



Introduction to conic optimization

June 20th 2016 - June 22th 2016

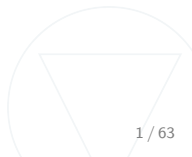
e.d.andersen@mosek.com

www.mosek.com





- Semidefinite optimization.
- Topological properties
- Duality revisited.
- Complementarity.



Section 1

Semidefinite optimization





Definition

$$\mathcal{K}_s := \{X \in \mathbb{R}^{n \times n} \mid X = X^T, \lambda_{\min}(X) \geq 0\}$$

- The cone of symmetric positive semidefinite matrices.
- Is a cone?

Proof.

\mathcal{K}_s is convex. Since, $X, Y \in \mathcal{K}_s$ implies

$$v^T(\lambda X + (1 - \lambda)Y)v \geq 0$$

for all v and $0 \leq \lambda \leq 1$. Moreover, $\alpha X \in \mathcal{K}_s$ for all $\alpha \geq 0$. □



$$\begin{aligned} &\text{minimize} && \sum_{j=1}^n c_j x_j + \sum_{j=1}^{\bar{n}} \langle \bar{C}_j, \bar{X}_j \rangle \\ &\text{subject to} && \sum_{j=1}^n a_{ij} x_j + \sum_{j=1}^{\bar{n}} \langle \bar{A}_{ij}, \bar{X}_j \rangle = b_i, \quad i = 1, \dots, m, \\ &&& x \in \mathcal{K}, \\ &&& \bar{X}_j \succeq 0, \quad j = 1, \dots, \bar{n}. \end{aligned}$$

Explanation:

- x_j is a scalar variable.
- \bar{X}_j is a square matrix variable.

- \mathcal{K} represents Cartesian product of conic quadratic constraints.
- \overline{C}_j and \overline{A}_j are required to be symmetric.
- Inner product between matrices

$$\begin{aligned}\langle A, B \rangle &= \text{tr}(A^T B) \\ &= \sum_i \sum_j A_{ij} B_{ij}.\end{aligned}$$

- Linear constraints + a conic constraint = conic optimization problem.





The nearest correlation matrix

X is a correlation matrix if

$$X \in \mathcal{C} := \{X \in K_s \mid \text{diag}(X) = e\}.$$

Links:

- https://en.wikipedia.org/wiki/Correlation_and_dependence
- <https://nickhigham.wordpress.com/2013/02/13/the-nearest-correlation-matrix/>

Higham:

A correlation matrix is a symmetric matrix with unit diagonal and nonnegative eigenvalues. In 2000 I was approached by a London fund management company who wanted to find the nearest correlation matrix (NCM) in the Frobenius norm to an almost correlation matrix: a symmetric matrix having a significant number of (small) negative eigenvalues. This problem arises when the data from which the correlations are constructed is asynchronous or incomplete, or when models are stress-tested by artificially adjusting individual correlations. Solving the NCM problem (or obtaining a true correlation matrix some other way) is important in order to avoid subsequent calculations breaking down due to negative variances or volatilities, for example.



The Frobenius norm

$$\|A\|_F := \sqrt{\sum_i \sum_j A_{ij}^2}.$$

Given a symmetric matrix A then the problem is

$$\begin{array}{ll} \text{minimize} & \|A - X\|_F \\ \text{subject to} & X \in \mathcal{K}_s. \end{array}$$



The problem

$$\begin{array}{ll} \text{minimize} & t \\ \text{subject to} & (t; \text{vec}(A - X)) \in \mathcal{K}_q, \\ & \text{diag}(X) = e, \\ & X \succeq 0, \end{array}$$

where

$$\text{vec}(U) = (U_{11}; \sqrt{2}U_{21}; \dots, \sqrt{2}U_{n1}; U_{22}; \sqrt{2}U_{32}; \dots, \sqrt{2}U_{n2}; \dots; U_{nn})^T.$$



