Approximation Algorithms

Martin Böhm (University of Bremen)

June 20, 2019

Rounding of semidefinite programs

Positive semidefiniteness

Definition

A symmetric real matrix M is *positive semidefinite* when for all real-valued vectors $x \in \mathbb{R}^n$ $x^T Mx \geq 0$.

Equivalent definitions: M is positive semidefinite iff:

1. Each eigenvalue of M is non-negative.
2. (Vector square root.) There exists a matrix $U \in \mathbb{R}^{n \times n}$ such that $U^T U = M$.

Can we check PSD in poly-time?

Yes, we can:

Cholesky factorization

Input: A symmetric real matrix $M \in \mathbb{R}^{n \times n}$.

Output: A decomposition $M = U^T U$, if such a decomposition exists.

Imagine M is written as

$$M = \begin{bmatrix} \alpha & \vec{q}^T \\ \vec{q} & N \end{bmatrix}$$

Then a recursive decomposition gives us:

$$M = \begin{bmatrix} \sqrt{\alpha} & \vec{0}^T \\ \frac{1}{\sqrt{\alpha}}\vec{q} & I_{n-1} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & N - \frac{1}{\alpha}\vec{q}\vec{q}^T \end{bmatrix} \begin{bmatrix} \sqrt{\alpha} & \frac{1}{\sqrt{\alpha}}\vec{q}^T \\ \vec{0} & I_{n-1} \end{bmatrix}$$

Correctness of algorithm

$$M = \begin{bmatrix} \sqrt{\alpha} & \vec{0}^T \\ \frac{1}{\sqrt{\alpha}}\vec{q} & I_{n-1} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & N - \frac{1}{\alpha}\vec{q}\vec{q}^T \end{bmatrix} \begin{bmatrix} \sqrt{\alpha} & \frac{1}{\sqrt{\alpha}}\vec{q}^T \\ \vec{0} & I_{n-1} \end{bmatrix}$$

Lemma

If a matrix M has the form $M = \begin{bmatrix} \alpha & \boldsymbol{q}^T \\ \boldsymbol{q} & N \end{bmatrix}$ with $\alpha > 0$ and if $M \succeq 0$, then also the matrix $N - \frac{1}{\alpha}\boldsymbol{q}\boldsymbol{q}^T$ is psd.

Lemma

If a matrix M has the form

$$M = \begin{bmatrix} \alpha & \boldsymbol{q}^T \\ \boldsymbol{q} & N \end{bmatrix}$$

with $\alpha = 0$ and if $M \succeq 0$, then $\boldsymbol{q} = \boldsymbol{0}$.

Semidefinite programs

Typical SDP:

$$\begin{aligned}
\max \quad & c_{1,1}x_{1,1} + c_{1,2}x_{1,2} + \ldots + c_{n,n}x_{n,n} \\
\text{s.t.} \quad & u_{1,1}x_{1,1} + u_{1,2}x_{1,2} + \ldots + u_{n,n}x_{n,n} \leq b_1 \\
& v_{1,1}x_{1,1} + v_{1,2}x_{1,2} + \ldots + v_{n,n}x_{n,n} \leq b_2 \\
& w_{1,1}x_{1,1} + w_{1,2}x_{1,2} + \ldots + w_{n,n}x_{n,n} \leq b_3 \\
& X \succeq 0
\end{aligned}$$

Matrix form:

$$\begin{aligned}
& \max C \bullet X \\
\text{s.t.} \quad & \forall i : A_i \bullet X \leq b_i \\
& X \succeq 0
\end{aligned}$$

In the **typical SDP** above, we have $A_1 = U$, $A_2 = V$, $A_3 = W$.

Scalar product of matrices: $A \bullet B = a_{1,1}b_{1,1} + \cdots + a_{n,n}b_{n,n}$.

$A \bullet B$ is just one real number.

