Approximating Sparse Semidefinite Programs

Kevin Shu[1]

[1]Georgia Institute of Technology

Background on Sparse Semidefinite Programming

Semidefinite Programming

$$\begin{aligned} \text{minimize} \quad & \langle B^0, X \rangle \\ \text{such that} \quad & \langle B^\ell, X \rangle = b_\ell \quad \text{for } \ell \in \{1, \ldots, k\} \\ & X \succeq 0 \end{aligned}$$

- $X \succeq 0$ means that X is an $n \times n$ positive semidefinite matrix.
- The B^i are all $n \times n$ symmetric matrices.

Solving Semidefinite Programs

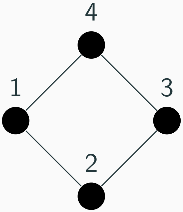
- Memory costs of solving semidefinite programs are often high in practice, especially when we need to compute Hessians.
- Part of the reason is that there are quadratically many variables.
- How can we improve performance costs of solving semidefinite programs?

Our notion of sparsity will always be parameterized by a graph, G.

Definition

A semidefinite program is G-**sparse** if it does not use the variables X_{ij} when $i, j \notin E(G)$ in the linear constraints or objective.

Example: Goemans-Williamson semidefinite programs are G-sparse. If $G = C_4$ is the 4-cycle



(a) The 4-cycle.

$$\begin{aligned} \text{minimize} \quad & X_{12} + X_{23} + X_{34} + X_{14} \\ \text{such that} \quad & X_{11} = X_{22} = X_{33} = X_{44} = 1 \\ & X \succeq 0 \end{aligned}$$

Figure 1: Goemans Williamson SDP for the 4-cycle

Sparsity

We define the G-partial matrices to be symmetric matrices, where entries corresponding to nonedges of G are 'forgotten'.

Definition

A G-partial matrix is **PSD-completable** if the missing entries can be chosen to make the resulting symmetric matrix PSD.

$\Sigma(G)$ is the convex cone of PSD completable G-partial matrices.

$$\begin{pmatrix} X_{11} & X_{12} & ? & X_{14} \\ X_{12} & X_{22} & X_{23} & ? \\ ? & X_{23} & X_{33} & X_{34} \\ X_{14} & ? & X_{34} & X_{44} \end{pmatrix}$$

Figure 2: A C_4-partial matrix.

Sparsity

G-sparse SDP's can be thought of as conic optimization problems over a projection of the PSD cone.

For example

$$\begin{aligned} \text{minimize} \quad & X_{12} + X_{23} + X_{34} + X_{14} \\ \text{such that} \quad & X_{11} = X_{22} = X_{33} = X_{44} = 1 \\ & X \succeq 0 \end{aligned}$$

can be rewritten

$$\begin{aligned} \text{minimize} \quad & X_{12} + X_{23} + X_{34} + X_{14} \\ \text{such that} \quad & X_{11} = X_{22} = X_{33} = X_{44} = 1 \\ & \begin{pmatrix} X_{11} & X_{12} & ? & X_{14} \\ X_{12} & X_{22} & X_{23} & ? \\ ? & X_{23} & X_{33} & X_{34} \\ X_{14} & ? & X_{34} & X_{44} \end{pmatrix} \in \Sigma(G) \end{aligned}$$

Sparsity

Optimization of G-sparse SDP's is equivalent to the problem of linear optimization over slices of $\Sigma(G)$.

Can we do this optimization without using the full SDP?

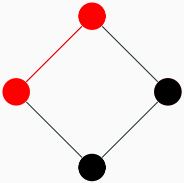
A Natural Relaxation

Definition

A G-partial matrix is G-**locally PSD** if all of its fully specified prinicipal submatrices are PSD.

$\mathcal{P}(G)$ is the convex cone of G-locally PSD matrices.

Fully specified prinicipal submatrices are in correspondence with cliques of G.



$$\begin{pmatrix} X_{11} & X_{12} & ? & X_{14} \\ X_{12} & X_{22} & X_{23} & ? \\ ? & X_{23} & X_{33} & X_{34} \\ X_{14} & ? & X_{34} & X_{44} \end{pmatrix}$$

A Natural Relaxation

If we have a G-sparse SDP,

$$\text{SDP} = \begin{array}{lll} \text{minimize} & \langle B^0, X\rangle & \\ \text{such that} & \langle B^\ell, X\rangle = b_\ell & \text{for } \ell \in \{1,\dots,k\} \\ & X \in \Sigma(G), & \end{array}$$

We will denote a modification

$$\text{SDP}^{SG} = \begin{array}{lll} \text{minimize} & \langle B^0, X\rangle & \\ \text{such that} & \langle B^\ell, X\rangle = b_\ell & \text{for } \ell \in \{1,\dots,k\} \\ & X \in \mathcal{P}(G) & \end{array}$$

A Natural Relaxation

Advantages

- Smaller PSD conditions are easier to check than larger PSD conditions.
- We don't need to consider variables that aren't in $E(G)$.