Python program

nearestcor.py



```
import sys
import mosek
import mosek.fusion
from mosek.fusion import *
from mosek import LinAlg
from math import sqrt

def vec(e):
    """
    Assuming that e is an NxN expression, return the lower triangular part as a vector.
    """
    N = e.getShape().dim(0)

    rows = [i for i in range(N) for j in range(i,N)]
    cols = [j for i in range(N) for j in range(i,N)]
    vals = [2.0**0.5 if i!=j else 1.0 for i in range(N) for j in range(i,N)]

    return Expr.flatten(Expr.mulElem(e, Matrix.sparse(N,N,rows,cols,vals)))

def nearestcorr(A):
    N = A.numRows()

    # Create a model with the name 'NearestCorrelation
    with Model("NearestCorrelation") as M:

        # Setting up the variables
        X = M.variable("X", Domain.inPSDCone(N))
        t = M.variable("t", 1, Domain.unbounded())

        # (t, vec (A-X)) \in Q
        M.constraint("C1", Expr.vstack(t, vec(Expr.sub(A,X))), Domain.inQCone() )

        # diag(X) = e
        M.constraint("C2", X.diag(), Domain.equalsTo(1.0))

        # Objective: Minimize t
        M.objective(ObjectiveSense.Minimize, t)
        M.solve()

    return X.level(), t.level()

if __name__ == '__main__':
    N = 5

    A = Matrix.dense(N,N,[ 0.0, 0.5, -0.1, -0.2, 0.5,
                           0.5, 1.25, -0.05, -0.1, 0.25,
                           -0.1, -0.05, 0.51, 0.02, -0.05,
                           -0.2, -0.1, 0.02, 0.54, -0.1,
                           0.5, 0.25, -0.05, -0.1, 1.25])

    X,t = nearestcorr(A)

    print("--- Nearest Correlation ---")
    print("X = ",X)
    print("t = ",t)
```



Consider a binary problem (Q possible indefinite)

$$\begin{array}{ll}\text{minimize} & x^T Q x + c^T x \\ \text{subject to} & x_i \in \{0, 1\}, \quad i = 1, \dots, n.\end{array}$$

Rewrite binary constraints $x_i \in \{0, 1\}$:

$$x_i^2 = x_i \quad \Longleftrightarrow \quad X = x x^T, \quad \text{diag}(X) = x.$$

Observe

$$x^T Q x = \langle Q, X \rangle.$$

Still non-convex, since $\text{rank}(X) = 1$.

Clearly,

$$X - xx^T \succeq 0$$

is relaxation of

$$X - xx^T = 0.$$

Also we have

$$X - xx^T \succeq 0 \Leftrightarrow \begin{pmatrix} X & x \\ x^T & 1 \end{pmatrix} \succeq 0.$$





Lifted non-convex problem:

$$\begin{array}{ll}\text{minimize} & \langle Q, X \rangle + c^T x \\ \text{subject to} & \text{diag}(X) - x = 0, \\ & X - xx^T = 0.\end{array}$$

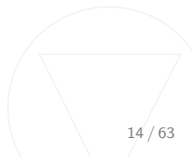
Semidefinite relaxation:

$$\begin{array}{ll}\text{minimize} & \langle Q, X \rangle + c^T x \\ \text{subject to} & \text{diag}(X) - x = 0, \\ & \begin{pmatrix} X & x \\ x^T & 1 \end{pmatrix} \succeq 0.\end{array}$$

- Relaxation is exact if the optimal solution satisfies $X = xx^T$.
- Can be strengthened, e.g., by adding $X_{ij} \geq 0$.
- Typical relaxation for combinatorial optimization.



- The linear and quadratic cones are special cases of the semidefinite cone.
- The semidefinite cone is a very powerful modeling construct.
- The semidefinite cone has a serious drawback.
 - Any suggestions for what that might be?