Vector form

Matrix form:

$$\max C \bullet X$$
$$\text{s.t. } \forall i: A_i \bullet X \leq b_i$$
$$X \succeq 0$$

As $X = U^T U$, we can rewrite the program to actually be a program that computes on vectors of U:

Vector form:

$$\max \sum_{a,b} c_{a,b}(\boldsymbol{u_a} \cdot \boldsymbol{u_b})$$
$$\text{s.t. } \forall i: \sum_{a,b} A_{i,a,b}(\boldsymbol{u_a} \cdot \boldsymbol{u_b}) \leq b_i$$
$$\forall a: \boldsymbol{u_a} \in \mathbb{R}^n$$

SDPs and solvability

Q: Are SDPs solvable in PTIME?

A: No, but *almost*:

1. We can only find an ε-approximate solution – an optimal solution such that all matrices X' in "radius ε" are also feasible solutions.
2. We can only solve SDPs where we can guarantee that the optimal solution will not be exponential in bit length – more on this later.

Approximating the maximum cut

Definition (Max Cut)

Input: Undirected graph $G = (V, E)$, a non-negative weight function on edges.

Output: Any subset of vertices $S \subseteq V$ – a *cut*.

Goal: Maximize sum of edges between S and $V \setminus S$.

Integer $\{0, 1\}$-program:

$$\max \sum_{ab \in E} w_{ab} z_{ab}$$

$$\forall ab \in E : z_{ab} \leq x_a + x_b$$

$$\forall ab \in E : z_{ab} \leq (1 - x_a) + (1 - x_b)$$

$$\forall v \in V : x_v \in \{0, 1\}$$

$$\forall ab \in E : z_{ab} \geq 0, z_{ab} \leq 1$$

Fact: There exists a 0.5-approximation algorithm for Max Cut.

From integer to SDP

Q1: What if we wanted a $\{1, -1\}$-integer quadratic program?

$$\max \sum_{ab \in \vec{E}(G)} w_{ab} \frac{1 - u_a u_b}{2}$$

$$\forall v \in V(G) \colon u_v \in \{-1, 1\}$$

Note: $u_i \in \{-1, 1\}$ is not a valid integer constraint.

Q2: What if instead of $\{-1, 1\}$, we had *all* unit size vectors?

$$\max \sum_{ab \in \vec{E}(G)} w_{ab} \frac{1 - \langle \boldsymbol{u_a}, \boldsymbol{u_b} \rangle}{2}$$

$$\forall v \in V(G) \colon \langle \boldsymbol{u_v}, \boldsymbol{u_v} \rangle = ||\boldsymbol{u}_v||_2^2 = 1$$

From a vector to a SDP form

$$\max \sum_{ab \in \vec{E}(G)} w_{ab} \frac{1 - \langle \boldsymbol{u_a}, \boldsymbol{u_b} \rangle}{2}$$

$$\forall v \in V(G) \colon \langle \boldsymbol{u_v}, \boldsymbol{u_v} \rangle = ||\boldsymbol{u}_v||_2^2 = 1$$

$$\max \sum_{ab \in \vec{E}(G)} w_{ab} \frac{1 - x_{a,b}}{2}$$

$$\forall v \in V(G) \colon x_{v,v} = 1$$

$$X \succeq 0$$

Fact: This SDP has bounded solutions and can be ε-approximately solved in polynomial time.

Theorem (Goemans, Williamson '95)

There exists a randomized $0.878 - \varepsilon$*-approximation for* Max Cut *for any* $\varepsilon > 0$.

Algorithm

1. Create vector SDP, solve in polynomial time. Get vectors U.
2. Uniformly randomly select a plane P intersecting origin 0.
3. Plane cuts unit ball into two sides. Round all vectors on one side to 1, other to -1.

Algorithm

1. Create vector SDP, solve in polynomial time. Get vectors U.
2. Uniformly randomly select a plane P intersecting origin 0.
3. Plane cuts unit ball into two sides. Round all vectors on one side to 1, other to -1.

Lemma

Let u, u' be two unit vectors in $\mathbb{R}^n$. The probability that a random plane P passing through the origin separates u from u' is

$$P[\boldsymbol{u} \textit{ separated from } \boldsymbol{u'}] = \frac{1}{\pi} \arccos \langle \boldsymbol{u}, \boldsymbol{u'} \rangle.$$

Lemma

For all $z \in [-1, 1]$,

$$\frac{\arccos z}{\pi} \geq 0.878 \frac{1 - z}{2}.$$