A Natural Relaxation

Advantages

- Smaller PSD conditions are easier to check than larger PSD conditions.
- We don't need to consider variables that aren't in $E(G)$.

Disadvantages

- Some graphs have exponentially many cliques
- The approximation need not be good.

Chordal Graphs and Equality

It is natural to ask when the above relaxation is exactly equal, i.e. when $\Sigma(G) = \mathcal{P}(G)$.

This was shown by Grone, Johnson, Sá, and Wolkowicz.

Definition

G is chordal if it has no induced cycles with more than 3 vertices.

Theorem

$\Sigma(G) = \mathcal{P}(G)$ if and only if G is chordal.

Chordal Graphs and Equality

Given a graph G, and a G-sparse SDP, it is standard practice to find a new graph G' that is chordal and contains G.

We can then think of a G-sparse SDP as a G' sparse SDP, and then use the above theorem to optimize over $\mathcal{P}(G)$ instead of $\Sigma(G)$.

Disadvantages

- Computing a chordal graph containing G with minimal number of edges is NP-hard (there are $O(\log(n))$-factor approximations though).
- Chordal graphs containing G might contain a lot more edges than G.

Approximate Semidefinite Programming

Key Question: How well does $\mathcal{P}(G)$ approximate $\Sigma(G)$?

For chordal G, these are equal.

In experimental settings, it is often seen that optimizing over $\mathcal{P}(G)$ is almost equivalent to optimizing over $\Sigma(G)$, even when G is not chordal. How can we quantify this?

Approximate PSDness

For any G, we let I_G be the projection of the identity onto the G-partial matrices.

Definition

If X is a G-partial matrix, then $\lambda(X)$ is the largest λ so that

$$X - \lambda I_G \in \Sigma(G).$$

- $\lambda(X)$ is the largest possible value of the minimum eigenvalue of $\hat{X}$, where $\hat{X}$ is a completion of X.
- $\lambda(X) \geq 0$ if and only if $X \in \Sigma(G)$.

Definition

For any G,

$$\epsilon(G) = \max\{-\lambda(X) : X \in \mathcal{P}(G),\ \mathrm{tr}(X) = 1\}.$$

This is 'how far from being PSD completable a matrix in $\mathcal{P}(G)$ can be'.

For any $X \in \mathcal{P}(G)$, $X + \epsilon(G)\,\mathrm{tr}(X) I_G$ is PSD completable.

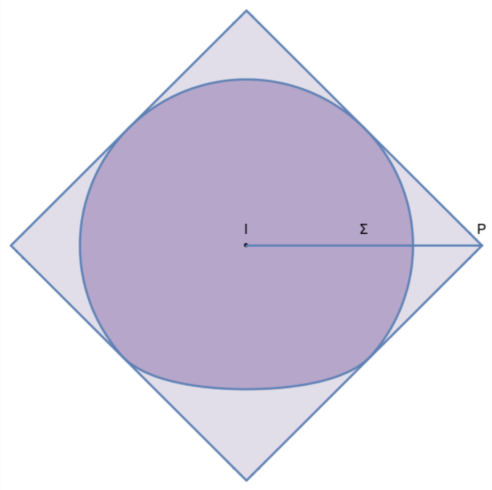


Figure 4: A geometric visualization of $\epsilon(G)$.

$\epsilon(G)$ is the smallest number so that

$$\tilde{\Sigma}(G) \subseteq \tilde{\mathcal{P}}(G) \subseteq (1 + n\epsilon(G))\tilde{\Sigma}(G).$$

Definition

A SDP is said to be of **Goemans-Williamson Type** if

- Every feasible point satisfies $\text{tr}(X) \leq n$.
- I_G is a feasible point.
- The trace of the objective is 0.
- It is a maximization problem.

Theorem

Let SDP be some G-sparse semidefinite program, and SDP^{SG} be its relaxation.

