Graph-Theoretic Convexification of Polynomial Optimization Problems: Theory and Applications

Javad Lavaei

Department of Electrical Engineering
Columbia University
Polynomial Optimization

**Different types of solutions:**

- **Point A:** Local solution
- **Point B:** Global solution
- **Point C:** Near-global solution

**Special case:** Combinatorial optimization and integer programming problems

Very hard to solve

**Polynomial Optimization:**

\[
\begin{align*}
\text{min} & \quad x^T M x \\
\text{s.t.} & \quad x_i^2 = 1, \quad i = 1, 2, \ldots, n
\end{align*}
\]
Objective

- **Focus of talk:** Find a near-global solution with a high optimality guarantee (close to 100%).

- **Problem 1: Convexification**
  Design a convex problem whose solution is near global for original problem.

- **Problem 2: Numerical Algorithm**
  Design an algorithm to solve the (high-dim) convex program numerically.

- **Approach:** Low-rank optimization, matrix completion, graph theory, convexification

Let’s see a real application before developing a rigorous theory.
Power system:

- A large-scale system consisting of generators, loads, lines, etc.
- Used for generating, transporting and distributing electricity.

ISO, RTO, TSO

1. Optimal power flow (OPF)
2. Security-constrained OPF
3. State estimation
4. Network reconfiguration
5. Unit commitment
6. Dynamic energy management

NP-hard
(real-time operation and market)
Optimal Power Flow:

- **Real-time operation:** OPF is solved every 5-15 minutes.
- **Market:** Security-constrained unit-commitment OPF
- **Complexity:** Strongly NP-complete with long history since 1962.
- **Common practice:** Linearization
- **FERC and NETSS Study:** Annual cost of approximation > $1 billion

A multi-billion critical system depends on optimization.
Convexification

- **Transformation**: Replace $xx^H$ with $W$.
- $W$ is positive semidefinite and rank 1.

- **Rank-1 SDP**: Recovery of a global solution $x$

- **Rank-1 penalized SDP**: Recovery of a near-global solution $x$
- SDP is not exact in general.
- SDP is exact for IEEE benchmark examples and several real data sets.

**Theorem:** Exact under positive LMPs with many transformers.

**Theorem:** Exact under positive LMPs.

Physics of power networks (e.g., passivity) reduces computational complexity for power optimization problems.

Promises of SDP

- **Observation**: SDP may not be exact for ISOs’ large-scale systems (some negative LMPs).

- **Remedy**: Design a penalized SDP to find a near-global solution.

<table>
<thead>
<tr>
<th>Case</th>
<th>Cost</th>
<th>Guarantee</th>
<th>Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Polish 2383wp</td>
<td>1874322.65</td>
<td>99.316%</td>
<td>529</td>
</tr>
<tr>
<td>Polish 2736sp</td>
<td>1308270.20</td>
<td>99.970%</td>
<td>701</td>
</tr>
<tr>
<td>Polish 2737sp</td>
<td>777664.02</td>
<td>99.995%</td>
<td>675</td>
</tr>
<tr>
<td>Polish 2746wp</td>
<td>1208453.93</td>
<td>99.985%</td>
<td>801</td>
</tr>
<tr>
<td>Polish 2746wp</td>
<td>1632384.87</td>
<td>99.962%</td>
<td>699</td>
</tr>
<tr>
<td>Polish 3012wp</td>
<td>2608918.45</td>
<td>99.188%</td>
<td>814</td>
</tr>
<tr>
<td>Polish 3120sp</td>
<td>2160800.42</td>
<td>99.073%</td>
<td>910</td>
</tr>
</tbody>
</table>

SDP looks very promising for energy applications

- **SDP revitalized the area**:
  - Follow-up work in academia
  - Interest from industry
  - Several talks at FERC’s summer workshops in 2012-14
  - One-day workshop on SDP at IBM Dublin

