3. SDP Relaxations for Quadratic Programming

- Schur complement
- Norm constraints
- Convex quadratic constraints
- Eigenvalue problems
- Quadratic programming
- MAXCUT
- Boolean optimization
- Primal and dual SDP relaxations
- Randomization
- Interpretations
- Examples
- LQR with binary inputs
completion of squares

If $A \in \mathbb{R}^{n \times n}$ and $D \in \mathbb{R}^{m \times m}$ are symmetric matrices and $B \in \mathbb{R}^{n \times m}$, then

$$
\begin{bmatrix}
    x \\
    y
\end{bmatrix}^T
\begin{bmatrix}
    A & B \\
    B^T & D
\end{bmatrix}
\begin{bmatrix}
    x \\
    y
\end{bmatrix} =
(x + A^{-1}By)^T A(x + A^{-1}By) + y^T(D - B^T A^{-1}B)y
$$

- this gives a test for global positivity:

$$
\begin{bmatrix}
    x \\
    y
\end{bmatrix}^T
\begin{bmatrix}
    A & B \\
    B^T & D
\end{bmatrix}
\begin{bmatrix}
    x \\
    y
\end{bmatrix} > 0 \text{ for all } \begin{bmatrix}
    x \\
    y
\end{bmatrix} \neq 0 \iff A \succ 0 \text{ and } D - B^T A^{-1}B \succ 0
$$

- gives a general formula for quadratic optimization; if $A \succ 0$, then

$$
\min_x \begin{bmatrix}
    x \\
    y
\end{bmatrix}^T
\begin{bmatrix}
    A & B \\
    B^T & D
\end{bmatrix}
\begin{bmatrix}
    x \\
    y
\end{bmatrix} = y^T(D - B^T A^{-1}B)y
$$

and the minimizing $x$ is $x_{opt} = -A^{-1}By$
the Schur complement

this also gives the matrix decomposition

\[
\begin{bmatrix}
x \\
y
\end{bmatrix}^T
\begin{bmatrix}
A & B \\
B^T & D
\end{bmatrix}
\begin{bmatrix}
x \\
y
\end{bmatrix} = (x + A^{-1}By)^T A(x + A^{-1}By)^T + y^T(D - B^T A^{-1}B)y
\]

\[
= \begin{bmatrix}
x \\
y
\end{bmatrix}^T
\begin{bmatrix}
I & 0 \\
B^T A^{-1} & I
\end{bmatrix}
\begin{bmatrix}
A & 0 \\
0 & D - B^T A^{-1}B
\end{bmatrix}
\begin{bmatrix}
I & A^{-1}B \\
0 & I
\end{bmatrix}
\begin{bmatrix}
x \\
y
\end{bmatrix}
\]

since this holds for all \( x, y \),

\[
\begin{bmatrix}
A & B \\
B^T & D
\end{bmatrix} = \begin{bmatrix}
I & 0 \\
B^T A^{-1} & I
\end{bmatrix}
\begin{bmatrix}
A & 0 \\
0 & D - B^T A^{-1}B
\end{bmatrix}
\begin{bmatrix}
I & A^{-1}B \\
0 & I
\end{bmatrix}
\]

holds whenever \( A \) is invertible
matrix fractional constraints

constraints involving *fractions* with positive denominator are SDP constraints; if $A \succ 0$ then

$$D - B^T A^{-1} B \succ 0 \quad \iff \quad \begin{bmatrix} A & B \\ B^T & D \end{bmatrix} \succ 0$$

in particular, if $d^T x + f > 0$ then

$$\frac{(c^T x + g)^2}{d^T x + f} < t \quad \iff \quad \begin{bmatrix} t & c^T x + g \\ c^T x + g & d^T x + f \end{bmatrix} \succ 0$$
norm constraints

for any matrix $A \in \mathbb{R}^{n \times m}$

\[
\|A\| < t \iff \begin{bmatrix} tI & A \\ A^T & tI \end{bmatrix} \succeq 0
\]

here $\|A\|$ is the maximum singular value (or spectral norm) $\sigma_1(A)$

in particular, when $x \in \mathbb{R}^n$, this gives

\[
\|x\| < t \iff \begin{bmatrix} tI & x \\ x^T & t \end{bmatrix} \succeq 0
\]
**convex quadratic constraints**

suppose $P$ is symmetric, and $P \succeq 0$; we can represent the convex quadratic constraint

$$x^T P x + q^T x + r < 0$$

as a semidefinite programming constraint as follows

write $P$ as the product $P = A^T A$ via Cholesky or eigenvalue decomposition, then