Estimate of an unknown convex density function

$$g : \mathbb{R}_+ \rightarrow \mathbb{R}_+.$$

- Let Y be the real-valued random variable with density function g .
- Let y_1, \dots, y_n be an ordered sample of n outcomes of Y .
- Assume $y_1 < y_2 < \dots < y_n$.
- The estimator of $\tilde{g} \geq 0$ is a piecewise linear function

$$\tilde{g} : [y_1, y_n] \rightarrow \mathbb{R}_+$$

with break points at $(y_i, \tilde{g}(y_i)), i = 1, \dots, n$.



Let

$$x_i > 0, i = 1, \dots, n,$$

be the estimator of $g(y_i)$.

The slope for i th segment is given by

$$\frac{x_{i+1} - x_i}{y_{i+1} - y_i}.$$

Hence the convexity requirement is

$$\frac{x_{i+1} - x_i}{y_{i+1} - y_i} \leq \frac{x_{i+2} - x_{i+1}}{y_{i+2} - y_{i+1}}, \forall i = 1, \dots, n - 2.$$

Recall the area under density function must be 1. Hence,

$$\sum_{i=1}^{n-1} (y_{i+1} - y_i) \left(\frac{x_{i+1} + x_i}{2} \right) = 1$$

must hold.

The problem

$$\begin{aligned} & \text{maximize} && \prod_{i=1}^n x_i \\ & \text{subject to} && \frac{x_{i+1} - x_i}{y_{i+1} - y_i} - \frac{x_{i+2} - x_{i+1}}{y_{i+2} - y_{i+1}} \leq 0 \quad \forall i = 1, \dots, n-2, \\ & && \sum_{i=1}^{n-1} (y_{i+1} - y_i) \left(\frac{x_{i+1} + x_i}{2} \right) = 1, \\ & && x \geq 0. \end{aligned}$$



The problem

$$\begin{aligned} &\text{maximize} && \left(\prod_{i=1}^n x_i \right)^{\frac{1}{n}} \\ &\text{subject to} && -\Delta y_{i+1} x_i \\ & && +(\Delta y_i + \Delta y_{i+1}) x_{i+1} \\ & && -\Delta y_i x_{i+2} \leq 0 \quad \forall i = 1, \dots, n-2, \\ & && \sum_{i=1}^{n-1} \Delta y_i \left(\frac{x_{i+1} + x_i}{2} \right) = 1, \\ & && x \geq 0. \end{aligned}$$

where

$$\Delta y_i = y_{i+1} - y_i.$$



Lemma

For $l = 1, 2, \dots$ and $n = 2^l$ and $g \in \mathbb{R}_+^{2n-1}$. Given

$$(g_{2i}, g_{2i+1}, g_i) \in \mathcal{K}_r, \text{ for } i = 1, \dots, n-1.$$

then

$$\sqrt{n}^n \prod_{i=n}^{2n-1} g_i \geq g_1^n$$

A fact:

$$\sum_{i=0}^l 2^i = 2n - 1.$$





```
import numpy
import math
import random
import sys

import mosek

from mosek.fusion import *

import gmmeancone

def buildandsolve(y): # y[i+1]-y[i]>0
    with Model("Max likelihood") as M:

        #M.setLogHandler(sys.stdout) # Make sure we get some output

        n = len(y)

        t = M.variable('t', 1, Domain.unbounded())
        x = M.variable('x', n, Domain.greaterThan(0.0))

        dy = [y[i+1]-y[i] for i in range(0,n-1)]

        eleft = Expr.mulElm(dy[1:n-1],x.slice(0,n-2))
        emid = Expr.add(Expr.mulElm(dy[0:n-2],x.slice(1,n-1)),Expr.mulElm(dy[1:n-1],x.slice(1,n-1)))
        eright = Expr.mulElm(dy[0:n-2],x.slice(2,n))

        # Debug print: print(eleft.toString())

        M.constraint('convex',Expr.sub(Expr.sub(emid,eleft),eright),Domain.equalsTo(0.0))
        M.constraint('area',Expr.mul(0.5,Expr.dot(dy,Expr.add(x.slice(0,n-1),x.slice(1,n))))),Domain.equalsTo(1.0))

        gmmeancone.appendcone(M,t,x)

        M.objective(ObjectiveSense.Maximize, t)

        M.solve()

    return x.level()
```