### Outline

<table>
<thead>
<tr>
<th>Arbitrary Real/Complex Polynomial Optimization</th>
<th>How does structure make SDP relaxation exact?</th>
<th>Complexity analysis based on generalized weighted graph</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conversion</td>
<td>Connection between sparsity and rank?</td>
<td>Proof of existence of low-rank solution using OS and treewidth</td>
</tr>
<tr>
<td>$\min_{x \in \mathbb{R}^n} \ x^H M_0 x$</td>
<td>How to design penalized SDP?</td>
<td>Propose two methods to design penalty</td>
</tr>
<tr>
<td>s.t. $x^H M_i x \leq a_i, \ i = 1, 2, \ldots, m$</td>
<td>Design scalable numerical algorithm?</td>
<td>Cheap iterations for large-scale problems</td>
</tr>
<tr>
<td>SDP/ Penalized SDP</td>
<td>Case Study: Optimal stochastic control</td>
<td>Implication for long-standing distributed decision making problem</td>
</tr>
<tr>
<td>$\min_{W} \ \text{trace}{M_0 W} + \lambda g(W)$</td>
<td>How to find a near-global solution for dense problems?</td>
<td>Find a sparse representation</td>
</tr>
<tr>
<td>s.t. $\text{trace}{M_i W} \leq a_i, \ i = 1, 2, \ldots, m$</td>
<td>$W \succeq 0$</td>
<td></td>
</tr>
</tbody>
</table>
Outline

Arbitrary Real/Complex Polynomial Optimization

Conversion

\[
\begin{align*}
\min_{x \in \mathbb{D}^n} & \quad x^H M_0 x \\
\text{s.t.} & \quad x^H M_i x \leq a_i, \quad i = 1, 2, \ldots, m
\end{align*}
\]

SDP/ Penalized SDP

How does structure make SDP relaxation exact?

Connection between sparsity and rank?

How to design penalized SDP?

Design scalable numerical algorithm?

Complexity analysis based on generalized weighted graph

Proof of existence of low-rank solution using OS and treewidth

Propose two methods to design penalty

Cheap iterations for large-scale problems

Case Study: Optimal stochastic control

How to find a near-global solution for dense problems?

Implication for long-standing distributed decision making problem

Find a sparse representation

How to find a near-global solution for dense problems?
**Problem:** How does structure affect computational complexity (e.g., positive coefficients)?

**Approach:** Map the structure into a *graph*.

Due to structure, SDP is always exact.

**Generalized weighted graph:**

\[
\min_{x_1, x_2} \quad x_1^4 + a_0 x_2^2 + b_0 x_1^2 x_2 + c_0 x_1 x_2 \\
\text{s.t.} \quad x_1^4 + a_i x_2^2 + b_i x_1^2 x_2 + c_i x_1 x_2 \leq \alpha_i, \quad i = 1, 2, \ldots, m
\]
Real-Valued Optimization

- Special cases:
  - **Positive optimization**: Bipartite graph
  - **Negative optimization**: Arbitrary graph

---

Complex-Valued Optimization

- **Real-valued case:** “$T$” is sign definite if $T$ and $-T$ are separable in $\mathbb{R}$:
- **Complex-valued case:** “$T$” is sign definite if $T$ and $-T$ are separable in $\mathbb{R}^2$:

Theorem: SDP is exact for acyclic graphs with sign definite sets and certain cyclic graphs.

- The proposed conditions include several existing ones ([Kim and Kojima, 2003], [Padberg, 1989], etc.).

---

Examples

Example 1: Physics of power grids reduces computational complexity.

Example 2: Graph idea generalizes to certain non-polynomial optimization problems.
## Outline

