POLYNOMIAL OPTIMIZATION WITH SUMS-OF-SQUARES INTERPOLANTS

Sercan Yıldız
syildiz@samsi.info

in collaboration with
Dávid Papp (NCSU)

OPT Transition Workshop
May 02, 2017
• Polynomial optimization and sums of squares
  - Sums-of-squares hierarchy
  - Semidefinite representation of sums-of-squares constraints
• Interior-point methods and conic optimization
• Sums-of-squares optimization with interior-point methods
  - Computational complexity
  - Preliminary results
Polynomial Optimization

- Polynomial optimization problem:

  \[
  \min_{z \in \mathbb{R}^n} \quad f(z) \\
  \text{s.t.} \quad g_i(z) \geq 0 \text{ for } i = 1, \ldots, m
  \]

  where \( f, g_1, \ldots, g_m \) are \( n \)-variate polynomials

Example:

  \[
  \min_{z \in \mathbb{R}^2} \quad z_1^3 + 3z_1^2z_2 - 6z_1z_2^2 + 2z_2^3 \\
  \text{s.t.} \quad z_1^2 + z_2^2 \leq 1
  \]

- Some applications:
  - Shape-constrained estimation
  - Design of experiments
  - Control theory
  - Combinatorial optimization
  - Computational geometry
  - Optimal power flow
Sums-of-Squares Relaxations

- Let $f$ be an $n$-variate degree-$2d$ polynomial.
- Unconstrained polynomial optimization:
  $$\min_{z \in \mathbb{R}^n} f(z)$$

- Equivalent “dual” formulation:
  $$\max_{y \in \mathbb{R}} \ y$$
  s.t. \( f(z) - y \in P \)
  where \( P = \{ f : f(z) \geq 0 \ \forall z \in \mathbb{R}^n \} \)

- Sums-of-squares cone:
  $$\text{SOS} = \{ f : f = \sum_{j=1}^{N} f_j^2 \text{ for some degree-}d \text{ polynomials } f_j \}$$
  \( f \in \text{SOS} \implies f \in P \)

- SOS relaxation:
  $$\max_{y \in \mathbb{R}} \ y$$
  s.t. \( f(z) - y \in \text{SOS} \)

NP-hard already for \( d = 2! \)
**Sums-of-Squares Relaxations**

- Let $f, g_1, \ldots, g_m$ be $n$-variate polynomials.
- Constrained polynomial optimization:

$$\min_{z \in \mathbb{R}^n} f(z)$$

s.t. $g_i(z) \geq 0$ for $i = 1, \ldots, m$

- Feasible set: $G = \{z \in \mathbb{R}^n : g_i(z) \geq 0$ for $i = 1, \ldots, m\}$.
- Dual formulation:

$$\max_{y \in \mathbb{R}} y$$

s.t. $f(z) - y \in P_G$

where $P_G = \{f : f(z) \geq 0 \quad \forall z \in G\}$

- “Weighted” SOS cone of order $r$:

$$SOS_{G,r} = \{f : f = \sum_{i=0}^{m} g_i s_i, \quad s_i \in SOS, \quad \deg(g_i s_i) \leq r\} \quad \text{where} \quad g_0 \equiv 1$$

$$f \in SOS_{G,r} \quad \Rightarrow \quad f \in P_G$$

- SOS relaxation of order $r$:

$$\max_{y \in \mathbb{R}} y$$

s.t. $f(z) - y \in SOS_{G,r}$
Why Do We Like SOS?

- Polynomial optimization is NP-hard.
- Increasing $r$ produces a **hierarchy of SOS relaxations** for polynomial optimization problems.
- Under mild assumptions:
  - The lower bounds from SOS relaxations **converge** to the true optimal value as $r \uparrow \infty$ (follows from Putinar’s Positivstellensatz).
- SOS relaxations can be represented as **semidefinite programs** of size $O(ml^2)$ where $L = \binom{n+r/2}{n}$ (follows from the results of Shor, Nesterov, Parrilo, Lasserre).
- There are efficient and stable numerical methods for solving SDPs.
Consider the cone of degree-$2d$ SOS polynomials:

\[
\text{SOS}_{2d} = \{ f \in \mathbb{R}[z]_{2d} : f = \sum_{j=1}^{N} f_j^2 \text{ for some } f_j \in \mathbb{R}[z]_d \}
\]

**Theorem (Nesterov, 2000)**

The univariate polynomial \( f(z) = \sum_{u=0}^{2d} \bar{f}_u z^u \) is SOS iff there exists a \((d + 1) \times (d + 1)\) PSD matrix \( S \) such that

\[
\bar{f}_u = \sum_{k+\ell=\ell} S_{k\ell} \quad \forall \, u = 0, \ldots, 2d.
\]
More generally, in the $n$-variate case:
- Let $L := \dim(\mathbb{R}[z]_d) = \binom{n+d}{n}$ and $U := \dim(\mathbb{R}[z]_{2d}) = \binom{n+2d}{n}$.
- Fix bases $\{p_\ell\}_{\ell=1}^L$ and $\{q_u\}_{u=1}^U$ for the linear spaces $\mathbb{R}[z]_d$ and $\mathbb{R}[z]_{2d}$.