\[
\begin{align*}
    x^T P x + q^T x + r < 0 & \iff \\
    \begin{bmatrix}
    -I & Ax \\
    x^T A^T & q^T x + r
    \end{bmatrix} < 0
\end{align*}
\]
eigenvalue problems

constraints on the maximum eigenvalue of a symmetric matrix $A$

$$\lambda_{\text{max}}(A) < t \iff tI - A \succeq 0$$

or the sum of the $r$ largest eigenvalues

$$\sum_{i=1}^{r} \lambda_i(A) < t \iff \text{there exists } X \in \mathbb{S}^n, y \in \mathbb{R} \text{ such that}$$

$$ry + \text{trace } X < t$$

$$yI + X - A \succeq 0$$

$$X \succeq 0$$

many other convex constraints are representable as SDP constraints
Quadratic programming

A quadratically constrained quadratic program (QCQP) has the form

\[
\begin{align*}
\text{minimize} & \quad f_0(x) \\
\text{subject to} & \quad f_i(x) \leq 0 \quad \text{for all } i = 1, \ldots, m
\end{align*}
\]

where the functions \( f_i : \mathbb{R}^n \to \mathbb{R} \) have the form

\[
f_i(x) = x^T P_i x + q_i^T x + r_i
\]

If \( P_i \succeq 0 \) then \( f_i \) is a convex function

- if all the \( f_i \) are convex then the QCQP may be solved by semidefinite programming
- but specialized software for second-order cone programming is more efficient
MAXCUT

given an undirected graph, with no self-loops

- vertex set $V = \{1, \ldots, n\}$
- edge set $E \subset \\{i,j\} \mid i,j \in V, i \neq j\}$

For a subset $S \subset V$, the capacity of $S$ is the number of edges connecting a node in $S$ to a node not in $S$

the MAXCUT problem

find $S \subset V$ with maximum capacity

the example above shows a cut with capacity 15; this is the maximum
example

a graph with 12 nodes, 24 edges; the maximum capacity $c_{\text{max}} = 20$
problem formulation

the graph is defined by its adjacency matrix

\[ Q_{ij} = \begin{cases} 
1 & \text{if } \{i, j\} \in E \\
0 & \text{otherwise} 
\end{cases} \]

and specify a cut \( S \) by a vector \( x \in \mathbb{R}^n \)

\[ x_i = \begin{cases} 
1 & \text{if } i \in S \\
-1 & \text{otherwise} 
\end{cases} \]

then \( 1 - x_i x_j = 2 \) if \( \{i, j\} \) is a cut, so the capacity of \( x \) is

\[ c(x) = \frac{1}{4} \sum_{i=1}^{n} \sum_{j=1}^{n} (1 - x_i x_j) Q_{ij} \]

the extra factor of \( \frac{1}{2} \) arises because \( A \) is symmetric
optimization formulation

so we’d like to solve

\[
\begin{align*}
\text{minimize} & \quad x^T Q x \\
\text{subject to} & \quad x_i \in \{-1, 1\} \quad \text{for all } i = 1, \ldots, n
\end{align*}
\]

call the optimal value \( p^* \), then the maximum cut is

\[
c_{\text{max}} = \frac{1}{4} \sum_{i=1}^{n} \sum_{j=1}^{n} Q_{ij} - \frac{1}{4} p^*
\]
**Boolean Optimization**

A classic combinatorial problem:

\[
\begin{align*}
\text{minimize} & \quad x^T Q x \\
\text{subject to} & \quad x_i \in \{-1, 1\}
\end{align*}
\]

- Many other examples; knapsack, LQR with binary inputs, etc.
- Can model the constraints with quadratic equations:

  \[
  x_i^2 - 1 = 0 \quad \iff \quad x_i \in \{-1, 1\}
  \]

- An exponential number of points. Cannot check them all!
- The problem is \emph{NP-complete} (even if \( Q \succeq 0 \)).

Despite the hardness of the problem, there are some very good approaches…
SDP Relaxations

we can find a lower bound on the minimum of this QP, (and hence an upper bound on MAXCUT) using the dual problem; the primal is

\[
\begin{align*}
\text{minimize} & \quad x^T Q x \\
\text{subject to} & \quad x_i^2 - 1 = 0
\end{align*}
\]

the Lagrangian is

\[
L(x, \lambda) = x^T Q x - \sum_{i=1}^{n} \lambda_i (x_i^2 - 1) = x^T (Q - \Lambda) x + \text{trace} \Lambda
\]

where \( \Lambda = \text{diag}(\lambda_1, \ldots, \lambda_n) \); the Lagrangian is bounded below w.r.t. \( x \) if \( Q - \Lambda \succeq 0 \)