<table>
<thead>
<tr>
<th>Arbitrary Real/Complex Polynomial Optimization</th>
<th>How does structure make SDP relaxation exact?</th>
<th>Complexity analysis based on generalized weighted graph</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conversion</td>
<td>Connection between sparsity and rank?</td>
<td>Proof of existence of low-rank solution using OS and treewidth</td>
</tr>
</tbody>
</table>
| \[
\min_{x \in \mathbb{D}^n} \quad x^H M_0 x \\
\text{s.t.} \quad x^H M_i x \leq a_i, \quad i = 1, 2, ..., m
\] | How to design penalized SDP?                  | Propose two methods to design penalty |
| SDP/ Penalized SDP                              | Design scalable numerical algorithm?           | Cheap iterations for large-scale problems |
| \[
\min_W \quad \text{trace}\{M_0 W\} + \lambda g(W) \\
\text{s.t.} \quad \text{trace}\{M_i W\} \leq a_i, \quad i = 1, 2, ..., m \\
W \succeq 0
\] | Case Study: Optimal stochastic control         | Implication for long-standing distributed decision making problem |
|                                                | How to find a near-global solution for dense problems? | Find a sparse representation |

**Case Study: Optimal stochastic control**

Implication for long-standing distributed decision making problem.
**Graph Notions**

- **OS-vertex sequence:** [Hackney et al, 2009]
  - Partial ordering of vertices
  - Assume $O_1, O_2, ..., O_m$ is a sequence.
  - $O_i$ has a neighbor $w_i$ not connected to the connected component of $O_i$ in the subgraph induced by $O_1, ..., O_i$

- **Tree decomposition:** Map the graph $G$ into a tree $T$
  - Each node of $T$ is a bag of vertices of $G$
  - Each edge of $G$ appears in one node of $T$
  - If a vertex shows up in multiple nodes of $T$, those nodes should form a subtree

- **Width of $T$:** Max cardinality minus 1

- Roughly speaking, very sparse graphs have high OS and low treewidth (tree: OS=$n-1$, TW=1)

---

Low-Rank Solution

- **Sparsity Graph** $G$: Generalized weighted graph with no weights.
- SDP may have infinitely many solutions.
- How to find a low-rank solution (if any)?
- Consider a supergraph $G'$ of $G$.

**Theorem:** Every solution of perturbed SDP satisfies the following:

$$\text{Rank}\{W^{\text{opt}}\} \leq |G'| - \min_{G_s} \{\text{OS}(G_s) \mid (G' - G) \subseteq G_s \subseteq G'\}$$

- **Equal bags:** $\text{TW}(G)+1$ for a right choice of $G'$
- **Unequal bags:** Needs nonlinear penalty to attain $\text{TW}(G)+1$

This result includes the recent work *Laurent and Varvitsiotis, 2012.*
Tree decomposition for IEEE 14-bus system:

Case studies:

<table>
<thead>
<tr>
<th>System $\mathcal{G}$</th>
<th>$tw{\mathcal{G}}$</th>
<th>System $\mathcal{G}$</th>
<th>Bound on $tw{\mathcal{G}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>IEEE 14-bus</td>
<td>2</td>
<td>Polish 2383wp</td>
<td>23</td>
</tr>
<tr>
<td>IEEE 30-bus</td>
<td>3</td>
<td>Polish 2736sp</td>
<td>23</td>
</tr>
<tr>
<td>New England 39-bus</td>
<td>3</td>
<td>Polish 2746wp</td>
<td>23</td>
</tr>
<tr>
<td>IEEE 57-bus</td>
<td>5</td>
<td>Polish 3012wp</td>
<td>24</td>
</tr>
<tr>
<td>IEEE 118-bus</td>
<td>4</td>
<td>Polish 3120sp</td>
<td>24</td>
</tr>
<tr>
<td>IEEE 300-bus</td>
<td>6</td>
<td>Polish 3375wp</td>
<td>25</td>
</tr>
</tbody>
</table>

SDP relaxation of every SC-UC-OPF problem solved over NY grid has rank less than 40 (size of $W$ varies from 8500 to several millions).