The geometric mean cone

gmmeancone.py



```
import math

import mosek
from mosek.fusion import *

def appendcone(M,t,x):
    # t is scalar variable
    # x is n dimensional variable

    lenx = x.size()
    l = 0
    while 2**l < lenx:
        l = l+1

    n = 2**l
    d = 2*n-1

    idx1 = range(1,d,2)
    idx2 = range(2,d,2)
    idx3 = range(0,d-n,1)

    g = M.variable('g', d, Domain.unbounded())

    M.constraint('gm_RQs', Expr.hstack(g.pick(idx1),g.pick(idx2),g.pick(idx3)), Domain.inRotatedQCone())

    # t = sqrt(n)*g(0)
    M.constraint('gm_t', Expr.sub(Expr.mul(math.sqrt(n),t),g.index(0)), Domain.equalsTo(0.0))

    # Set leaf nodes equal to x.
    M.constraint('gm_g=x', Expr.sub(x,g.slice(d-n,d-n+lenx)), Domain.equalsTo(0.0))

    # Only the leaf nodes has to be psotive
    M.constraint('gm_t>=0', t, Domain.greaterThan(0.0))
    M.constraint('gm_g>=0', g.slice(d-n,d-n+lenx), Domain.greaterThan(0.0))

    if lenx < n:
        # Handle the uneven case
        M.constraint('gm_rem', Expr.sub(g.slice(d-n+lenx,d),Expr.outer([1.0]*(n-lenx),t)), Domain.equalsTo(0.0))
```



```
import numpy
import math

import maxlikeden

# Testing using the exponential distribution
n      = 10
y      = numpy.random.exponential(scale=1.0, size=n)
y      = numpy.sort(y)

xstar = maxlikeden.buildandsolve(y)

viol   = 0.0
a      = 0.0
for i in range(n-1):
    a = a+(y[i+1]-y[i])*(xstar[i]+xstar[i+1])

a = 0.5*a
for i in range(n-2):
    viol = max(viol, (xstar[i+1]-xstar[i])/(y[i+1]-y[i])-(xstar[i+2]-xstar[i+1])/(y[i+2]-y[i+1]))

print(y)
print(xstar)

print('Area: %e Viol: %e min(x): %e\n' % (a,viol,numpy.min(xstar)))
```



Test problem

$$\begin{array}{ll}\text{maximize} & t \\ \text{subject to} & x^{\frac{1}{n}} \geq t, \\ & x \leq 100.\end{array}$$



```
import numpy
import math
import random
import sys

import mosek

from mosek.fusion import *

import gmmeancone

v = 100.0
for n in range(2,11):
    with Model("Testing") as M:
        t = M.variable('t', 1, Domain.unbounded())
        x = M.variable('x', n, Domain.unbounded())

        #  $(x[0] \dots x[n-1]) \geq t^n$ ,  $x, t \geq 0$ 
        gmmeancone.appendcone(M,t,x)

        #  $x[0] \leq v$ 
        M.constraint('xltv',x.index(0),Domain.lessThan(v))

        #  $x[i]=1.0$ , for  $i=1, \dots, n-1$ 
        c = M.constraint('fixtoone',x.slice(1,n),Domain.equalsTo(1.0))

    print(c.toString())

    M.objective(ObjectiveSense.Maximize, t)

    M.writeTask('dump.opf')

    M.solve()

    print('Check %e %e' % (v**(1.0/n),t.level()))
```



```
Constraint( 'fixtoone', (1),  
    fixtoone[0] : + 1.0 x[1] = 1.0 )  
Check 1.000000e+01 1.000000e+01
```



```
[objective maximize]
  't[0]'
[/objective]

[constraints]
[con 'gm_RQs[0,0]'] 'g[1]' - 'gm_RQs[0,0].coneslack' = 0e+000 [/con]
[con 'gm_RQs[0,1]'] 'g[2]' - 'gm_RQs[0,1].coneslack' = 0e+000 [/con]
[con 'gm_RQs[0,2]'] 'g[0]' - 'gm_RQs[0,2].coneslack' = 0e+000 [/con]
[con 'gm_t[0]'] 1.414213562373095e+000 't[0]' - 'g[0]' = 0e+000 [/con]
[con 'gm_g=x[0]'] 'x[0]' - 'g[1]' = 0e+000 [/con]
[con 'gm_g=x[1]'] 'x[1]' - 'g[2]' = 0e+000 [/con]
[con 'gm_t>=0[0]'] 0e+000 <= 't[0]' [/con]
[con 'gm_g>=0[0]'] 0e+000 <= 'g[1]' [/con]
[con 'gm_g>=0[1]'] 0e+000 <= 'g[2]' [/con]
[con 'xltv[0]'] 'x[0]' <= 1e+002 [/con]
[con 'fixtoone[0]'] 'x[1]' = 1e+000 [/con]
[/constraints]

[bounds]
[b] 't[0]', 'x[0]', 'x[1]', 'g[0]', 'g[1]', 'g[2]' free [/b]
[b] 'gm_RQs[0,0].coneslack', 'gm_RQs[0,1].coneslack', 'gm_RQs[0,2].coneslack' free [/b]
[cone rquad 'gm_RQs[0]'] 'gm_RQs[0,0].coneslack', 'gm_RQs[0,1].coneslack', 'gm_RQs[0,2].coneslack' [/cone]
[/bounds]
```