**Theorem (Nesterov, 2000)**
The polynomial $f(z) = \sum_{u=1}^U \bar{f}_u q_u(z)$ is SOS iff there exists a $L \times L$ PSD matrix $S$ such that
\[
\bar{f} = \Lambda^*(S)
\]
where $\Lambda : \mathbb{R}^U \to \mathbb{S}^L$ is the linear map satisfying $\Lambda([q_u]_{u=1}^U) = [p_\ell]_{\ell=1}^L [p_\ell]_{\ell=1}^L \top$.

- If univariate polynomials are represented in the **monomial basis**:
\[
\Lambda(x) = \begin{bmatrix}
x_0 & x_1 & x_2 & \ldots & x_d \\
x_1 & x_2 & \ldots & & \\
x_2 & & \ddots & & \\
\vdots & & & \ddots & \\
x_d & & & & x_{2d-1}
\end{bmatrix}
\quad \Lambda^*(S) = \left[\sum_{k+\ell=u} S_{k\ell}\right]_{u=0}^{2d}
\]

- These results easily extend to the weighted case.
SOS CONSTRAINTS ARE SEMIDEFINITE REPRESENTABLE

- To keep things simple, we focus on optimization over a single SOS cone.
- SOS problem:
  \[
  \begin{align*}
  \max_{y \in \mathbb{R}^k, S \in S^L} & \quad b^T y \\
  \text{s.t.} & \quad A^T y + \Lambda^*(S) = c \\
  & \quad S \succeq 0
  \end{align*}
  \]
  \[
  \quad A^T y + s = c \\
  \quad s \in \text{SOS} := \{s \in \mathbb{R}^U : s = \Lambda^*(S), S \succeq 0\}
  \]
- Moment problem:
  \[
  \begin{align*}
  \min_{x \in \mathbb{R}^U} & \quad c^T x \\
  \text{s.t.} & \quad Ax = b \\
  & \quad \Lambda(x) \succeq 0
  \end{align*}
  \]
  \[
  x \in \text{SOS}^* := \{x \in \mathbb{R}^U : \Lambda(x) \succeq 0\}
  \]

Disadvantages of existing approaches

- **Problem:** SDP representation roughly squares the number of variables.
  **Solution:** We solve the SOS and moment problems in their original space.
- **Problem:** Standard basis choices lead to ill-conditioning.
  **Solution:** We use orthogonal polynomials and interpolation bases.
INTERIOR-POINT METHODS FOR CONIC PROGRAMMING

- Primal-dual pair of conic programs:

\[
\begin{align*}
\min \ & c^T x \\
\text{s.t.} \ & Ax = b \\
& x \in K
\end{align*}
\]

\[
\begin{align*}
\max \ & b^T y \\
\text{s.t.} \ & A^T y + s = c \\
& s \in K^*
\end{align*}
\]

- \(K\): closed, convex, pointed cone with nonempty interior

Examples: \(K = \mathbb{R}_+^n, S_+^n, \text{SOS}, P\)

- Interior-point methods make use of **self-concordant barriers (SCBs)**.

Examples:
- \(K = \mathbb{R}_+^n\): \(F(x) = -\log x\)
- \(K = S_+^n\): \(F(X) = -\log \det X\)
Given SCBs $F$ and $F^*$, IPMs converge to the optimal solution by solving a sequence of equality-constrained barrier problems for $\mu \downarrow 0$:

$$\begin{align*}
\min & \quad c^\top x + \mu F(x) \\
\text{s.t.} & \quad Ax = b
\end{align*}$$

and

$$\begin{align*}
\max & \quad b^\top y - \mu F^*(s) \\
\text{s.t.} & \quad A^\top y + s = c
\end{align*}$$

- **Primal** IPMs:
  - solve only the primal barrier problem,
  - are not considered to be practical.

- **Primal-dual** IPMs:
  - solve both the primal and dual barrier problems simultaneously,
  - are preferred in practice.
• In principle, any closed convex cone admits a SCB (Nesterov and Nemirovski, 1994).