Outline

**Arbitrary Real/Complex Polynomial Optimization**

**Conversion**

**How does structure make SDP relaxation exact?**
- Complexity analysis based on generalized weighted graph

**Connection between sparsity and rank?**
- Proof of existence of low-rank solution using OS and treewidth

**How to design penalized SDP?**
- Propose two methods to design penalty

**Design scalable numerical algorithm?**
- Cheap iterations for large-scale problems

**SDP/ Penalized SDP**

min \( x^H M_0 x \)

s.t. \( x^H M_i x \leq a_i, \quad i = 1, 2, ..., m \)

**Case Study: Optimal stochastic control**
- Implication for long-standing distributed decision making problem

min \( \text{trace}\{M_0 W\} + \lambda g(W) \)

s.t. \( \text{trace}\{M_i W\} \leq a_i, \quad i = 1, 2, ..., m \)

\( W \succeq 0 \)

**How to find a near-global solution for dense problems?**
- Find a sparse representation
Non-convexity Localization

\[
\begin{align*}
\min_{x \in \mathbb{D}^n} & \quad x^H M_0 x \\
\text{s.t.} & \quad x^H M_i x \leq a_i, \quad i = 1, 2, \ldots, m
\end{align*}
\]

Sparse

\[
\begin{align*}
\min_W & \quad \text{trace} \{ M_0 W \} \\
\text{s.t.} & \quad \text{trace} \{ M_i W \} \leq a_i, \quad i = 1, 2, \ldots, m \\
& \quad W \succeq 0
\end{align*}
\]

Low-rank

\[
\begin{align*}
\min_W & \quad \text{trace} \{ M_0 W \} + \lambda g(W) \\
\text{s.t.} & \quad \text{trace} \{ M_i W \} \leq a_i, \quad i = 1, 2, \ldots, m \\
& \quad W \succeq 0
\end{align*}
\]

Rank-1

SDP works if $G$ has no edges:

\[
\frac{x^2}{y_i} \quad \Rightarrow \quad y_i
\]

(LP)

• Assume SDP fails.
• Can we identify what edges caused the failure?
• Localized non-convexity v.s. uniform non-convexity?

Approach for localized case:
Penalty over problematic edges
Problematic Edges

Problematic edges: Identified based on high-rank submatrices

IEEE 300-bus: 2
Polish 2383-bus: 11

**Example: Near-Global Solutions**

**Strategy:** Penalize reactive loss over problematic lines

- **Modified IEEE 118-bus:**
  - 3 local solutions
  - Costs: 129625, 177984, 195695

<table>
<thead>
<tr>
<th>Case</th>
<th>TW</th>
<th>Cost</th>
<th>Guarantee</th>
<th>Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chow’s 9 bus</td>
<td>2</td>
<td>5296.68</td>
<td>100%</td>
<td>≤ 5</td>
</tr>
<tr>
<td>IEEE 14 bus</td>
<td>2</td>
<td>8081.53</td>
<td>100%</td>
<td>≤ 5</td>
</tr>
<tr>
<td>IEEE 24 bus</td>
<td>4</td>
<td>63352.20</td>
<td>100%</td>
<td>≤ 5</td>
</tr>
<tr>
<td>IEEE 30 bus</td>
<td>3</td>
<td>576.89</td>
<td>100%</td>
<td>≤ 5</td>
</tr>
<tr>
<td>NE 39 bus</td>
<td>3</td>
<td>41864.40</td>
<td>99.994%</td>
<td>≤ 5</td>
</tr>
<tr>
<td>IEEE 57 bus</td>
<td>5</td>
<td>41737.78</td>
<td>100%</td>
<td>≤ 5</td>
</tr>
<tr>
<td>IEEE 118 bus</td>
<td>4</td>
<td>129660.81</td>
<td>99.995%</td>
<td>≤ 5</td>
</tr>
<tr>
<td>IEEE 300 bus</td>
<td>6</td>
<td>719725.10</td>
<td>99.998%</td>
<td>13.9</td>
</tr>
<tr>
<td>Polish 2383wp</td>
<td>23</td>
<td>1874322.65</td>
<td>99.316%</td>
<td>529</td>
</tr>
<tr>
<td>Polish 2736sp</td>
<td>23</td>
<td>1308270.20</td>
<td>99.970%</td>
<td>701</td>
</tr>
<tr>
<td>Polish 2737sp</td>
<td>23</td>
<td>777664.02</td>
<td>99.995%</td>
<td>675</td>
</tr>
<tr>
<td>Polish 2746wp</td>
<td>23</td>
<td>1208453.93</td>
<td>99.985%</td>
<td>801</td>
</tr>
<tr>
<td>Polish 2746wp</td>
<td>24</td>
<td>1632384.87</td>
<td>99.962%</td>
<td>699</td>
</tr>
<tr>
<td>Polish 3012wp</td>
<td>24</td>
<td>2608918.45</td>
<td>99.188%</td>
<td>814</td>
</tr>
<tr>
<td>Polish 3120sp</td>
<td>24</td>
<td>2160800.42</td>
<td>99.073%</td>
<td>910</td>
</tr>
</tbody>
</table>