The dual is therefore the SDP

\[
\begin{align*}
\text{maximize} & \quad \text{trace} \Lambda \\
\text{subject to} & \quad Q - \Lambda \succeq 0
\end{align*}
\]
SDP Relaxations

From this SDP we obtain a \textit{primal-dual pair of SDP relaxations}

\[
\begin{align*}
\text{minimize} & \quad x^T Q x \\
\text{subject to} & \quad x_i^2 = 1
\end{align*}
\]

\[
\begin{align*}
\text{minimize} & \quad \text{trace } Q X \\
\text{subject to} & \quad X \succeq 0 \\
& \quad X_{ii} = 1
\end{align*}
\]

\[
\begin{align*}
\text{maximize} & \quad \text{trace } \Lambda \\
\text{subject to} & \quad Q \succeq \Lambda \\
& \quad \Lambda \text{ diagonal}
\end{align*}
\]

- We derived them from Lagrangian and SDP duality
- But, these SDP relaxations arise in \textit{many} other ways
- Well-known in combinatorial optimization, graph theory, etc.
- Several interpretations
SDP Relaxations: Dual Side

Gives a simple underestimator of the objective function.

\[
\begin{align*}
\text{maximize} & \quad \text{trace } \Lambda \\
\text{subject to} & \quad Q \succeq \Lambda \\
& \quad \Lambda \text{ diagonal}
\end{align*}
\]

Directly provides a lower bound on the objective: for any feasible \( x \):

\[
x^T Q x \geq x^T \Lambda x = \sum_{i=1}^{n} \Lambda_{ii} x_i^2 = \text{trace } \Lambda
\]

- The first inequality follows from \( Q \succeq \Lambda \)
- The second equation from \( \Lambda \) being diagonal
- The third, from \( x_i \in \{+1, -1\} \)
**SDP Relaxations: Primal Side**

The original problem is:

\[
\begin{align*}
\text{minimize} & \quad x^T Q x \\
\text{subject to} & \quad x_i^2 = 1
\end{align*}
\]

Let \( X := xx^T \). Then

\[
x^T Q x = \text{trace} \ Q xx^T = \text{trace} \ Q X
\]

Therefore, \( X \succeq 0 \), has \textit{rank one}, and \( X_{ii} = x_i^2 = 1 \).

Conversely, any matrix \( X \) with

\[
X \succeq 0, \quad X_{ii} = 1, \quad \text{rank} \ X = 1
\]

\textit{necessarily} has the form \( X = xx^T \) for some \( \pm 1 \) vector \( x \).
Primal Side

Therefore, the original problem can be exactly rewritten as:

\[
\begin{align*}
& \text{minimize} \quad \text{trace } QX \\
& \text{subject to} \quad X \succeq 0 \\
& \quad X_{ii} = 1 \\
& \quad \text{rank}(X) = 1
\end{align*}
\]

Interpretation: *lift* to a higher dimensional space, from \( \mathbb{R}^n \) to \( \mathbb{S}^n \).
Dropping the (nonconvex) rank constraint, we obtain the relaxation.

*If* the solution \( X \) has rank 1, then we have solved the original problem.

Otherwise, *rounding schemes* to project solutions. In some cases, approximation guarantees (e.g. Goemans-Williamson for MAX CUT).
feasible points and certificates

\[
\begin{array}{ll}
\text{minimize} & \text{trace } QX \\
\text{subject to} & X \succeq 0 \\
& X_{ii} = 1
\end{array}
\quad
\begin{array}{ll}
\text{maximize} & \text{trace } \Lambda \\
\text{subject to} & Q \succeq \Lambda \\
& \Lambda \text{ diagonal}
\end{array}
\]

- Dual relaxations give \textit{certified} bounds.
- Primal relaxations give information about possible \textit{feasible} points.
- Both are solved \textit{simultaneously} by primal-dual SDP solvers.
Example

\[
\begin{align*}
\text{minimize} & \quad 2x_1x_2 + 4x_1x_3 + 6x_2x_3 \\
\text{subject to} & \quad x_i^2 = 1
\end{align*}
\]

The associated matrix is
\[
Q = \begin{bmatrix}
0 & 1 & 2 \\
1 & 0 & 3 \\
2 & 3 & 0
\end{bmatrix}
\]

The SDP solutions are:
\[
X = \begin{bmatrix}
1 & 1 & -1 \\
1 & 1 & -1 \\
-1 & -1 & 1
\end{bmatrix}, \quad \Lambda = \begin{bmatrix}
-1 & 0 & 0 \\
0 & -2 & 0 \\
0 & 0 & -5
\end{bmatrix}
\]