Section 2

Other properties in the cones





The interior of the linear cone is given by

$$\text{int}(\mathcal{K}_l) := \{x \in \mathbb{R}^1 : x > 0\}.$$

The interior of the quadratic cone is given by

$$\text{int}(\mathcal{K}_q) := \{x \in \mathbb{R}^n : x_1 > \|x_{2:n}\|\}.$$

The interior of the semidefinite cone is given by

$$\text{int}(\mathcal{K}_s) := \{X \in \mathbb{R}^{n \times n} : X \succ 0\}.$$

Section 3

Duality revisited





Consider

$$\begin{array}{ll}\text{minimize} & x^{-1} \\ \text{subject to} & x \geq 0.\end{array}$$

CQ representation

$$\begin{array}{ll}\text{minimize} & s \\ \text{subject to} & (x; s; \sqrt{2}) \in \mathcal{K}_r.\end{array}$$

Let

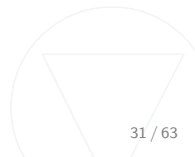
$$(x; s; \sqrt{2}) = \left(\frac{1}{\epsilon^k}; \epsilon^k; \sqrt{2}\right)$$

where $0 < \epsilon < 1$ and k is a positive integer, then it defines a sequence converging towards the optimal solution for $k \rightarrow \infty$.

- The optimal solution is not finite.
- Optimal value of 0 is never attained.
 - Replace minimize by inf).



- Bad things can happen in conic optimization! E.g. nonattainment, infinite values.
- But it did not happen in the (finite dimensional) linear case.
- What is going on?





The primal problem

$$\begin{aligned} \nu_p = \inf \quad & c^T x \\ \text{subject to} \quad & Ax = b, \\ & x \in \mathcal{K}. \end{aligned} \tag{1}$$

and the dual problem

$$\begin{aligned} \nu_d = \sup \quad & b^T y \\ \text{subject to} \quad & A^T y + s = c, \\ & s \in \mathcal{K}^*, \end{aligned} \tag{2}$$

where

$$\mathcal{K}^* := \{s : s^T x \geq 0, \forall x \in \mathcal{K}\}.$$



Lemma

- 1 If \mathcal{K} is convex and closed, then $(\mathcal{K}^*)^* = \mathcal{K}$.
- 2 \mathcal{K}^* is closed and convex. (Holds even if \mathcal{K} is not convex but is a cone).
- 3 $\mathcal{K}_1 \subseteq \mathcal{K}_2$ implies $\mathcal{K}_2^* \subseteq \mathcal{K}_1^*$.