• However, the success of the IPM approach depends on the availability of a SCB whose \textbf{gradient} and \textbf{Hessian} can be computed efficiently.
  – For the cone $P$, there are complexity-based reasons for suspecting that there are no computationally tractable SCBs.
  – For the cone $SOS$, there is evidence suggesting that there may not exist any tractable SCBs.
  – On the other hand, the cone $SOS^*$ inherits a tractable SCB from the PSD cone.
Until recently:
- The practical success of primal-dual IPMs had been limited to optimization over *symmetric* cones: LP, SOCP, SDP.
- Existing primal-dual IPMs for *non-symmetric* conic programs required *both* the primal and dual SCBs (e.g., Nesterov, Todd, and Ye, 1999; Nesterov, 2012).

Skajaa and Ye (2015) proposed a primal-dual IPM for non-symmetric conic programs which
- requires a SCB for only the *primal cone*,
- achieves the *best-known iteration complexity*. 
Using Skajaa and Ye’s IPM with the SCB for $SOS^*$, the SOS and moment problems can be solved without recourse to SDP.

For any $\epsilon \in (0, 1)$, the algorithm finds a primal-dual solution that has $\epsilon$ times the duality gap of an initial solution in $O(\sqrt{L} \log(1/\epsilon))$ iterations where $L = \dim(\mathbb{R}[x]_d)$.

Each iteration of the IPM requires
- the computation of the Hessian of the SCB for $SOS^*$,
- the solution of a Newton system.

Solving the Newton system requires $O(U^3)$ operations where $U = \dim(\mathbb{R}[z]_{2d})$. 
The choice of bases \( \{ p_\ell \}_{\ell=1}^L \) and \( \{ q_u \}_{u=1}^U \) for \( \mathbb{R}[z]_d \) and \( \mathbb{R}[z]_{2d} \) has a significant effect on how efficiently the Newton system can be compiled.

- In general, computing the Hessian requires \( O(L^2U^2) \) operations.
- If both bases are chosen to be monomial bases, the Hessian can be computed faster but requires specialized methods such as FFT and the “inversion” of Hankel-like matrices.
- Following Löfberg and Parrilo (2004), we choose
  - \( \{ q_u \}_{u=1}^U \): Lagrange interpolating polynomials,
  - \( \{ p_\ell \}_{\ell=1}^L \): orthogonal polynomials.
- With this choice, the Hessian can be computed in \( O(LU^2) \) operations.
• Putting everything together:
  – The algorithm runs in $O(\sqrt{L}\log(1/\epsilon))$ iterations.
  – At each iteration:
    • Computing the Hessian requires $O(LU^2)$ operations,
    • Solving the Newton system requires $O(U^3)$ operations.
• Overall complexity: $O(U^3\sqrt{L}\log(1/\epsilon))$ operations.
• This matches the best-known complexity bounds for LP!
• In contrast:
  – Solving the SDP formulation with a primal-dual IPM requires $O(L^{6.5}\log(1/\epsilon))$ operations.
  – For fixed $n$:
    $$\frac{L^2}{U} = \Theta(d^n).$$
The conditioning of the moment problem is directly related to the conditioning of the interpolation problem with \( \{p_k p_\ell\}_{k, \ell=1}^L \).

Good interpolation nodes are understood well only in few low-dimensional domains:
- Chebyshev points in \([-1, 1]\),
- Padua points in \([-1, 1]^2\) (Caliari et al., 2005).

For problems in higher dimensions, we follow a heuristic approach to choose interpolation nodes.
A TOY EXAMPLE

- Minimizing the Rosenbrock function on the square:

\[
\min_{z \in \mathbb{R}^2} \ (z_1 - 1)^2 + 100(z_2 - z_1^2)^2
\]

s.t. \( z \in [-1, 1]^2 \)

- We can obtain a lower bound on the optimal value of this problem from the moment relaxation of order \( r \).

- For \( r = 60 \):
  - The moment relaxation has 1891 variables.
  - The SDP representation requires one 496 \( \times \) 496 and two 494 \( \times \) 494 matrix inequalities.
  - Sedumi quits with numerical errors after 948 seconds. Primal infeasibility: \( 1.2 \times 10^{-4} \).
  - Our implementation solves the moment relaxation in 482 seconds.
Final Remarks

- We have shown that SOS programs can be solved using a primal-dual IPM in $O(U^3 \sqrt{L} \log(1/\epsilon))$ operations where $L = \binom{n+d}{n}$ and $U = \binom{n+2d}{n}$.
- This improves upon the standard SDP-based approach which requires $O(L^{6.5} \log(1/\epsilon))$ operations.
- In progress: Numerical stability.
  - For multivariate problems, the conditioning of the problem formulation becomes important.
  - How do we choose interpolation nodes for better-conditioned problems?

Thank you!