We have \( X \succeq 0, \ X_{ii} = 1, \ Q - \Lambda \succeq 0, \) and
\[
\text{trace} \ QX = \text{trace} \ \Lambda = -8
\]

Since \( X \) is rank 1, from \( X = xx^T \) we recover the optimal \( x = [1 \ 1 \ -1]^T \),
Spectrahedron

We can visualize this (in $3 \times 3$):

$$X = \begin{bmatrix}
1 & p_1 & p_2 \\
p_1 & 1 & p_3 \\
p_2 & p_3 & 1
\end{bmatrix} \succeq 0$$

in $(p_1, p_2, p_3)$ space.

When optimizing the linear objective function

$$\text{trace } QX = 2p_1 + 4p_2 + 6p_3,$$

the optimal solution is at the vertex $(1, -1, -1)$. 
Randomization

Suppose we solve the primal relaxation

\[
\begin{align*}
\text{minimize} & \quad \text{trace } QX \\
\text{subject to} & \quad X \succeq 0 \\
& \quad X_{ii} = 1 \quad \text{for all } i = 1, \ldots, n
\end{align*}
\]

and the optimal \( X \) is not rank 1

The following randomized algorithm gives a feasible point

Factorize \( X \) as \( X = V^T V \), where \( V = [v_1 \ldots v_n] \in \mathbb{R}^{r \times n} \)

Then \( X_{ij} = v_i^T v_j \), and since \( X_{ii} = 1 \) this factorization gives \( n \) vectors on the unit sphere in \( \mathbb{R}^r \)

Interpretation; instead of assigning either 1 or \(-1\) to each vertex, we have assigned a point on the unit sphere in \( \mathbb{R}^r \) to each vertex
randomized slicing

pick a random vector $q \in \mathbb{R}^r$, and choose cut

$$S = \{ i \mid v_i^T q \geq 0 \}$$

then the probability that $\{i, j\}$ is a cut edge is

$$\frac{\text{angle between } v_i \text{ and } v_j}{\pi} = \frac{1}{\pi} \arccos v_i^T v_j$$

$$= \frac{1}{\pi} \arccos X_{ij}$$

so the expected cut capacity is

$$c_{\text{sdp-expected}} = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{1}{\pi} Q_{ij} \arccos X_{ij}$$
randomization

the upper bound on the cut capacity obtained from the SDP is

\[ c_{\text{sdp-upper-bound}} = \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{1}{4} (1 - X_{ij}) Q_{ij} \]

with \( \alpha = 0.878 \), we have

\[ \alpha (1 - t) \frac{\pi}{2} \leq \arccos(t) \quad \text{for all} \ t \in [-1, 1] \]

so we have

\[ c_{\text{sdp-upper-bound}} \leq \frac{1}{2\alpha \pi} \sum_{i=1}^{n} \sum_{j=1}^{n} Q_{ij} \arccos X_{ij} \]

\[ = \frac{1}{\alpha} c_{\text{sdp-expected}} \]
randomization

so far, we have

\[ c_{\text{sdp-upper-bound}} \leq \frac{1}{\alpha} c_{\text{sdp-expected}} \]

since \( c_{\text{sdp-expected}} \leq c_{\text{max}} \), we have

the SDP upper-bound is no more than 14% too large

this is the smallest approximation ratio of any known polynomial-time MAXCUT algorithm

after solving the SDP, we slice randomly to generate a random family of feasible points; since \( c_{\text{max}} \leq c_{\text{sdp-upper-bound}} \), we have

the expected capacity of these feasible points is at least 87% of the optimal
**coin-flipping approach**

suppose we just randomly assigned vertices to $S$ with probability $\frac{1}{2}$; then

$$c_{\text{coinflip-expected}} = \frac{1}{4} \sum_{i=1}^{n} \sum_{j=1}^{n} Q_{ij}$$

also a trivial upper bound on the maximum cut is just the total number of edges

$$c_{\text{trivial-upper-bound}} = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} Q_{ij}$$

and so $c_{\text{coinflip-expected}} = \frac{1}{2} c_{\text{trivial-upper-bound}}$

again, since $c_{\text{coinflip-expected}} \leq c_{\text{max}}$, we have

the trivial upper-bound is no more than double the optimal

and since $c_{\text{max}} \leq c_{\text{trivial-upper-bound}}$, we have

the expected capacity of the feasible points is at least half of the optimal
example

- 12 nodes, 18 edges
- SDP upper bound 15
A General Scheme

- The relaxed $X$ suggests candidate points.
- The diagonal matrix $\Lambda$ certifies a lower bound.