If α is the value of SDP, and α' is the value of SDP^{SG}, then

$$\alpha \leq \alpha' \leq (1 + n\epsilon(G))\alpha.$$

Advantages

- As long as $\epsilon(G)$ is $o(\frac{1}{n})$, we get a good approximation.
- Don't need chordal supergraphs, as long as we can enumerate the cliques of G.

Advantages

- As long as $\epsilon(G)$ is $o(\frac{1}{n})$, we get a good approximation.
- Don't need chordal supergraphs, as long as we can enumerate the cliques of G.

Disadvantages

- The approximation is not always good..
- Computing $\epsilon(G)$ is a concave minimization problem, which tend to be difficult.

What graphs have $\epsilon(G) = o(\frac{1}{n})$?

Computing $\epsilon(G)$

For cycles, we have

Theorem

$$\epsilon(C_n) = \frac{1}{n}\left(\frac{1}{\cos(\frac{\pi}{n})} - 1\right) = \theta(\frac{1}{n^3}).$$

This is the $-\lambda(X)$ where

$$X = \begin{pmatrix} 1 & 1 & ? & ? & \dots & -1 \\ 1 & 1 & 1 & ? & \dots & ? \\ ? & 1 & 1 & 1 & \dots & ? \\ & & \dots & & & \\ -1 & ? & ? & ? & \dots & 1 \end{pmatrix}.$$

Key idea is the cycle conditions, and the fact that they are convex in certain parameters.

Definition

The chordal girth of G is the smallest number of vertices in an induced cycle of G with at least 3 vertices, and ∞ if G is chordal.

We denote this by $\gamma(G)$.

Corollary

If G is series parallel, then

$$\epsilon(G) = \epsilon(C_{\gamma(G)}) = \theta(\frac{1}{\gamma(G)^3}).$$

Theorem

If G and H are graphs, and $K \subseteq G$ and $K \subseteq H$ are cliques, then we denote by $G \oplus H$ the **clique sum** of G and H.

$$\epsilon(G \oplus H) = \max\{\epsilon(G), \epsilon(H)\}.$$

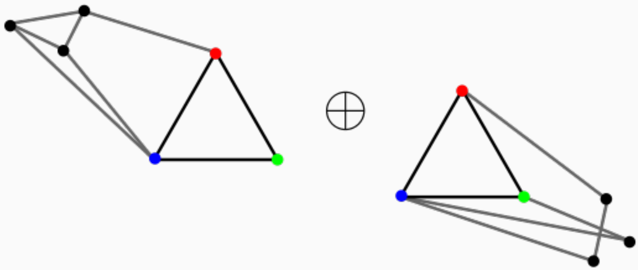


Figure 5: Clique sums are obtained by gluing together two graphs along a clique.

Theorem

If G is a graph, let $\hat{G}$ denote the **cone** over G, then

$$\epsilon(\hat{G}) = \epsilon(G).$$

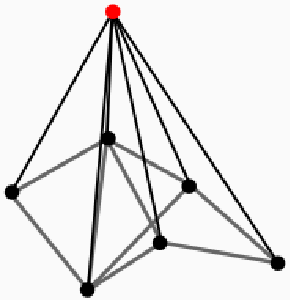


Figure 6: A cone of a graph is a graph that adds a single new vertex to G connected to all vertices of G.

Thickened Graphs

Definition

Given a graph G, a **thickening of** G is obtained by replacing the edges of G by chordal graphs with marked endpoints.

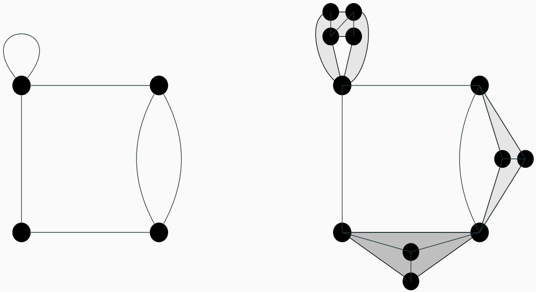


Figure 7: An example of a thickened graph. To the left, is a graph, and to the right is a thickening, where some of the edges have been replaced by other chordal graphs.

