CSE5401 Lec. 16

Semidefinite Programming

1. Semidefinite programs
2. Algorithms: Cholesky factorization & separation
3. Example

Linear programs are an extremely useful technique for designing algorithms. But, sometimes we want to solve problems that can't be captured by linear functions and constraints. Can we find methods for solving, for example, quadratic objectives and constraints? (I.e., of the form $a(\vec{x}) = \sum_{i,j} a_{ij} x_i x_j + \sum_i a_i x_i + a_0 \geq 0$)

Unfortunately, not in general. Consider the following problem

Max-Cut Given an undirected graph $G=(V,E)$, find $S \subseteq V$ maximizing $|E(S)|$ ($= |\{(u,v) \in E : u \in S, v \notin S\}|$)

Max-Cut is an NP-complete optimization problem, so we don't expect to find a polynomial-time algorithm for it.

Now, consider the following quadratic optimization problem

maximize $\frac{1}{4} \sum_{(u,v) \in E} (s_u - s_v)^2$ s.t. $s_v^2 = 1$ for all $v \in V$.

The constraint $s_v^2 = 1$ is equivalent to $s_v \in \{-1, 1\}$, and the cut $S = \{v \in V : s_v = 1\}$ has value $|E(S)| = \sum_{(u,v) \in E} \underbrace{1[u \in S, v \notin S]}_{\frac{1}{4}(s_u - s_v)^2}$

Conversely, we can assign $s_v = \begin{cases} 1 & \text{if } v \in S \\ 0 & \text{o.w.} \end{cases}$ to get a solution with objective value $|E(S)|$. So this simple quadratic problem precisely captures Max-Cut, and clearly there is no hope for a PTAS since $|E(S)|$ is always integer valued, and then rounding to the nearest integer would give the true opt. for $\varepsilon < \frac{1}{2|E|}$.

A family of (essentially) quadratic problems that we can solve is semidefinite programs:

Defn: $M \in \mathbb{R}^{n \times n}$ is positive semidefinite ("PSD") if for all $\vec{x} \in \mathbb{R}^n$ $\vec{x}^T M \vec{x} \geq 0$. We denote this by $M \succeq 0$.

We let $M \bullet N$ denote $\sum_{i,j} M_{ij} N_{ij}$ $(= \text{trace}(MN^T))$.

A semidefinite optimization problem in standard form is for $A_1, \ldots, A_m \in \mathbb{R}^{n \times n}$, $b \in \mathbb{R}^m$, $C \in \mathbb{R}^{n \times n}$

maximize $C \bullet X$ s.t. $A_i \bullet X = b_i$ for $i = 1, \ldots, m$, $X \succeq 0$.

$X \in \mathbb{R}^{n \times n}$: $X^T = X$

Apart from the $X \succeq 0$ constraint, this is simply a linear program. And, if the matrices $A_1, \ldots, A_m$ and C are diagonal, we obtain standard form linear programs as a special case. The "quadratic" aspect of PSD X is captured by the following alternative characterization of PSD:

Theorem The following are equivalent for symmetric M:

(a) $M \succeq 0$ (b) $\exists U \in \mathbb{R}^{n \times n}$ s.t. $M = UU^T$ "Cholesky decomp."

(c) all eigenvalues of M are nonnegative.

Proof (a) $\Rightarrow$ (c): consider any eigenvector of M, $\vec{v}$. We recall that the associated eigenvalue $\lambda = \frac{\vec{v}^T M \vec{v}}{\|\vec{v}\|_2^2} \geq 0$ since $M \succeq 0$.

(c) $\Rightarrow$ (b): Recall that for eigenvectors $\vec{v}_1, \ldots, \vec{v}_n$ and eigenvalues $\lambda_1, \ldots, \lambda_n$, we can write $M = V \Lambda V^T$ where $V = \left[\frac{1}{\|\vec{v}_1\|}\vec{v}_1 \cdots \frac{1}{\|\vec{v}_n\|}\vec{v}_n\right]$ and $\Lambda = \begin{bmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_n \end{bmatrix}$. Since each $\lambda_i \geq 0$, there exists $\sqrt{\lambda_i}$ s.t. $\sqrt{\lambda_i}^2 = \lambda_i$

Put $U = V \begin{bmatrix} \sqrt{\lambda_1} & & 0 \\ & \ddots & \\ 0 & & \sqrt{\lambda_n} \end{bmatrix}$. Then $UU^T = V \begin{bmatrix} \sqrt{\lambda_1} & & 0 \\ & \ddots & \\ 0 & & \sqrt{\lambda_n} \end{bmatrix} \begin{bmatrix} \sqrt{\lambda_1} & & 0 \\ & \ddots & \\ 0 & & \sqrt{\lambda_n} \end{bmatrix} V^T = M$.

(b) $\Rightarrow$ (a): For any $\vec{x} \in \mathbb{R}$, $\vec{x}^T M \vec{x} = (\vec{x}^T U)(U^T \vec{x}) = \|U^T \vec{x}\|_2^2 \geq 0$ ▨

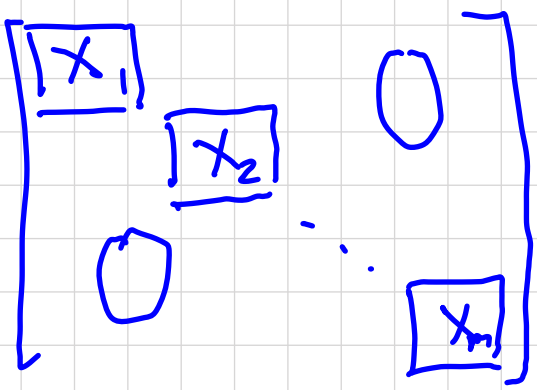
The factorization of the PSD matrix variable X into UU^T means that we can interpret $X_{ij} = \langle \vec{u}^{(i)}, \vec{u}^{(j)} \rangle$ for some vector variables $\vec{u}^{(1)}, \ldots, \vec{u}^{(n)} \in \mathbb{R}^n$. In particular, we then have $\langle \vec{u}^{(i)}, \vec{u}^{(i)} \rangle = \|\vec{u}^{(i)}\|_2^2 \geq 0$ and other standard vector inequalities, like Cauchy-Schwarz: if $\|\vec{u}^{(i)}\|_2^2 \leq b_1^2$ and $\|\vec{u}^{(j)}\|_2^2 \leq b_2^2$, $\langle \vec{u}^{(i)}, \vec{u}^{(j)} \rangle \leq b_1 \cdot b_2$. The constraint $A_i \bullet X = b_i$ is then interpreted as $\sum_{jk} A_{ijk} \langle \vec{u}^{(j)}, \vec{u}^{(k)} \rangle = b_i$.

With this interpretation, we can relax the Max-Cut quadratic program to maximize $\frac{1}{4} \sum_{(u,v) \in E} \|\vec{u} - \vec{v}\|_2^2 = \frac{1}{2} \sum_{(u,v) \in E} (1 - \langle \vec{u}, \vec{v} \rangle)$ s.t. $\|\vec{v}\|_2^2 = 1 \ \forall v \in V$.

(Setting $v_2, \ldots, v_n = 0$ recovers the original program in v_1's, so it is clearly a relaxation.) We'll return to the question of what to do with the vector solution in a future lecture; today we only focus on how to solve the relaxed problem.

As with linear programs, seemingly much more general constraints of the form $A(X) \succeq B$ ($\stackrel{\text{def}}{=} [\langle A_{ij}, X \rangle] - B \succeq 0$) can be reduced to standard-form programs. How?

We first note that we can combine many PSD constraints into one using a block-diagonal matrix: for $X_1, \ldots, X_m$, we use a program of dim. mn and use a constraint to set each var. outside the blocks to 0. This matrix is PSD iff



each X_i is: if some X_i is not, i.e., has $\vec{y}^T X_i \vec{y} < 0$ for $\vec{y} \in \mathbb{R}^n$, then $(-0-\vec{y}-0-) \begin{bmatrix} X_1 & & 0 \\ & \ddots & \\ 0 & & X_m \end{bmatrix} \begin{pmatrix} 0 \\ \vdots \\ \vec{y} \\ \vdots \\ 0 \end{pmatrix} < 0$. And conversely, if $\forall i \ X_i \succeq 0$, then $\vec{y} = (\vec{y}_1 \ \vec{y}_2 \cdots \vec{y}_m)$ gives $\sum_{i=1}^m \vec{y}_i^T X_i \vec{y}_i \geq 0$. Then we can use an equality constraint to fix $X_{ijk} - \sum a_{ijk} \cdot X_{0jk} = -B_{ijk}$ (letting X_0 serve as our "real" variables) so that $A_i(X) - B_i = X_i \succeq 0$.

Algorithms

Semidefinite programs are convex optimization problems, and we solve them using a separation oracle for the PSD constraint. Intuitively, if there is a negative eigenvalue, then the associated eigenvector $\vec{v}$ gives a violated constraint: $\vec{v}^T X \vec{v} = X \bullet \vec{v}^T \vec{v} < 0$ where a PSD X would have $\vec{v}^T X \vec{v} \geq 0$. Unfortunately, using the eigenvectors is numerically challenging. A method that actually works can be obtained while (attempting) computing a Cholesky factorization, which we need for many applications anyway. Here is the algorithm. Given M, it either returns U s.t. $M = UU^T$ or $\vec{v}$ s.t. $\vec{v}^T M \vec{v} < 0$:

Cholesky-or-Violation $\left(M = \begin{bmatrix} \alpha & \vec{q}^T \\ \vec{q} & N \end{bmatrix}\right)$ (dim > 1)

- If $\alpha = 0$
  - If $\vec{q} = 0$ call Cholesky-or-Violation (N)
    - If it returns V, return $\begin{pmatrix} 0 & \cdots \\ \vdots & V \end{pmatrix}$
    - Else it returns $\vec{v}$, return $(0, \vec{v})$

    } extend with 0s to dim. n
  - Else (some $q_i \neq 0$)
    - If $q_i > 0$ return $\left(\frac{\max\{N_{ii}, 1\}}{q_i}, 0 \cdots 0, -\max\{N_{ii}, 1\}, 0 \cdots 0\right)$ (pos. $i+1$)
    - Else return $\left(-\frac{\max\{N_{ii}, 1\}}{q_i}, 0 \cdots 0, \max\{N_{ii}, 1\}, 0 \cdots 0\right)$ (1)
- Else If $\alpha < 0$ Return $(1, 0, \ldots, 0)$
- Else $(\alpha > 0)$
  - call Cholesky-or-Violation $\left(N - \frac{1}{\alpha}\vec{q}\vec{q}^T\right)$
  - If it returns V, return $\begin{bmatrix} \sqrt{\alpha} & \vec{0} \\ \frac{1}{\sqrt{\alpha}}\vec{q} & V \end{bmatrix}$
  - Else (it returns $\vec{v}$), return $\left(-\frac{1}{\alpha}\langle \vec{v}, \vec{q} \rangle, \vec{v}\right)$

Cholesky-or-Violation $([\alpha])$ (dim. 1)

- If $\alpha < 0$ return (1).

| Else ($\alpha \geq 0$) return $[\sqrt{\alpha}]$

Theorem If $M \succcurlyeq 0$, Cholesky-or-Violation returns U s.t. $M = UU^T$. Otherwise, it returns $\vec{v}$ s.t. $\vec{v}^T M \vec{v} < 0$.

Proof by induction on dim M: **Base case, dim. 1:**
If $M = [\alpha]$ for $\alpha \geq 0$, then indeed $U = [\sqrt{\alpha}]$ satisfies $[\sqrt{\alpha}][\sqrt{\alpha}] = [\alpha]$. Otherwise, for $\alpha < 0$, $1[\alpha]1 = \alpha < 0$.

Induction step. $M = \begin{bmatrix} \alpha & \vec{q}^T \\ \vec{q} & N \end{bmatrix}$ of dim $n+1$. If $\alpha < 0$, $(1, 0-0)$ satisfies $(1,0-0)\begin{bmatrix} \alpha & \vec{q}^T \\ \vec{q} & N \end{bmatrix}\begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \alpha < 0$. If $\alpha = 0$ and some $q_i > 0$ then $\vec{v} = \left(\frac{\max\{N_{ii}, 1\}}{q_i}, 0-0, \frac{-1}{\max\{N_{ii}, 1\}}, 0-0\right)$ satisfies $\vec{v}^T \begin{bmatrix} 0 & \vec{q}^T \\ \vec{q} & N \end{bmatrix} \vec{v} = -2 + \frac{N_{ii}}{\max\{N_{ii}, 1\}^2} \leq -2 + 1 = -1$. Similarly, if $q_i < 0$, $\vec{v} = \left(\frac{-\max\{N_{ii}, 1\}}{q_i}, 0-0, \frac{1}{\max\{N_{ii}, 1\}}, 0-0\right)$ satisfies $\vec{v}^T \begin{bmatrix} 0 & \vec{q}^T \\ \vec{q} & N \end{bmatrix} \vec{v} = -2 + \frac{N_{ii}}{\max\{N_{ii}, 1\}^2} \leq -1$ again. If $\alpha = 0$ and $\vec{q} = \vec{0}$, then by IH,

Cholesky-or-Violation either returns $\vec{v}$ s.t. $\vec{v}^T N \vec{v} < 0$ — in which case, indeed $(0, \vec{v}^T)\begin{bmatrix} 0 & \vec{0}^T \\ \vec{0} & N \end{bmatrix}\begin{bmatrix} 0 \\ \vec{v} \end{bmatrix} = \vec{v}^T N \vec{v} < 0$ — or it returns V s.t. $VV^T = N$. Then observe, $\begin{bmatrix} 0 & \vec{0}^T \\ \vec{0} & V \end{bmatrix}\begin{bmatrix} 0 & \vec{0}^T \\ \vec{0} & V^T \end{bmatrix} = \begin{bmatrix} 0 & \vec{0}^T \\ \vec{0} & VV^T \end{bmatrix} = \begin{bmatrix} 0 & \vec{0}^T \\ \vec{0} & N \end{bmatrix}$.

This leaves the case of $\alpha > 0$. Again, by IH, Cholesky-or-violation either returns $\vec{v}$ s.t. $\vec{v}^T(N - \frac{1}{\alpha}\vec{q}\vec{q}^T)\vec{v} < 0$ or V s.t. $VV^T = N - \frac{1}{\alpha}\vec{q}\vec{q}^T$. In the first case, $\left(-\frac{1}{\alpha}\langle \vec{v}, \vec{q}\rangle, \vec{v}^T\right)\begin{bmatrix} \alpha & \vec{q}^T \\ \vec{0} & N \end{bmatrix}\begin{bmatrix} -\frac{1}{\alpha}\langle \vec{v}, \vec{q}\rangle \\ \vec{v} \end{bmatrix}$
$= \alpha\left(\frac{1}{\alpha}\langle \vec{v}, \vec{q}\rangle\right)^2 - 2 \cdot \frac{1}{\alpha}\langle \vec{v}, \vec{q}\rangle\langle \vec{v}, \vec{q}\rangle + \vec{v}^T N \vec{v} = \vec{v}^T\left(N - \frac{1}{\alpha}\vec{q}\vec{q}^T\right)\vec{v} < 0$.
In the second (final) case, $\begin{bmatrix} \sqrt{\alpha} & \vec{0} \\ \frac{1}{\sqrt{\alpha}}\vec{q} & V \end{bmatrix}\begin{bmatrix} \sqrt{\alpha} & \frac{1}{\sqrt{\alpha}}\vec{q}^T \\ \vec{0} & V \end{bmatrix} = \begin{bmatrix} \alpha & \sqrt{\alpha}\frac{1}{\sqrt{\alpha}}\vec{q}^T \\ \sqrt{\alpha}\frac{1}{\sqrt{\alpha}}\vec{q} & VV^T + \frac{1}{\alpha}\vec{q}\vec{q}^T \end{bmatrix}$
$= \begin{bmatrix} \alpha & \vec{q}^T \\ \vec{q} & N \end{bmatrix}$ since $VV^T = N - \frac{1}{\alpha}\vec{q}\vec{q}^T$. ▨

Since Cholesky-or-Violation runs in time $O(n^3)$ for $n \times n$ M, this gives a separation oracle for the $X \succcurlyeq 0$ constraint, and we can (approximately) solve semidefinite programs to ε-accuracy in $\text{poly}(n, \log\frac{1}{\varepsilon})$ time using, e.g., the Ellipsoid algorithm.

Example: fastest mixing Markov chain

Here's an example of an interesting problem that is solvable by semidefinite optimization: Suppose $G=(V,E)$ is an undirected graph that defines "adjacent" states for a Markov chain. We wish to find a symmetric transition matrix P that minimizes the number of steps to approach the uniform distribution, which we know to be the stationary distribution. We know that this is equivalent to minimizing the second-largest eigenvalue of P (in magnitude). . Recall, this is $\max_{\vec{x}:\vec{x}\perp\vec{1}} \left|\frac{\vec{x}^T P\vec{x}}{\|\vec{x}\|_2^2}\right| = \lambda$. Equivalently, we wish to find P and λ that minimizes λ subject to the constraint $-\lambda\|\vec{x}\|_2^2 \le \vec{x}^T P\vec{x} \le \lambda\|\vec{x}\|_2^2$ for all $\vec{x}:\vec{x}\perp\vec{1}$. (and P is a valid transition matrix...) Note that we can project $\vec{x}$ to the subspace orthogonal to $\vec{1}$ by taking $\vec{y} = \vec{x} - \frac{1}{n}\vec{1}(\vec{1}^T\vec{x}) = (\widetilde{I_n - \frac{1}{n}\vec{1}\vec{1}^T})\vec{x}$ $\pi_{\perp}$

So, equivalently, $0 \le \vec{x}^T \pi_\perp^T (\lambda I_n - P)\pi_\perp \vec{x}$ for all $\vec{x}$

$0 \le \vec{x}^T \pi_\perp^T (P + \lambda I_n)\pi_\perp \vec{x}$ for all $\vec{x}$. These correspond to PSD constraints: $0 \preceq \lambda(\pi_\perp^T I_n \pi_\perp) - \pi_\perp^T P \pi_\perp$ and

$$0 \preceq \lambda(\pi_\perp^T \pi_\perp) + \pi_\perp^T P \pi_\perp.$$

The remaining constraints are linear: $\vec{1}^T P = \vec{1}^T$, $P_{ij} \ge 0\ \forall ij$, and $P_{ij} = 0$ if $(i,j)\notin E$ for $P = P^T$. (Note that $\vec{1}^T P = \vec{1}$ and $P = P^T$ imply $\pi_\perp^T P \pi_\perp = P - 2\frac{1}{n}\vec{1}\vec{1}^T P + \frac{1}{n^2}\vec{1}\vec{1}^T P \vec{1}\vec{1}^T = P - \frac{1}{n}\vec{1}\vec{1}^T$ and $\pi_\perp^T \pi_\perp = I_n - \frac{1}{n}\vec{1}\vec{1}^T$ also, so the constraints can be simplified.) In summary, the program is: minimize λ s.t. $0 \preceq \lambda(I_n - \frac{1}{n}\vec{1}\vec{1}^T) - P + \frac{1}{n}\vec{1}\vec{1}^T$, $0 \preceq \lambda(I_n - \frac{1}{n}\vec{1}\vec{1}^T) + P - \frac{1}{n}\vec{1}\vec{1}^T$, $\vec{1}^T P = \vec{1}^T$, $P_{ij} = 0\ \forall (i,j)\notin E$, and $P_{ij} \ge 0\ \forall (i,j)\in E$, where $P^T = P$.