- Linear case:

$$(\mathcal{K}_l)^* = \mathcal{K}_l.$$

- Conic quadratic case:

$$(\mathcal{K}_q)^* = \mathcal{K}_q.$$

- Semidefinite case:

$$(\mathcal{K}_s)^* = \mathcal{K}_s.$$

All 3 cones are **self-dual**!



- The problem is *primal feasible* if a solution x exists satisfying the constraints of (1).
- The problem is *dual feasible* if a solution (y, s) exists satisfying the constraints of (2).
- If (1) is infeasible, then $\nu_p = \infty$.
- If (2) is infeasible, then $\nu_d = -\infty$.





Lemma

(1) is infeasible if

$$\exists y : \quad b^T y > 0 \quad -A^T y \in \mathcal{K}^*. \quad (3)$$

Proof.

Assume (3) holds and x^* is a feasible solution then

$$\begin{aligned} 0 &< b^T y \\ &= (Ax^*)^T y \\ &= -(-A^T y)^T x^* \\ &\leq 0 \end{aligned}$$

which is a contradiction.





Lemma

(2) is infeasible if

$$\exists x : \quad c^T x < 0, \quad Ax = 0, \quad x \in \mathcal{K}. \quad (4)$$

Assume x^* is a feasible solution and x satisfies (4)

$$\begin{aligned} A(x^* + \alpha x) &= b, \\ x^* + \alpha x &\in \mathcal{K}, \forall \alpha \geq 0. \end{aligned}$$

And

$$\lim_{\alpha \rightarrow \infty} c^T(x^* + \alpha x) = -\infty.$$

A conclusion anyone?



Lemma

Given a primal-dual feasible solution (x, y, s) then

$$\begin{aligned} \text{duality gap} &:= \nu_p - \nu_d \\ &\geq 0. \end{aligned}$$

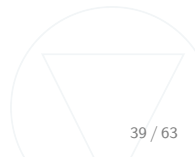
Follows from $x \in \mathcal{K}$ and $s \in \mathcal{K}^$ implies $x^T s \geq 0$.*

A break

Why are the previous 3 lemmas important



Suggestions/Comments?





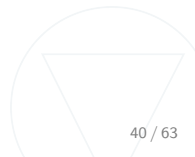
(1) is said to be strongly **feasible** if there $\exists \varepsilon > 0$ such that

$$\{x \in \mathbb{R}^n : Ax = \hat{b}, x \in \mathcal{K}\} \neq \emptyset$$

for all \hat{b} satisfying

$$\|\hat{b} - b\| \leq \varepsilon.$$

This is the same as saying that a small perturbation in b does NOT make the problem infeasible.





(2) is said to be strongly **infeasible** if there $\exists \varepsilon > 0$ such that

$$\{x \in \mathbb{R}^n : Ax = \hat{b}, x \in \mathcal{K}\} = \emptyset$$

for all \hat{b} satisfying

$$\|\hat{b} - b\| \leq \varepsilon.$$

This is the same as saying that a small perturbation in b does NOT make the problem feasible.



Lemma

(1) is strongly infeasible if and only if

$$b^T y = 1, \quad A^T y + s = 0, \quad s \in \mathcal{K}^*$$

is strongly feasible.

Lemma

(2) is strongly infeasible if and only if

$$c^T x = -1, \quad Ax = 0, \quad x \in \mathcal{K}$$

is strongly feasible.



Theorem

(Strong duality) If either (1) or (2) is strong feasible, then $\nu_d = \nu_p$.



Observe:

- For proofs see [2, p. 73] and [1].
- If A is of full row rank and

$$\text{int}(\{x \in \mathbb{R}^n : Ax = b, x \in \mathcal{K}\}) \neq \emptyset$$

then (1) is strongly feasible.

- When does it go wrong?
 - If a small perturbation in the problem data makes the problem status flip from feasible to infeasible or from infeasible to feasible.
- Such problems must be intrinsically hard to solve.
 - Consider that computations are done in finite precision.