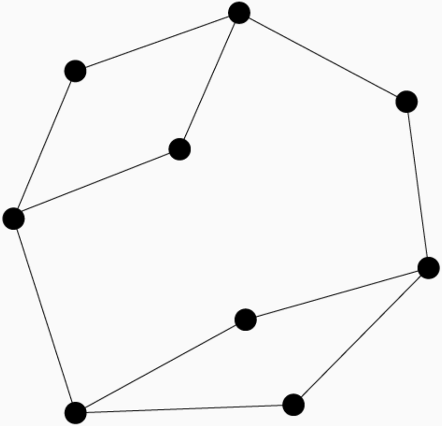
Theorem

Suppose that G is a thickened graph, and e is any edge of G. Let G/e be the contraction of G along the edge e.

$$\epsilon(G) \leq \epsilon(G/e).$$

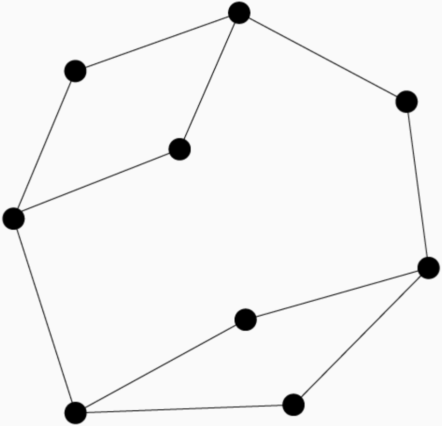
Completing to Thickened Graphs

If G is any graph, and we can break the graph down into pieces, and then find chordal covers of each piece separately to get a completion of G to a thickened graph.



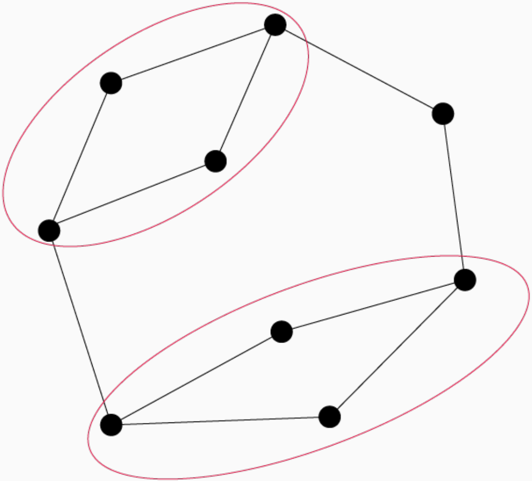
Completing to Thickened Graphs

If G is any graph, and we can break the graph down into pieces, and then find chordal covers of each piece separately to get a completion of G to a thickened graph.



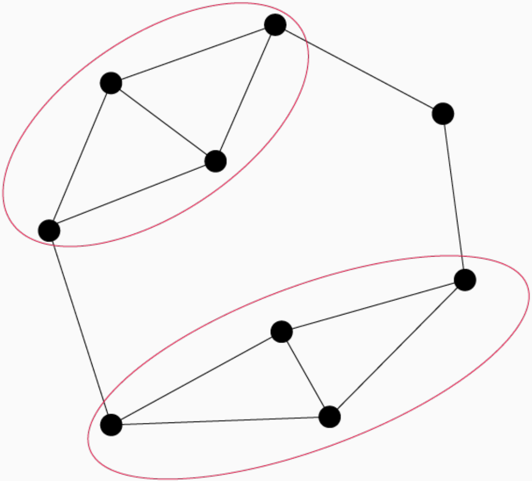
Completing to Thickened Graphs

If G is any graph, and we can break the graph down into pieces, and then find chordal covers of each piece separately to get a completion of G to a thickened graph.



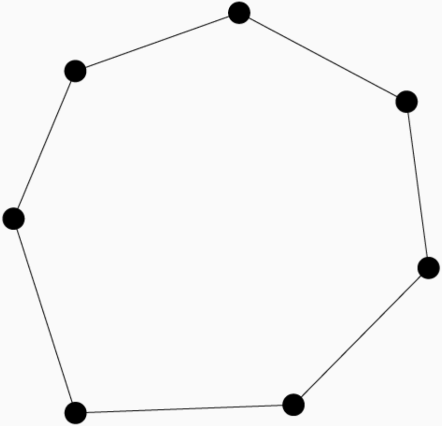
Completing to Thickened Graphs

If G is any graph, and we can break the graph down into pieces, and then find chordal covers of each piece separately to get a completion of G to a thickened graph.



Completing to Thickened Graphs

If G is any graph, and we can break the graph down into pieces, and then find chordal covers of each piece separately to get a completion of G to a thickened graph.



Theorem

If G is a graph, and H is a thickening of G obtained by replacing all of the edges of G by paths of length ℓ, then

$$\epsilon(H) \leq \epsilon(C_\ell)$$