Fast ADMM for Sum of Squares Programs Using Partial Orthogonality

Antonis Papachristodoulou
Department of Engineering Science
University of Oxford

www.eng.ox.ac.uk/control/sysos
antonis@eng.ox.ac.uk

with Yang Zheng (U Oxford) and Giovanni Fantuzzi (Imperial CL)

EPSRC
Engineering and Physical Sciences Research Council

UNIVERSITY OF OXFORD

BBSRC
bioscience for the future

OUTLINE

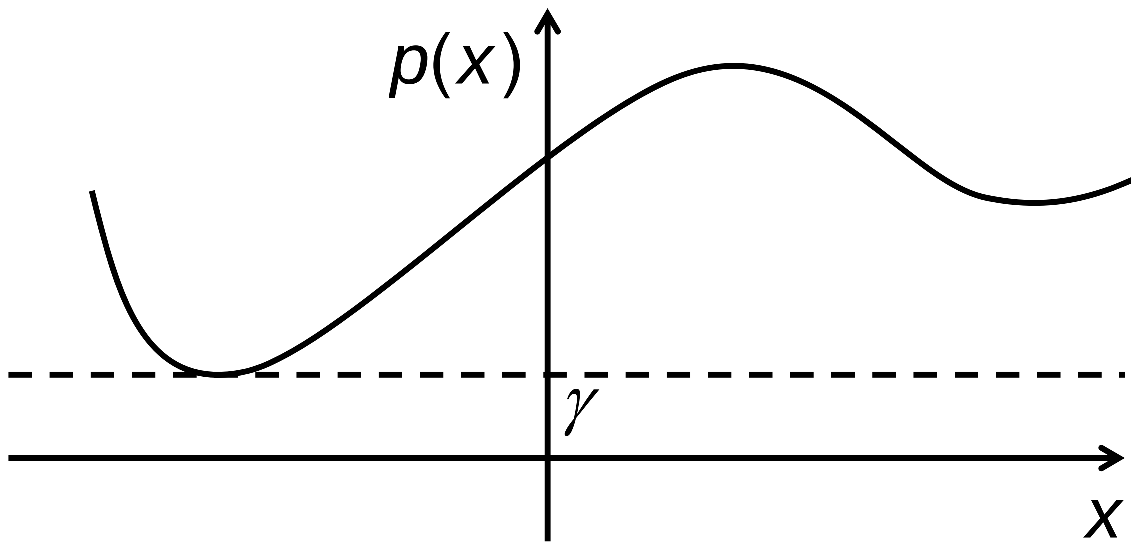
Outline

Polynomial Optimisation

$$\text{Is } p(x) = \sum_{\alpha} p_{\alpha} x^{\alpha} \geq 0\,?$$

- Unconstrained polynomial optimisation

$$\min_{x \in \mathbb{R}^n} \; p(x) \quad \Longleftrightarrow \quad \begin{aligned} \max_{x \in \mathbb{R}^n} \quad & \gamma \\ \text{s.t.} \quad & p(x) - \gamma \geq 0 \end{aligned}$$



- Other areas: graph partition, matrix copositivity test, etc.

Analysis in Dynamical Systems

Lyapunov:

$$\dot{x} = f(x),\ f(0) = 0,\ f \text{ Lipschitz},$$

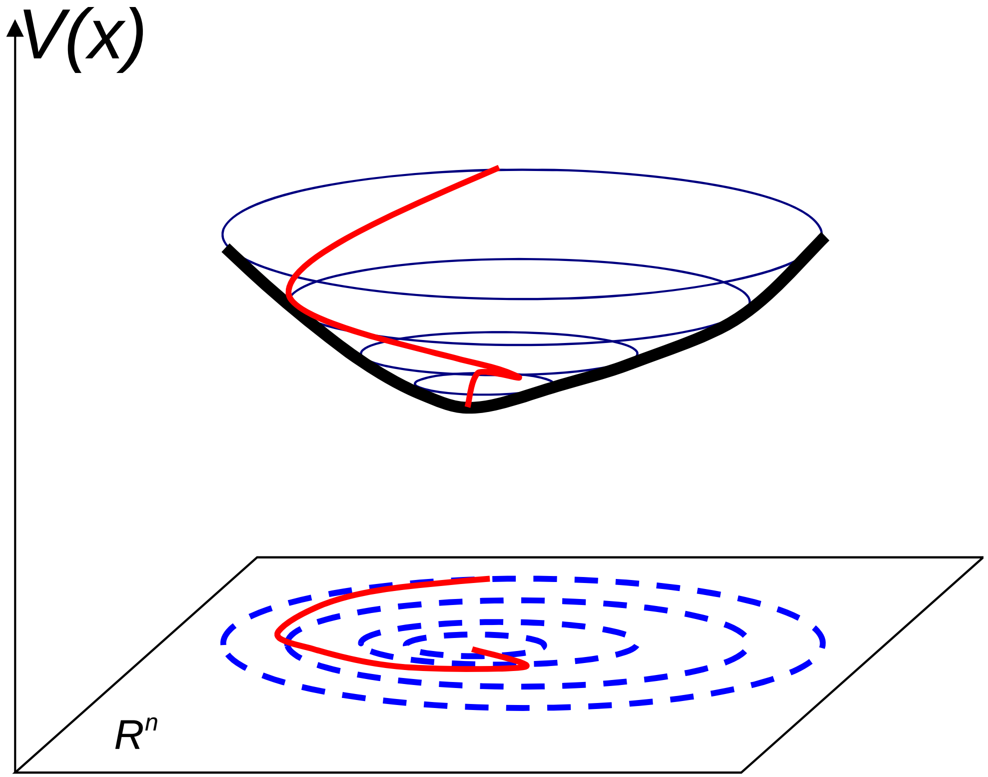
$$x(t) \in D \subseteq \mathbb{R}^{n_x},\ 0 \in D$$

If $\exists\ V : D \to R$ cont. differentiable s.t.

$$V(0) = 0, \text{ and } V(x) > 0 \text{ in } D \setminus \{0\}$$

$$-\dot{V}(x) = -\frac{\partial V(x)}{\partial x} f(x) > 0 \text{ in } D \setminus \{0\}$$

then $x = 0$ is asymptotically stable.



Checking if $p(x) \geq 0$ is NP-hard when $\deg(p) \geq 4$, $p \in \mathbb{R}[x]$.

Positive Polynomials and Sum of Squares

Shor: $p(x)$ is Sum of Squares $\Rightarrow$ $p(x)$ is positive semi-definite

$$p(x) = \sum_{i=1}^{m} q_i^2(x) \underset{\not\Leftarrow}{\Rightarrow} p(x) \geq 0$$

Worst-case polynomial time complexity

NP-hard when $\deg(p) \geq 4$

Can be solved using semidefinite programming (SDP) (Parrilo), which can be setup and solved using SOSTOOLS:

www.eng.ox.ac.uk/control/sostools

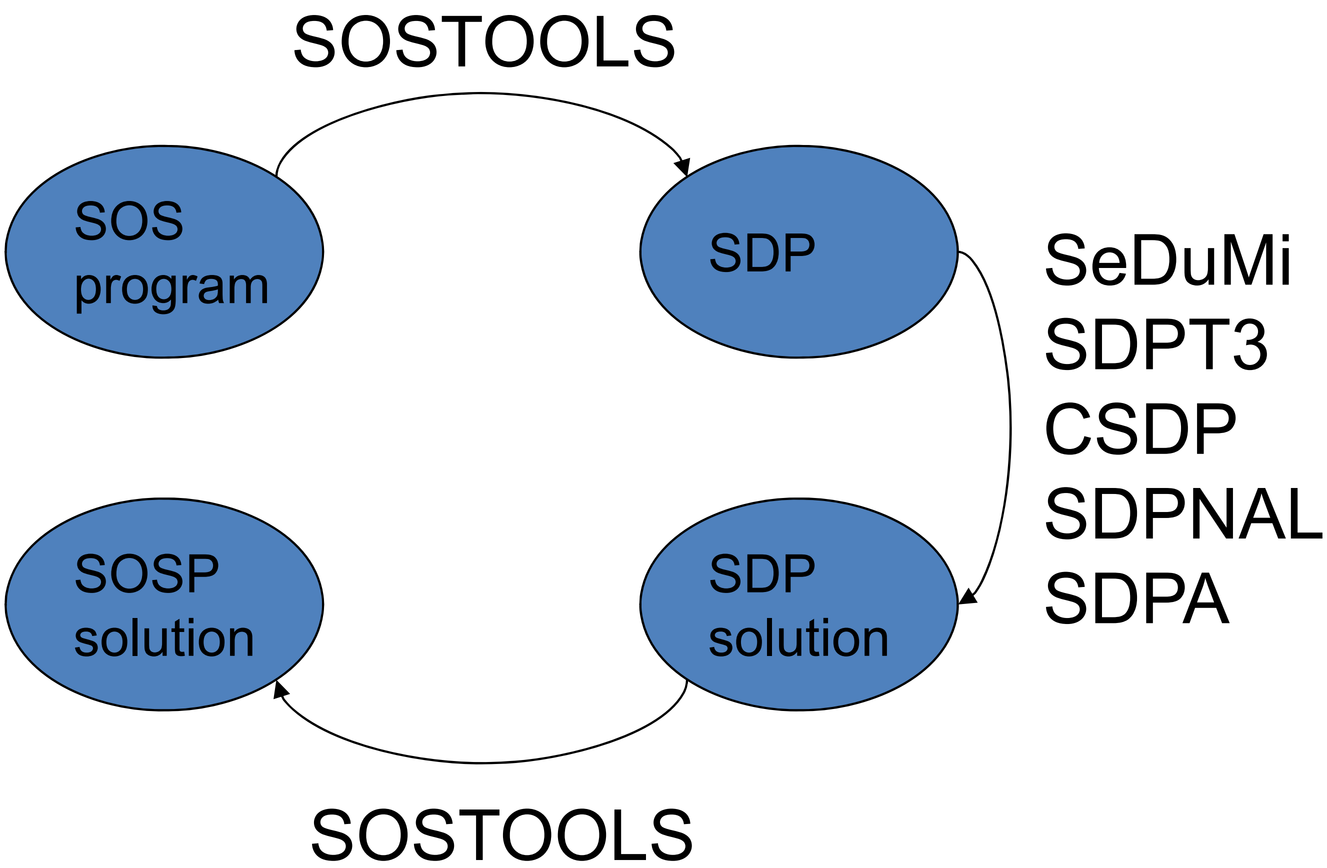
Can also search for unknown coefficients of $p(x)$ so that $p(x)$ is SOS

P., Anderson, Valmorbida, Prajna, Seiler, Parrilo, 2013

SOSTOOLS

Formulates and solves the equivalent semidefinite programme (SDP)

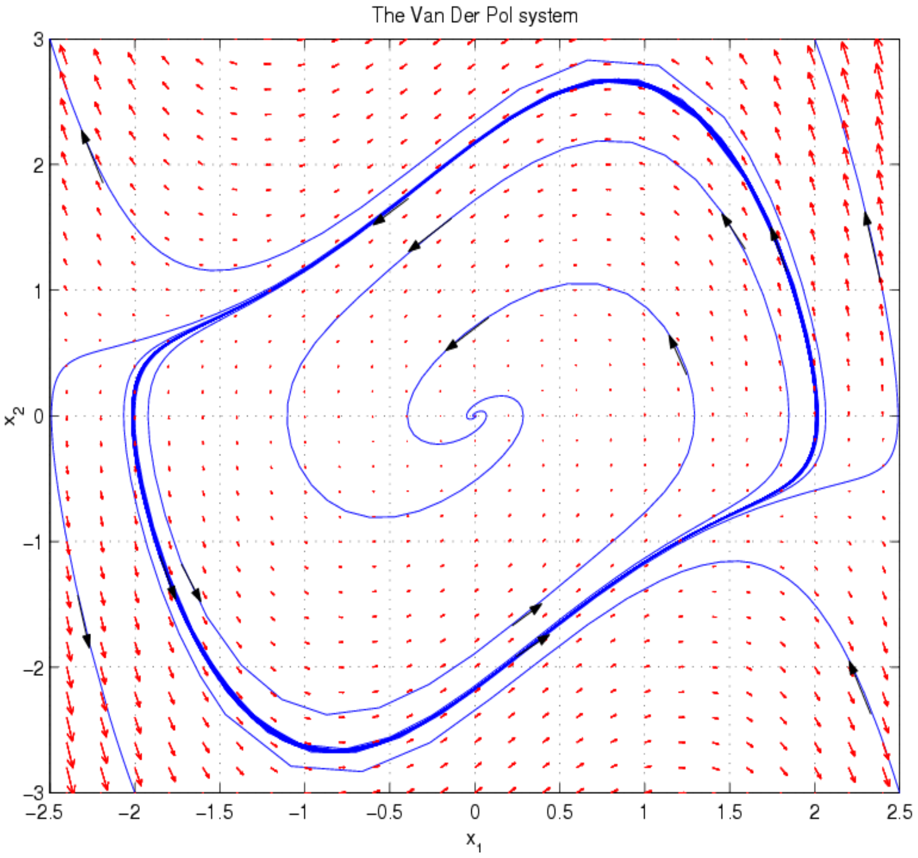
www.eng.ox.ac.uk/control/sostools

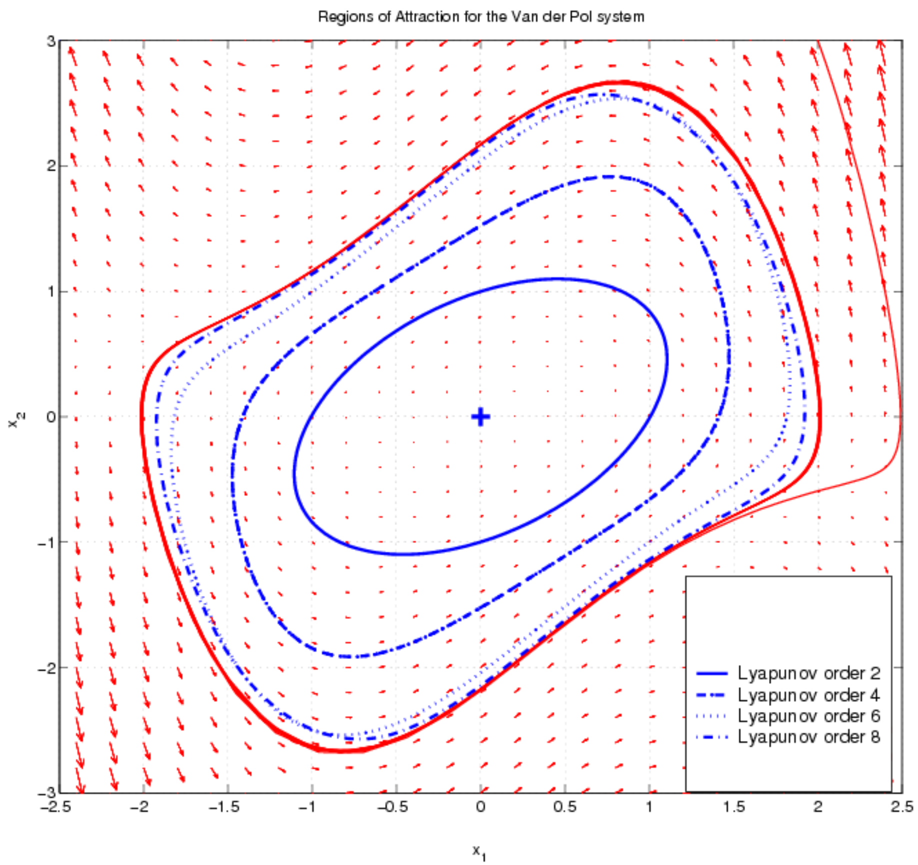


Van der Pol Oscillator

$$\dot{x}_1 = -x_2$$

$$\dot{x}_2 = x_1 - (1 - x_1^2)x_2$$





Sum of Squares and Semidefinite Programming

$$p(x) = \sum_{i=1}^{m} q_i^2(x) = v_d(x)^T X v_d(x)$$

Proposition: p is SOS $\Leftrightarrow \exists X \geq 0,\ v_d(x)$ monomials of degree $\dfrac{\deg(p)}{2}$

Example: $p(x_1, x_2) = 5x_1^4 + 2x_2^4 - x_1^2 x_2^2 - 2x_1^3 x_2 - 2x_1 x_2^3$

$$= \begin{bmatrix} x_1^2 \\ x_2^2 \\ x_1 x_2 \end{bmatrix}^T \underbrace{\begin{bmatrix} q_{11} & q_{12} & q_{13} \\ q_{12} & q_{22} & q_{23} \\ q_{13} & q_{23} & q_{33} \end{bmatrix}}_{X} \underbrace{\begin{bmatrix} x_1^2 \\ x_2^2 \\ x_1 x_2 \end{bmatrix}}_{v_d(x)}$$

- Expand out and match coefficients:

  $q_{11} = 5,\ \ q_{22} = 2,\ \ 2q_{12} + q_{33} = -1,\ q_{13} = -1,\ q_{23} = -1$

- Require that $X \geq 0$

What restricts SOS methods?

Computation

$$p(x) = \sum_{i=1}^{m} q_i^2(x) = v_d(x)^T X v_d(x),\ x \in \mathbb{R}^n$$

$$v_d(x) = \left\{ x^\alpha \mid \alpha \in \mathbb{N}_d^n \right\}$$

$$= \begin{bmatrix} 1 & x_1 & x_2 & \cdots & x_n & x_1^2 & x_1 x_2 & \cdots & x_n^d \end{bmatrix}, \text{ of size } \binom{n+d}{d}.$$

- Reduce the size of $v_d(x)$? Newton polytope, diagonal inconsistency, symmetry properties, and facial reduction
- Alternative formulation (DSOS/SDSOS), leading to linear programs (LPs) or second-order-cone programs (SOCPs)
- Solving the SDPs by more scalable first-order methods (FOMs) at the cost of reduced accuracy.

Outline

Formalism

$$s(x) = g_0(x) - \sum_{i=1}^{t} u_i g_i(x) \text{ is SOS}$$

$$\text{iff } s(x) = v_d(x)^T X v_d(x) = \left\langle X, v_d(x) v_d(x)^T \right\rangle, X \geq 0$$

- Indicator matrices $\left(B_\alpha\right)_{\beta,\gamma} = \begin{cases} 1 & \text{if } \beta + \gamma = \alpha \\ 0 & \text{otherwise} \end{cases}$
- Then $v_d(x) v_d(x)^T = \sum_\alpha B_\alpha x^\alpha$
- And therefore $\sum_\alpha \left(g_{0,\alpha} - \sum_{i=1}^{t} u_i g_{i,\alpha} \right) x^\alpha = \sum_\alpha \left\langle B_\alpha, X \right\rangle x^\alpha$
- Coefficient Matching Conditions $\left(g_{0,\alpha} - \sum_{i=1}^{t} u_i g_{i,\alpha} \right) = \left\langle B_\alpha, X \right\rangle$

Our strategy

$$\min \quad w^T u$$
$$\text{s. t.} \quad s(x) = g_0(x) - \sum_{i=1}^{t} u_i g_i(x),$$
$$s \in \Sigma[x]$$

$$\min \quad w^T u$$
$$\text{s. t.} \quad \langle B_\alpha, X \rangle + \sum_{i=1}^{t} u_i g_{i,\alpha} = g_{0,\alpha}, \; \forall \alpha \in \mathbb{N}^n_{2d},$$
$$X \geq 0$$

- Size of the SDP

n	4	6	8	10	12	14	16	18
$2d = 2$	(5, 15)	(7, 28)	(9, 45)	(11, 66)	(13, 91)	(15, 120)	(17, 153)	(19, 190)
$2d = 4$	(15, 70)	(28, 210)	(45, 495)	(66, 1001)	(91, 1820)	(120, 3060)	(153, 4845)	(190, 7315)
$2d = 6$	(35, 210)	(84, 924)	(165, 3003)	(286, 8008)	(455, 18564)	(680, 38760)	(969, 74613)	(1330, 134596)

- Partial orthogonality in the resulting Semidefinite Programme

- A fast first order (ADMM) algorithm that exploits the partial orthogonality – trades accuracy for scalability.

- Implemented in CDCS-sos, downloadable

Standard SDP formulation

$$\begin{aligned} \min \quad & w^T u \\ \text{s. t.} \quad & \langle B_\alpha, X \rangle + \sum_{i=1}^{t} u_i g_{i,\alpha} = g_{0,\alpha,}, \ \forall \alpha \in \mathbb{N}_{2d}^n, \\ & X \geq 0 \end{aligned}$$

- Notation

$$A_1 = \begin{bmatrix} g_{1,1} & \cdots & g_{t,1} \\ \vdots & \ddots & \vdots \\ g_{1,m} & \cdots & g_{t,m} \end{bmatrix}, \quad A_2 = \begin{bmatrix} vec(B_1)^T \\ \vdots \\ vec(B_m)^T \end{bmatrix}$$

$$A \doteq [A_1 \ A_2] \in R^{m \times (t+N^2)},$$
$$b \doteq [g_{0,1}, \ldots, g_{0,m}]^T \in R^m,$$
$$c \doteq [w^T, 0, \ldots, 0]^T \in R^{t+N^2},$$
$$\xi \doteq [u^T, vec(X)^T]^T \in R^{t+N^2},$$
$$\mathcal{K} \doteq R^t \times S_+$$

- Standard SDP formulation

$$\begin{aligned} \min_\xi \quad & c^T \xi \\ \text{s. t.} \quad & A\xi = b, \\ & \xi \in \mathcal{K} \end{aligned}$$

Standard SDP formulation

$$\begin{aligned} \min_{\xi} \quad & c^T \xi \\ \text{s. t.} \quad & A\xi = b, \\ & \xi \in K \end{aligned}$$

$$A_1 = \begin{bmatrix} g_{1,1} & \cdots & g_{t,1} \\ \vdots & \ddots & \vdots \\ g_{1,m} & \cdots & g_{t,m} \end{bmatrix}, \quad A_2 = \begin{bmatrix} vec(B_1)^T \\ \vdots \\ vec(B_m)^T \end{bmatrix}$$

$$A \doteq [A_1 \; A_2] \in R^{m\times(t+N^2)}$$

- The density of nonzero elements in A_2 is $O\left(\frac{1}{n^{2d}}\right)$.

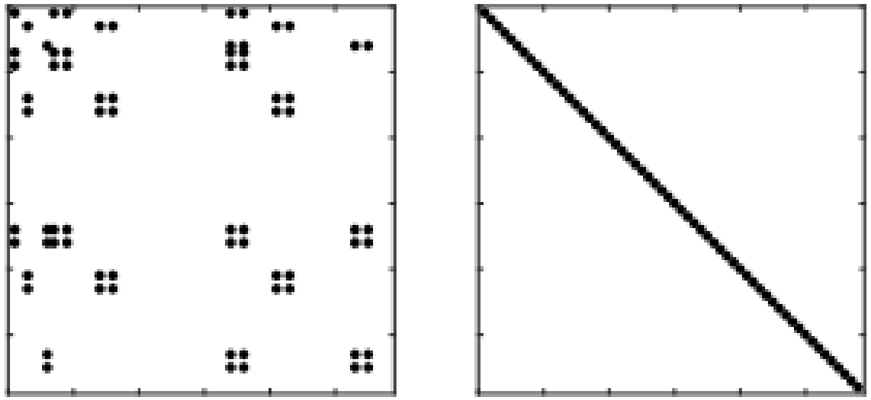
n	4	6	8	10	12	14	16
$2d = 4$	1.42×10^{-2}	4.76×10^{-3}	2.02×10^{-3}	9.99×10^{-4}	5.49×10^{-4}	3.27×10^{-4}	2.06×10^{-4}
$2d = 6$	4.76×10^{-3}	1.08×10^{-3}	3.33×10^{-4}	1.25×10^{-4}	5.39×10^{-5}	2.58×10^{-5}	1.34×10^{-5}
$2d = 8$	2.02×10^{-3}	3.33×10^{-4}	7.77×10^{-5}	2.29×10^{-5}	7.94×10^{-6}	3.13×10^{-6}	1.36×10^{-6}

- $A_2 A_2^T$ is diagonal.

- The $m \times m$ matrix AA^T is of "diagonal plus low rank" form.

Partial Orthogonality

$$\left(g_{0,\alpha} - \sum_{i=1}^{t} u_i g_{i,\alpha}\right) = B_\alpha, \quad A_1 = \begin{bmatrix} g_{1,1} & \cdots & g_{t,1} \\ \vdots & \ddots & \vdots \\ g_{1,m} & \cdots & g_{t,m} \end{bmatrix}, \quad A_2 = \begin{bmatrix} vec(B_1)^T \\ \vdots \\ vec(B_m)^T \end{bmatrix}$$



$A_1 A_1^T$ $\qquad$ $A_2 A_2^T$

1. D. Bertsimas, R. M. Freund, and X. A. Sun, “An accelerated first order method for solving sos relaxations of unconstrained polynomial optimization problems,” Optimization Methods and Software, vol. 28, no. 3, pp. 424–441, 2013.
2. D. Henrion and J. Malick, “Projection methods in conic optimization,” in Handbook on Semidefinite, Conic and Polynomial Optimization. Springer, 2012, pp. 565–600.

Outline

ADMM for the homogeneous self-dual embedding

$$\begin{aligned} \min \quad & w^T u \\ \text{s. t.} \quad & \langle B_\alpha, X \rangle + \sum_{i=1}^{t} u_i g_{i,\alpha} = g_{0,\alpha,}, \ \forall \alpha \in \mathbb{N}_{2d}^n, \\ & X \geq 0 \end{aligned}$$

$$\begin{aligned} \min_{\xi} \quad & c^T \xi \\ \text{s.t.} \quad & A\xi = b \\ & \xi \in \mathcal{K} \end{aligned} \qquad \begin{aligned} \max_{y} \quad & b^T y \\ \text{s.t.} \quad & A^T y + z = c \\ & z \in \mathcal{K}^* \end{aligned}$$

- KKT conditions
  - Primal feasibility

$$A\xi = b, \quad \xi \in \mathcal{K}$$

  - Dual feasibility

$$A^T y + z = c, \ z \in \mathcal{K}^*$$

  - Zero-duality gap

$$c^T \xi - b^T y = 0$$

$$\begin{bmatrix} z \\ s \\ \kappa \end{bmatrix} = \begin{bmatrix} 0 & -A^T & c \\ A & 0 & -b \\ -c^T & b^T & 0 \end{bmatrix} \begin{bmatrix} \xi \\ y \\ \tau \end{bmatrix}$$

ADMM for the Homogeneous self-dual embedding

$$\begin{bmatrix} z \\ s \\ \kappa \end{bmatrix} = \begin{bmatrix} 0 & -A^T & c \\ A & 0 & -b \\ -c^T & b^T & 0 \end{bmatrix} \begin{bmatrix} \xi \\ y \\ \tau \end{bmatrix}$$

τ, κ are two non-negative and complementary variables

- Notational simplicity

$$v \triangleq \begin{bmatrix} z \\ s \\ \kappa \end{bmatrix}, \; u \triangleq \begin{bmatrix} \xi \\ y \\ \tau \end{bmatrix}, \; Q \triangleq \begin{bmatrix} 0 & -A^T & c \\ A & 0 & -b \\ -c^T & b^T & 0 \end{bmatrix}, \; \mathcal{C} = \mathcal{K} \times R^m \times R_+$$

- Feasibility problem

$$\begin{aligned} \text{find} \quad & (u,v) \\ \text{s.t.} \quad & v = Qu \\ & (u,v) \in \mathcal{C} \times \mathcal{C}^* \end{aligned}$$

ADMM for the Homogeneous self-dual embedding

$$\begin{aligned} &\text{find} \quad (u,v) \\ &\text{s.t.} \quad v = Qu, \quad (u,v) \in \mathcal{C} \times \mathcal{C}^* \end{aligned}$$

- ADMM steps (similar to solver SCS [1])

$$\hat{u}^{k+1} = (I+Q)^{-1}(u^k + v^k)$$ → Affine projection

$$u^{k+1} = \mathrm{P}_{\mathcal{C}}(\hat{u}^{k+1} - v^k)$$ → Conic projection

$$v^{k+1} = v^k - \hat{u}^{k+1} + u^{k+1}$$

$$Q \triangleq \begin{bmatrix} 0 & -A^T & c \\ A & 0 & -b \\ -c^T & b^T & 0 \end{bmatrix}$$

✓ Affine projection takes advantage of diagonal plus low rank for efficient computation

[1] O'Donoghue, B., Chu, E., Parikh, N. and Boyd, S. (2016). Conic optimization via operator splitting and homogeneous self-dual embedding. Journal of Optimization Theory and Applications, 169(3), 1042– 1068

ADMM for the Homogeneous self-dual embedding

$$\hat{u}^{k+1} = (I+Q)^{-1}(u^k + v^k)$$ → Affine projection

$$u^{k+1} = P_{\mathcal{C}}(\hat{u}^{k+1} - v^k)$$

$$v^{k+1} = v^k - \hat{u}^{k+1} + u^{k+1}$$

- In affine projection, one needs to invert/factorize

$$I + AA^T \in R^{m \times m}$$

- Recall that AA^T is "diagonal plus low rank"

$$(I + AA^T)^{-1} = (P + A_1 A_1^T)^{-1} = P^{-1} - P^{-1} A_1 (I + A_1^T P^{-1} A_1)^{-1} A_1^T P^{-1}$$

$$I + A_1^T P^{-1} A_1 \in R^{t \times t}$$

- Since in typical SOS programs $t \ll m$, partial orthogonality can provide a speed-up in the affine projection.

OUTLINE

CDCS: Cone Decomposition Conic Solver

- An open source MATLAB solver for partially decomposable conic programs;
- CDCS supports constraints on the following cones:
  - ✓ Free variables
  - ✓ non-negative orthant
  - ✓ second-order cone
  - ✓ the positive semidefinite cone.
- Input-output format is aligned with SeDuMi;
- SOS module can be invoked with option ‘sos’
- Works with latest YALMIP release.

Syntax:

```
[x,y,z,info] = cdcs(At,b,c,K,opts);
```

Download from https://github.com/OxfordControl/CDCS

Numerical Examples

- Implemented in CDCS, invoked by the option 'sos'
- Compared to SCS and SeDuMi, SDPA, SDPT3
- Stopping condition for CDCS and SCS: $\varepsilon = 10^{-4}$, 2000 iterations

$$\min_x \sum_{1\le i<j\le n} \left(x_i x_j + x_i^2 x_j - x_j^3 - x_i^2 x_j^2\right)$$

$$\text{s. t.} \quad \sum_{i=1}^{n} x_i^2 \le 1$$

	Dimensions			CPU time (s)						
n	N	m	t	SeDuMi	SDPT3	SDPA	CSDP	SCS-direct	SCS-indirect	CDCS-sos
10	66	1 000	66	2.6	2.1	1.6	2.5	0.4	0.4	0.4
12	91	1 819	91	12.3	7.0	5.7	4.0	0.7	0.8	0.7
14	120	3 059	120	68.4	24.2	18.1	13.5	1.7	1.7	1.4
17	171	5 984	171	516.9	129.6	97.9	75.8	4.6	4.4	3.5
20	231	10 625	231	2 547.4	494.1	452.7	374.2	10.6	10.6	8.5
24	325	20 474	325	**	**	2 792.8	2 519.3	32.0	31.2	22.8
29	465	40 919	465	**	**	**	**	125.9	126.3	67.1
35	666	82 250	666	**	**	**	**	425.3	431.3	216.9
42	946	163 184	946	**	**	**	**	1 415.8	1 436.9	686.6

**: the problem could not be solved due to memory limitations.

Numerical Examples

$$\min_x \sum_{1 \le i < j \le n} \left(x_i x_j + x_i^2 x_j - x_j^3 - x_i^2 x_j^2 \right)$$

$$\text{s. t.} \quad \sum_{i=1}^{n} x_i^2 \le 1$$

n	†Interior-point solvers	SCS-direct		SCS-indirect		CDCS-sos	
	Objective	Objective	Accuracy	Objective	Accuracy	Objective	Accuracy
10	−9.114	−9.124	0.10 %	−9.125	0.11 %	−9.090	0.27 %
12	−11.117	−11.095	0.20 %	−11.095	0.19 %	−11.106	0.10 %
14	−13.118	−13.089	0.22 %	−13.093	0.19 %	−13.155	0.29 %
17	−16.119	−16.087	0.20 %	−16.088	0.19 %	−16.062	0.35 %
20	−19.120	−19.165	0.24 %	−19.167	0.25 %	−19.078	0.22 %
24	−23.121	−23.043	0.34 %	−23.038	0.36 %	−23.145	0.10 %
29	**	−28.174	—	−28.178	—	−28.170	—
35	**	−34.054	—	−34.052	—	−34.075	—
42	**	−41.212	—	−41.214	—	−41.052	—

†: The objective values computed by SeDuMi, SDPT3, SDPA and CSDP (when available) differ by less than 10^{-8}.
**: The problem could not be solved due to memory limitations.
—: No comparison is possible due to the lack of a reference accurate optimal value.

Lyapunov function construction

- Randomly generated polynomial dynamical systems $\dot{x} = f(x)$ of degree three with a locally asymptotically stable equilibrium at the origin.

	Dimensions			CPU time (s)						
n	N	m	t	SeDuMi	SDPT3	SDPA	CSDP	SCS-direct	SCS-indirect	CDCS-sos
10	65	1 100	110	2.8	1.8	2.0	2.6	0.2	0.2	0.3
12	90	1 963	156	6.3	4.9	3.5	1.0	0.3	0.3	0.4
14	119	3 255	210	36.2	16.3	44.8	2.6	0.8	0.7	0.6
17	170	6 273	306	265.1	78.0	204.7	9.5	1.3	1.3	1.1
20	230	11 025	420	1 346.0	361.3	940.5	40.4	3.1	3.0	2.4
24	324	21 050	600	**	**	8 775.5	238.4	15.1	6.6	5.1
29	464	41 760	870	**	**	**	**	17.1	16.9	14.3
35	665	83 475	1260	**	**	**	**	67.6	57.1	37.4
42	945	164 948	1806	**	**	**	**	133.7	129.2	92.8

**: The problem could not be solved due to memory limitations.

Nuclear Receptor Signalling

- Tested our algorithm on a model of nuclear receptor signalling.
- The model has 37 states, and a cubic vector field
- Tested local stability in the ball with radius 0.01
- Constructing a quadratic Lyapunov function

✓ Dimension of the resulting SDP:

✓ Cone size: K.s = [37 37 741], K.f = 1406; Number of constraints: 102 676, the size of A: 102676×553225, (553225 is the number of variables)

Interior point solvers				First-order solvers	
SeDuMi	SDPT3	SDPA	CSDP	SCS	CDCS
Failed	Failed	Failed	Failed	151 s	57.3 s

Failed: Out of memory

Numerical results on a PC with 2.8 GHz Intel Core i7 and 8 GB of RAM

OUTLINE

Conclusions

- SOS programs have found in a wide range of applications, e.g. polynomial optimization problems and nonlinear systems analysis questions.
- One fundamental challenge is the poor scalability of SOS programs to large instances
- Fortunately, the resulting SDPs are highly sparse and structured, (partial orthogonality in this talk)
- We have developed a fast ADMM algorithm to solve this class of SDPs at the cost of reduced accuracy.

- CDCS: Download from https://github.com/OxfordControl/CDCS

Thank you for your attention!
Q & A

1. Zheng, Y., Fantuzzi, G., & Papachristodoulou, A. (2017). Fast ADMM for sum-of-squares programs using partial orthogonality. *arXiv preprint arXiv:1708.04174*.
2. Zheng, Yang, Giovanni Fantuzzi, and Antonis Papachristodoulou. "Exploiting Sparsity in the Coefficient Matching Conditions in Sum-of-Squares Programming using ADMM." *IEEE Control Systems Letters* (2017).

Paper ThB03.3