Nasty example 1



(1) has an optimal solution but the dual is infeasible

For instance the problem

$$\begin{array}{ll}\text{minimize} & -x_2 \\ \text{subject to} & x_1 - x_3 = 0, \\ & \sqrt{x_2^2 + x_3^2} \leq x_1,\end{array}$$

has the set feasible solutions:

$$\{(x_1, x_2, x_3) : x_1 \geq 0, x_2 = 0, x_3 \geq 0\}.$$

Hence, $x = (0, 0, 0)$ is an optimal solution.

The corresponding dual problem is

$$\begin{array}{llll} \text{maximize} & 0 & & \\ \text{subject to} & y + s_1 & = & 0, \\ & s_2 & = & -1, \\ & -y + s_3 & = & 0, \\ & \sqrt{s_2^2 + s_3^2} & \leq & s_3. \end{array}$$

Hence,

$$\sqrt{s_1^2 + 1} \leq s_1$$

which implies the dual problem is infeasible.



Nasty example 2



Given a primal-dual feasible solution there might be a duality gap

Consider

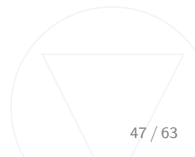
$$\begin{array}{ll} \min & x_2 \\ \text{subject to} & \sqrt{x_1^2 + (x_2 - 1)^2} \leq x_1, \\ & \sqrt{(-x_1 + x_2)^2} \leq x_1. \end{array}$$

From the first constraint it follows

$$x_2 = 1$$

Using this fact and the second constraint then

$$1 \leq 2x_1.$$



The set of primal feasible solutions is

$$\left\{ (x_1, x_2) : x_1 \geq \frac{1}{2}, x_2 = 1 \right\}$$

and the optimal objective value is 1.

The corresponding dual problem is

$$\begin{array}{ll} \max & z_2 \\ \text{subject to} & z_1 + w_1 - z_3 + w_2 = 0, \\ & z_2 + z_3 = 1, \\ & \sqrt{z_1^2 + z_2^2} \leq w_1, \\ & \sqrt{z_3^2} \leq w_2. \end{array}$$



The two last constraints implies

$$w_1 \geq |z_1| \text{ and } w_2 \geq |z_3|$$

we have

$$w_1 + z_1 \geq 0 \text{ and } w_2 - z_3 \geq 0.$$

Using the first constraint this implies

$$w_1 = -z_1 \text{ and } w_2 = z_3.$$

Now using the second constraint we have that

$$z_2 = 1 - z_3 = 1 - w_2.$$



Therefore, the dual problem is equivalent to

$$\begin{array}{ll}\text{maximize} & 1 - w_2 \\ \text{subject to} & \sqrt{w_1^2 + (1 - w_2)^2} \leq w_1, \\ & \sqrt{w_2^2} \leq w_2\end{array}$$

which has the feasible set $\{(w_1, w_2) : w_1 \geq 0, w_2 = 1\}$ and the optimal objective value is zero. Hence,

$$\begin{aligned}\text{duality gap} &= 1 - 0 \\ &= 1.\end{aligned}$$



It can be verified that if

$$(x_2 - 1)^2$$

with

$$(x_2 - \alpha)^2$$

where $\alpha > 0$ then the dual gap will be α .





- Almost identical.
 - Dual problems look a like.
 - Weak duality is identical
 - Infeasibility certificates exists.
- Linear optimization:
 - No duality gap occur.
 - The optimal value is always attained.
- Conic optimization:
 - Any bad situation imaginable can occur.



In linear optimization complementarity means something like

$$x_i s_i = 0.$$

What does the complementarity conditions look like for conic quadratic optimization?