Penalty Design

Why was penalty chosen as loss?

\[
\min_W \text{ trace}\{M_0 W\} + \lambda g(W) \\
\text{s.t. } \text{ trace}\{M_i W\} \leq a_i, \quad i = 1, 2, ..., m \\
W \succeq 0
\]

Proposed penalty:

\[g(W) = \text{trace}\{MW\}\]

First try:

\[g(W) = \|W\|_*\]

- Compressed sensing and phase retrieval
- Need \(n \log n\) measurements for a much simpler problem [Candes and Recht, 2009].

Algorithm design: Can we design an SDP to find the best \(M\)?

Good penalty: Minimization of penalty by itself \((\lambda = \infty)\) leads to a rank-1 solution.

Study of a simpler case:

\[
\min_W \text{ trace}\{MW\} \\
\text{s.t. } \text{ trace}\{M_i W\} = a_i, \quad i = 1, 2, ..., n \\
W \succeq 0
\]

Guess for solution of original QCQP: \(x_*\)

- \(M \succeq 0\)
- \(Mx_* = 0\)
- Zero is a simple eig of \(M\).
Theorem: If Jacobian is nonsingular, then SDP is exact in a vicinity of $x^*$.

Local behavior: Linearization solves approximately but SDP solves exactly.

Global behavior: The region could be as big as the entire space.

Recoverable region for $x$:

$$R_M = \{ x \mid g(x, M) \geq 0 \}$$

Design of $M$: Include $x^*$ and a set of points

Power flow equations for power systems: $M$ is a one-time design independent of loads.


Outline

Arbitrary Real/Complex Polynomial Optimization

Conversion

SDP/ Penalized SDP

How does structure make SDP relaxation exact?

How to design penalized SDP?

How to find a near-global solution for dense problems?

Connection between sparsity and rank?

Proof of existence of low-rank solution using OS and treewidth

Propose two methods to design penalty

Cheap iterations for large-scale problems

Implication for long-standing distributed decision making problem

Find a sparse representation

Design scalable numerical algorithm?

Complexity analysis based on generalized weighted graph

Case Study: Optimal stochastic control
**Low-Complex Algorithm**

**Goal:** Design a low-complex algorithm for sparse LP/QP/QCQP/SOCP/SDP

- **Distributed Algorithm:** ADMM-based dual decomposed SDP (related work: [Parikh and Boyd, 2014], [Wen, Goldfarb and Yin, 2010], [Andersen, Vandenberghe and Dahl, 2010]).

- **Iterations:** Closed-form solution for every iteration (eigen-decomposition on submatrices)

---

Example

- Number of blocks (agents): 2000
- Size of each block: 40
- Number of constraints per block: 5
- Overlapping degree: 25%
- Number of entries for full SDP: 6.4B
- Number of entries for decomposed SDP: Over 3M
- Number of constraints: Several thousands

- 20 minutes in MATLAB with cold start (2.4 GHz and 8 GB):

99.9% feasible and globally optimal
Outline

Arbitrary Real/Complex Polynomial Optimization

Conversion

\[
\begin{align*}
\min_{x \in \mathbb{D}^n} & \quad x^H M_0 x \\
\text{s.t.} & \quad x^H M_i x \leq a_i, \quad i = 1, 2, \ldots, m
\end{align*}
\]

SDP/ Penalized SDP

How does structure make SDP relaxation exact?

Connection between sparsity and rank?

How to design penalized SDP?

Design scalable numerical algorithm?

Case Study: Optimal stochastic control

How to find a near-global solution for dense problems?

Complexity analysis based on generalized weighted graph

Proof of existence of low-rank solution using OS and treewidth

Propose two methods to design penalty

Cheap iterations for large-scale problems

Implication for long-standing distributed decision making problem

Find a sparse representation
Distributed Control of Stochastic Systems

Distributed control
(NP-hard: Witsenhausen’s example)

Stochastic Distributed Control: Design \( u[\tau] = K y[\tau] \) for

\[
\begin{cases}
    y[\tau] = C x[\tau] + F v[\tau]
\end{cases}
\]

to minimize:

\[
\lim_{\tau \to +\infty} \mathcal{E} (x[\tau]^T Q x[\tau] + u[\tau]^T R u[\tau])
\]

Theorem: Rank of SDP solution in the Lyapunov domain is 1, 2 or 3.

New England Test System

Outline

Arbitrary Real/Complex Polynomial Optimization

Conversion

How does structure make SDP relaxation exact?

Connection between sparsity and rank?

Complexity analysis based on generalized weighted graph

Proof of existence of low-rank solution using OS and treewidth

How to design penalized SDP?

Propose two methods to design penalty

Design scalable numerical algorithm?

Cheap iterations for large-scale problems

Case Study: Optimal stochastic control

Implication for long-standing distributed decision making problem

How to find a near-global solution for dense problems?

Find a sparse representation
**Sparsification**

- **What if the optimization under study is not sparse?**

  ![Polyomial Optimization ↔ Dense QCQP ↔ Sparse QCQP](image)

  **Technique 1:** Vertex Duplication Procedure

  \[
  x_i \iff (x_{i1}, x_{i2}) \quad \text{s.t.} \quad x_{i1} = x_{i2}
  \]

  **Technique 2:** Edge Elimination Procedure

  \[
  x_i x_j \iff z_1^2 - z_2^2 \quad \text{s.t.} \quad z_1 = \frac{x_i + x_j}{2}, \quad z_2 = \frac{x_i - x_j}{2}
  \]

- The treewidth can be reduced to 1 thru sparsification.

**Theorem:** Every polynomial optimization has a quadratic formulation whose SDP relaxation has a solution with rank 1 or 2.

- Sparsification is useful for finding approximation ratio but the price is loss of performance.

---

Conclusions

Problem: Find a near-global solution together with a global optimality guarantee

Approach: Graph-theoretic convexification

- **Generalized weighted graph**: Connection between complexity and structure
- **OS and treewidth**: Connection between rank and sparsity
- **Non-convexity diagnosis**: Graph-based localization
- **Penalized SDP**: Obtaining a near-global solution
- **Scalable algorithm**: High-dimensional sparse SDP
- **Sparsification**: Rank reduction for dense optimization
- **Applications**: Power optimization and stochastic control
Future Work: Incomplete List

Energy:
- Find approximation ratio for power optimization (99% ?).
- Study rounding techniques for mixed-integer problems (UC-OPF).
- Software development
- Collaboration with industry

Theory:
- Compute approximation ratio (and infeasibility degree) based on low-rank optimization.
- Systematic rounding procedure.
- Connection to sum-of-squares, valid inequalities, ...
- Stochastic problems and robust optimization
- Case studies: Hard graph problems

Applications in other areas:
- Big data, machine learning, societal problems, etc.
Funding Acknowledgements

- **ONR YIP**: Graph-theoretic and low-rank optimization

- **NSF CAREER**: Control and optimization for power systems

- **NSF EPCN**: Contingency analysis for power systems

- **Google**: Numerical algorithms for nonlinear optimization
Collaborators

Caltech and UT Austin:
- John Doyle
- Richard Murray
- Steven Low
- Ross Baldick

Stanford, Washington and NYU:
- Stephen Boyd
- David Tse
- Baosen Zhang
- Somayeh Sojoudi

Columbia:
- Ramtin Madani
- Abdulrahman Kalbat
- Salar Fattahi
- Morteza Ashraphijuo
Thank You