellipsoid is no more than a factor $(m/n)^{n/2}$.

Problem from Additional Exercises:
A5.3.

Lagrangian is:

$$L(x,\nu,\lambda) = \sum_{k=1}^{n} x_k \log(x_k/y_k) + \nu^T(Ax-b) + \lambda(1^T x - 1)$$

Dual function is:

$$\begin{aligned} g(\nu,\lambda) &= \inf_x L(x,\nu,\lambda) \\ &= \inf_x \sum_{k=1}^{n} x_k \log(x_k/y_k) + \nu^T(Ax-b) + \lambda(1^T x - 1) \end{aligned}$$

Try to solve this by taking the derivative of L:

$$\begin{aligned} &\frac{\partial}{\partial x_k}\sum_{k=1}^{n} x_k \log(x_k/y_k) + \nu^T(Ax-b) + \lambda(1^T x - 1) \\ &= \log x_k + 1 - \log y_k + \nu^T a_k + \lambda \end{aligned}$$

The optimal value for x_k is:

$$x_k = y_k e^{-(1+\nu^T a_k + \lambda)}$$

Put back into the dual function:

$$\begin{aligned} -\nu^T b - \log\sum_{k=1}^{n} y_k e^{a_k^T \nu} + 1 &= -\sum_{k=1}^{n}(1+\nu^T a_k+\lambda)x_k + \nu^T(Ax-b) + \lambda(1^T x - 1) \\ &= -\sum_{k=1}^{n} y_k e^{-(1+\nu^T a_k+\lambda)} - \nu^T b - \lambda \end{aligned}$$

Taking the derivative of $g(\nu, \lambda)$ by λ,

$$\frac{\partial}{\partial \lambda} = \sum_{k=1}^{n} y_k e^{-(1+\nu^T a_k + \lambda)} - 1$$

So that

$$\lambda = \log \sum_{k=1}^{n} y_k e^{-a_k^T \nu} - 1$$

Set $\nu = -z$. Then,

$$\begin{aligned} g(\nu, \lambda) &= -\sum_{k=1}^{n} y_k e^{-1} \sum_{k=1}^{n} y_k e^{-\nu^T a_k} \sum_{k=1}^{n} y_k e^{-\lambda} - \nu^T b - \log \sum_{k=1}^{n} y_k e^{a_k^T \nu} + 1 \\ &= -1 - \nu^T b - \log \sum_{k=1}^{n} y_k e^{a_k^T z} + 1 \\ &= b^T z - \log \sum_{k=1}^{n} y_k e^{a_k^T z} \end{aligned}$$

A Appendix: Codes for A4.3

```
import cvxpy as cp
```

```
# A4.3(a)
x, y = cp.Variable(), cp.Variable()
objective = cp.Minimize(x+y)
constraints = [cp.inv_pos(x) + cp.inv_pos(y) <= 1, x >= 0, y >= 0]
prob = cp.Problem(objective, constraints)
prob.solve()
print("status:", prob.status)
print("optimal value", prob.value)
print("optimal var", x.value, y.value)
```

```
status: optimal
optimal value 3.9999999942618447
optimal var 1.999999997130576 1.9999999971312687
```

```
# A4.3(b)
x, y = cp.Variable(), cp.Variable()
objective = cp.Minimize(x+y)
constraints = [x >= cp.inv_pos(y), x >= 0, y >= 0]
prob = cp.Problem(objective, constraints)
prob.solve()
print("status:", prob.status)
print("optimal value", prob.value)
print("optimal var", x.value, y.value)
```

```
status: optimal
optimal value 1.9999999990988524
optimal var 1.0000014640195074 0.9999985350793449
```

```
# A4.3(c)
x, y = cp.Variable(), cp.Variable()
objective = cp.Minimize(x+y)
constraints = [cp.quad_over_lin(x+y, cp.sqrt(y)) <= x - y + 5]
prob = cp.Problem(objective, constraints)
prob.solve()
print("status:", prob.status)
print("optimal value", prob.value)
print("optimal var", x.value, y.value)
```

```
status: optimal
optimal value -1.3715395933070145
optimal var -1.9762812107395356 0.6047416174325209
```

```
#·A4.3(d)
x,·y,·z·=·cp.Variable(),·cp.Variable(),·cp.Variable()
objective·=·cp.Minimize(x+y+z)
constraints·=·[x·+·z·<=·1·+·cp.geo_mean(cp.hstack([x·-·cp.quad_over_lin(z,y),·y])),·x·
```

```
prob = cp.Problem(objective, constraints)
prob.solve()
print("status:", prob.status)
print("optimal value", prob.value)
print("optimal var", x.value, y.value, z.value)
```

```
status: optimal
optimal value 1.1058302763345625e-10
optimal var -1.068114553406185e-10 5.684880922432107e-10 -3.510936092691359e-10
```

✓ 0s completed at 2:24 PM