First define the arrow head matrix

$$V := \text{mat}(v) = \begin{bmatrix} v_1 & v_2 & \cdots & v_n \\ v_2 & v_1 & & \\ \vdots & & \ddots & \\ v_n & & & v_1 \end{bmatrix}.$$

Observe

$$\begin{aligned}\text{mat}(x)s &= \begin{bmatrix} x_1 & x_{2:n} & \cdots & x_n \\ x_2 & x_1 & & \\ \vdots & & \ddots & \\ x_n & & & x_1 \end{bmatrix} s \\ &= \begin{bmatrix} x^T s \\ x_1 s_2 + s_1 x_2 \\ \vdots \\ x_1 s_n + s_1 x_n \end{bmatrix}\end{aligned}$$



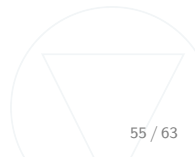


Lemma

Assume $\mathcal{K} = \mathcal{K}^1 \times \dots \times \mathcal{K}^r$ and each \mathcal{K}^k is a quadratic cone. If $x, s \in \mathcal{K}$, then x and s are complementary, i.e. $x^T s = 0$, if and only if

$$X^k S^k e^k = S^k X^k e^k = 0, \quad k = 1, \dots, r,$$

where $X^k := \text{mat}(x^k)$, $S^k := \text{mat}(s^k)$ and $e^k = (0, 0, \dots, 1, \dots, 0)^T \in \mathbb{R}^{n^k}$.



Proof:
Clearly

$$X^k S^k e^k = 0 \Rightarrow (x^k)^T s^k = 0$$

because

$$\begin{aligned} 0 &= \sum_{i=1}^n (e^k)^T X^k S^k e^k \\ &= \sum_{k=1}^n (x^k)^T s^k \\ &= x^T s. \end{aligned}$$




Next we prove if

$$(x^k)^T s^k = 0 \Rightarrow X^k S^k e^k = 0$$

This is clearly true if $x_1^k = 0$ or $s_1^k = 0$. Therefore, we can assume that $x_1^k > 0$ and $s_1^k > 0$.

Now

$$\begin{aligned} 0 &= x^T s \\ &= \sum_{k=1}^r (x^k)^T (s^k) \\ &= \sum_{k=1}^r \left(x_1^k s_1^k + (x_{2:n^k}^k)^T s_{2:n^k}^k \right) \\ &\geq \sum_{k=1}^r \left(x_1^k s_1^k - \left\| (x_{2:n^k}^k) \right\| \left\| s_{2:n^k}^k \right\| \right) \\ &\geq 0. \end{aligned}$$


We can conclude

$$\begin{aligned}x_1^k s_1^k &= \left\| x_{2:n^k}^k \right\| \left\| s_{2:n^k}^k \right\|, \\x_1^k &= \left\| x_{2:n^k}^k \right\|, \\s_1^k &= \left\| s_{2:n^k}^k \right\|.\end{aligned}$$

(Why?).



Now

$$|(x_{2:n^k}^k)^T s_{2:n^k}^k| = \|x_{2:n^k}^k\| \|s_{2:n^k}^k\|$$

can only be the case if

$$\exists \alpha : x_{2:n^k}^k = \alpha s_{2:n^k}^k.$$

Therefore,

$$\begin{aligned} 0 &= (x^k)^T s^k \\ &= x_1^k s_1^k + \alpha \|s_{2:n^k}^k\|^2 \\ &= x_1^k s_1^k + \alpha (s_1^k)^2 \end{aligned}$$

and

$$\alpha = -\frac{x_1^k}{s_1^k}$$

implying that the complementarity conditions $X^k s^k = 0$ are satisfied.



Lemma

Let $X, S \in K_s$ then they are complementarity if

$$XS = 0.$$

Proof.

See



Section 4

Summary





- Have introduced semidefinite optimization.
- Reviewed conic duality.
 - Shown that robust feasibility is important.
- Discussed complementarity conditions.



- [1] A. Ben-Tal and A. Nemirovski.
Lectures on Modern Convex Optimization: Analysis, Algorithms, and Engineering Applications.
MPS/SIAM Series on Optimization. SIAM, 2001.
- [2] J. Renegar.
A mathematical view of interior-point methods in convex optimization.
MPS/SIAM Series on Optimization. SIAM, 2001.