Ubiquitous scheme in optimization (convex hulls, fractional colorings, etc.)
We will learn systematic ways of constructing these relaxations, and more...
LQR with Binary Inputs

Consider the discrete-time LQR problem

\[
\text{minimize } \|y(t) - y_r(t)\|^2 \quad \text{subject to } \begin{cases} x(t + 1) = Ax(t) + Bu(t) \\ y(t) = Cx(t) \end{cases}
\]

where \(y_r\) is the reference output trajectory, and the input \(u(t)\) is constrained by \(|u(t)| = 1\) for all \(t = 0, \ldots, N\).

An open-loop LQR-type problem, but with a \textit{bang-bang} input.
LQR with Binary Inputs

The objective $\|y(t) - y_r(t)\|^2$ is a \textit{quadratic} function of the input $u$:

$$
\begin{bmatrix}
    y(0) \\ y(1) \\ y(2) \\ \vdots \\ y(t)
\end{bmatrix} =
\begin{bmatrix}
    0 & 0 & 0 & \cdots & 0 \\
    CB & 0 & 0 & \cdots & 0 \\
    CAB & CB & 0 & \cdots & 0 \\
    \vdots & \vdots & \vdots & \ddots & \vdots \\
    CA^tB & CA^{t-1}B & \cdots & CB & 0
\end{bmatrix}
\begin{bmatrix}
    u(0) \\ u(1) \\ u(2) \\ \vdots \\ u(t)
\end{bmatrix}
$$

So the problem can be written as:

$$
\text{minimize } \begin{bmatrix} u \\ 1 \end{bmatrix}^T \begin{bmatrix} Q & r \\ r^T & s \end{bmatrix} \begin{bmatrix} u \\ 1 \end{bmatrix}
$$

subject to $u_i \in \{+1, -1\}$ for all $i$

where $Q, r, s$ are functions of the problem data.
This is a quadratic \textit{boolean optimization} problem.
**LQR with Binary Inputs**

\[
\begin{align*}
\text{minimize} & \quad \begin{bmatrix} u \\ 1 \end{bmatrix}^T \begin{bmatrix} Q & r \\ r^T & s \end{bmatrix} \begin{bmatrix} u \\ 1 \end{bmatrix} \\
\text{subject to} & \quad u_i \in \{+1, -1\} \text{ for all } i \\
\end{align*}
\]

for some matrices \((Q, r, s)\) function of the problem data \((A, B, C, N)\).

An **SDP dual bound**:  

\[
\begin{align*}
\text{maximize} & \quad \text{trace}(\Lambda) + \mu \\
\text{subject to} & \quad \begin{bmatrix} Q - \Lambda & r \\ r^T & s - \mu \end{bmatrix} \succeq 0, \quad \Lambda \text{ diagonal}
\end{align*}
\]

Let \(q^*, q_*\) be the optimal value of both problems. Then, \(q^* \geq q_*\):  

\[
\begin{bmatrix} u \\ 1 \end{bmatrix}^T \begin{bmatrix} Q & r \\ r^T & s \end{bmatrix} \begin{bmatrix} u \\ 1 \end{bmatrix} \geq \begin{bmatrix} u \\ 1 \end{bmatrix}^T \begin{bmatrix} \Lambda & 0 \\ 0 & \mu \end{bmatrix} \begin{bmatrix} u \\ 1 \end{bmatrix} = \text{trace} \Lambda + \mu
\]
LQR with Binary Inputs

\[
\text{maximize } \quad \text{trace}(\Lambda) + \mu \\
\text{subject to } \quad \begin{bmatrix} Q - \Lambda & r \\ r^T & s - \mu \end{bmatrix} \succeq 0, \quad \Lambda \text{ diagonal}
\]

Since \((\Lambda, \mu) = (0, 0)\) is always feasible, \(q_* \geq 0\).

Furthermore, the bound is never worse than the LQR solution obtained by dropping the \pm 1 constraint, since

\[
\Lambda = 0, \quad \mu = s - r^T Q^{-1} r
\]

is a feasible point.

<table>
<thead>
<tr>
<th>N</th>
<th>LQR cost</th>
<th>SDP bound</th>
<th>Bang Bang</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>14.005</td>
<td>15.803</td>
<td>15.803</td>
</tr>
<tr>
<td>15</td>
<td>15.216</td>
<td>16.698</td>
<td>16.705</td>
</tr>
<tr>
<td>20</td>
<td>15.364</td>
<td>16.905</td>
<td>16.927</td>
</tr>
</tbody>
</table>

Example: