Polyhedral Homotopy Methods vs Semidefinite Programming Relaxations for Problems Involving Polynomials

Spring 2007 Workshop
Geometry of Mechanism Science

University of Notre Dame, March 1-4, 2007

Masakazu Kojima

Tokyo Institute of Technology

Contents

1. PHoMpara — Parallel implementation of polyhedral homotopy method ([1] Gunji-Kim-Fujisawa-Kojima ’06)
2. SparsePOP — Matlab implementation of SDP relaxation for sparse POPs ([2] Waki-Kim-Kojima-Muramatsu ’05)
3. Numerical comparison between the polyhedral homotopy method and the SDP relaxation
([1]+[2]+[3] Mevissen-Kojima-Nie-Takayama)
4. Concluding remarks

SDP $=$ Semidefinite Program or Programming
POP $=$ Polynomial Optimization Problem

Contents

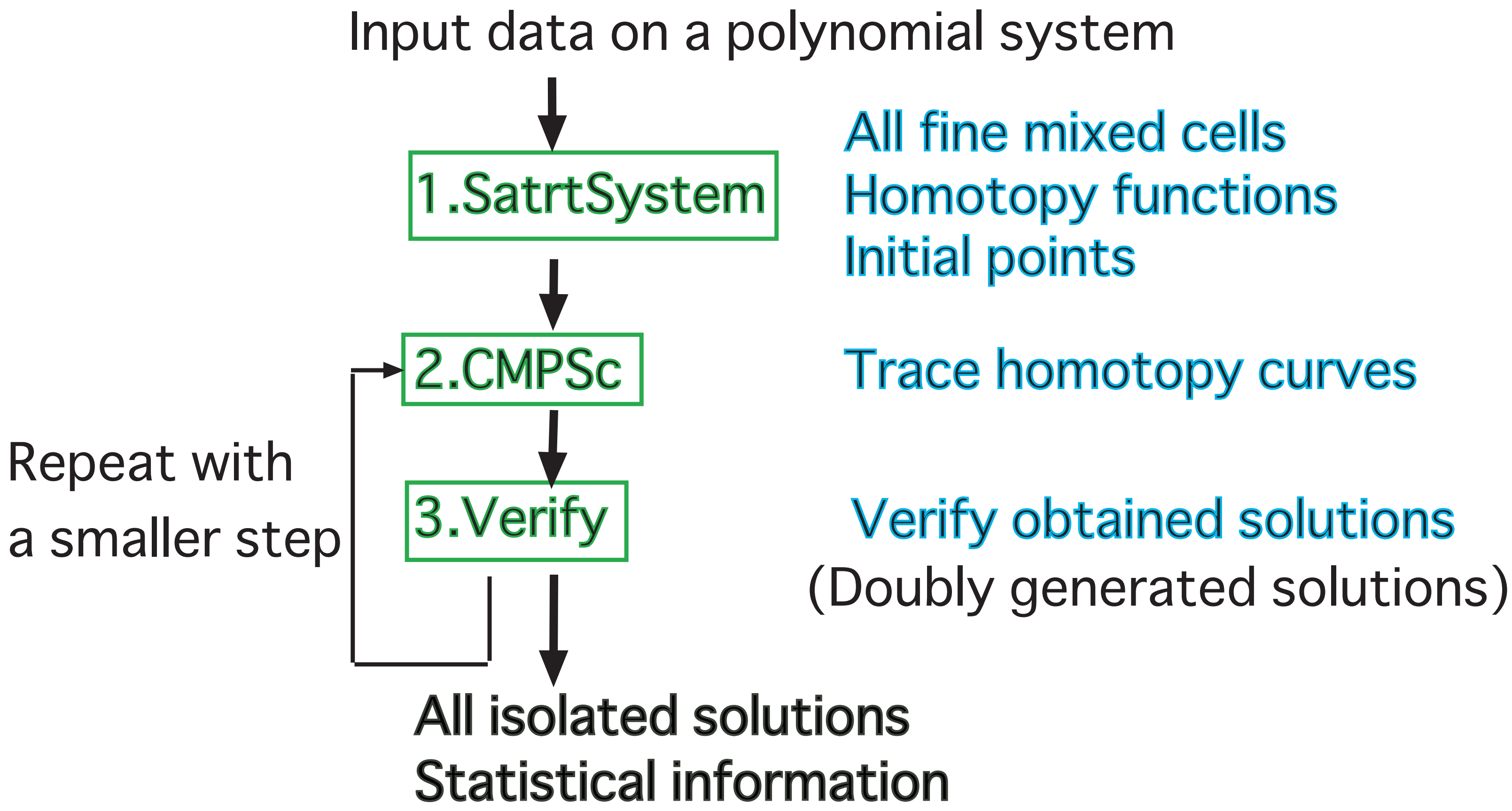
1. PHoMpara — Parallel implementation of the polyhedral homotopy method ([1] Gunji-Kim-Fujisawa-Kojima ’06)
2. SparsePOP — Matlab implementation of SDP relaxation for sparse POPs ([2] Waki-Kim-Kojima-Muramatsu ’05)
3. Numerical comparison between the polyhedral homotopy method and the SDP relaxation
   ([1]+[2]+[3] Mevissen-Kojima-Nie-Takayama)
4. Concluding remarks

SDP $=$ Semidefinite Program or Programming
POP $=$ Polynomial Optimization Problem

The polyhedral homotopy method

- Implementation on a single CPU:
  - PHCpack [Verschelde]
  - HOM4PS [Li-Li-Gao]
  - PHoM [Gunji-Kim-Kojima-Takeda-Fujisawa-Mizutani]
- Suitable for parallel computation — all isolated solutions can be computed independently in parallel.
  - PHoMpara [Gunji, Kim, Fujisawa and Kojima] — Next
  - Leykin, Verschelde and Zhuang

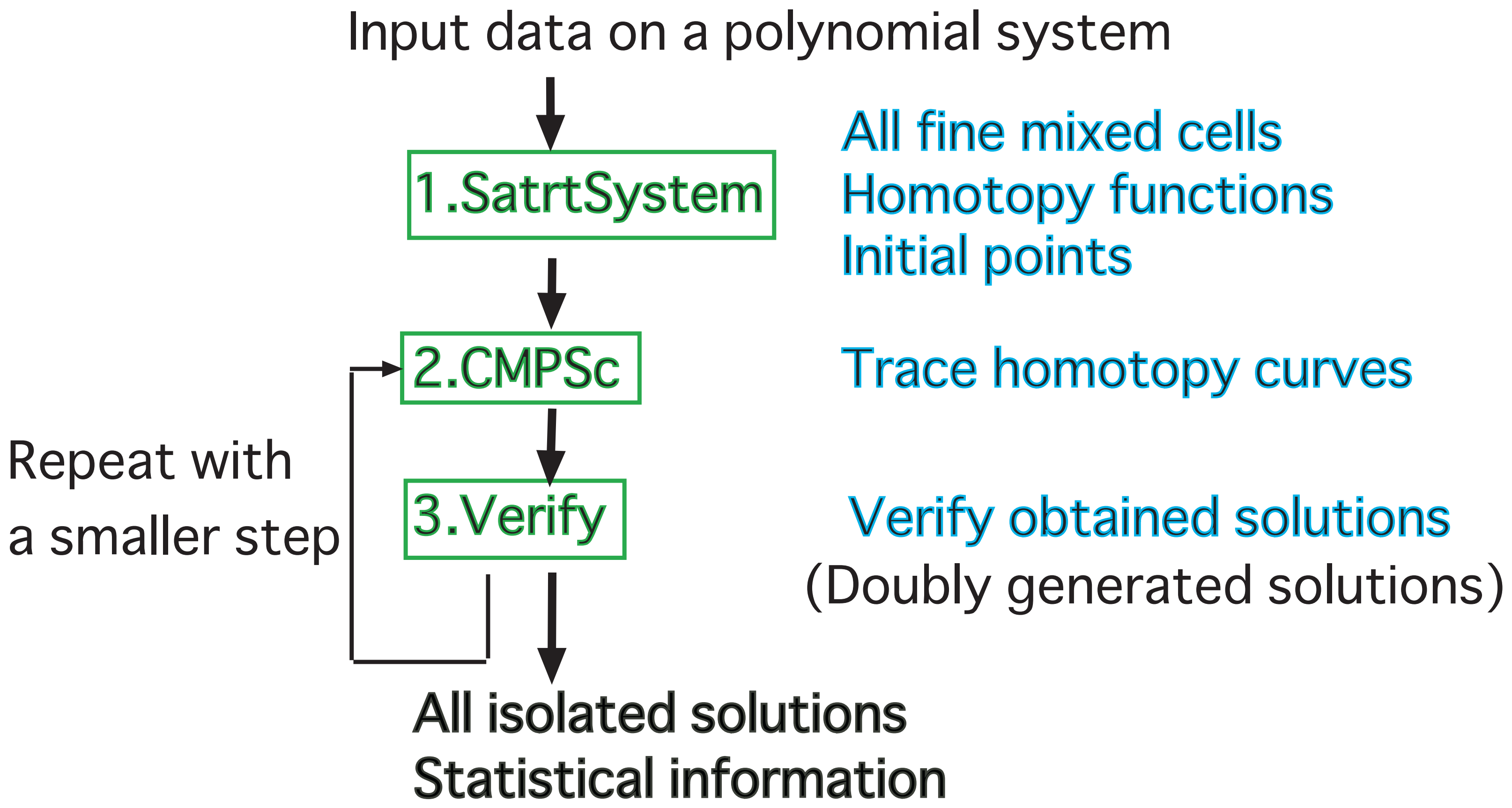
Structure of PHoMpara:



Parallel computation in 1. StartSystem

- Computation of all fine mixed cells
- Balancing powers of the homo. parameter (Li-Verschelde)
  - LP with small # variables & large # ineq. constraints
  - a cutting plane (a column generation simplex) method

Structure of PHoMpara:



Parallel computation in 2. CMPSc

- Each homotopy curve can be traced by pred.corr. meth. independently
  — easy to execute in parallel; divide the h.curves to be traced into ($10\times$ #workers) sets with almost equal size, and distribute each set to each worker.

Numerical results: Hardware — PC cluster (AMD Athlon 2.0GHz)

Problem (#sol)	#CPUs	cpu time in second StartSy	CMPSc	Total	speedup ratio
eco-14	1	13,620	9,033	22,653	1.0
(4,096)	40	388	238	626	36.2
noon-10	1	66	62,606	62,672	1.0
(59,029)	40	27	1,770	1,797	34.9
eco-16 (16,384)	40	10,470	1,581	12,051	
noon-12 (531,417)	40	78	49,380	49,458	

StartSy — mixed vol., homotopy functions, init. points
CMPSc — tracing homotopy curves + Verify

Contents

1. PHoMpara — Parallel implementation of the polyhedral homotopy method ([1] Gunji-Kim-Fujisawa-Kojima '06)
2. SparsePOP — Matlab implementation of SDP relaxation for sparse POPs ([2] Waki-Kim-Kojima-Muramatsu '05)
3. Numerical comparison between the polyhedral homotopy method and the SDP relaxation
([1]+[2]+[3] Mevissen-Kojima-Nie-Takayama)
4. Concluding remarks

SDP $=$ Semidefinite Program or Programming
POP $=$ Polynomial Optimization Problem

SparsePOP (Waki-Kim-Kojima-Muramatsu '06) = Lasserre's SDP relaxation '01 + "structured sparsity" — c-sparsity

POP min. $f_0(\boldsymbol{x})$ s.t. $f_j(\boldsymbol{x}) \geq 0$ or $= 0$ $(j = 1, \ldots, m)$.

Example: $f_0(\boldsymbol{x}) = \sum_{k=1}^{n} (-x_k^2)$
$f_j(\boldsymbol{x}) = 1 - x_j^2 - 2x_{j+1}^2 - x_n^2 \ (j = 1, \ldots, n-1)$.

$\boldsymbol{H} f_0(\boldsymbol{x})$: the $n \times n$ Hes. mat. of $f_0(\boldsymbol{x})$,

$\boldsymbol{J}\boldsymbol{f}_*(\boldsymbol{x})$: the $m \times n$ Jacob. mat. of $\boldsymbol{f}_*(\boldsymbol{x}) = (f_1(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x}))^T$,

$\boldsymbol{R}$: the csp matrix, the $n \times n$ sparsity pattern matrix of $\boldsymbol{I} + \boldsymbol{H} f_0(\boldsymbol{x}) + \boldsymbol{J}\boldsymbol{f}_*(\boldsymbol{x})^T \boldsymbol{J}\boldsymbol{f}_*(\boldsymbol{x})$ (no cancellation in '$+$').

$[\boldsymbol{J}\boldsymbol{f}_*(\boldsymbol{x})^T \boldsymbol{J}\boldsymbol{f}_*(\boldsymbol{x})]_{ij} \neq 0$ iff x_i and x_j are in a common constraint.

Example with n = 6:

$$\text{the csp matrix } \boldsymbol{R} = \begin{pmatrix} \star & \star & 0 & 0 & 0 & \star \\ \star & \star & \star & 0 & 0 & \star \\ 0 & \star & \star & \star & 0 & \star \\ 0 & 0 & \star & \star & \star & \star \\ 0 & 0 & 0 & \star & \star & \star \\ \star & \star & \star & \star & \star & \star \end{pmatrix}$$

POP min. $f_0(\boldsymbol{x})$ s.t. $f_j(\boldsymbol{x}) \geq 0$ or $= 0$ $(j = 1, \ldots, m)$.

Example: $f_0(\boldsymbol{x}) = \sum_{k=1}^{n} (-x_k^2)$
$f_j(\boldsymbol{x}) = 1 - x_j^2 - 2x_{j+1}^2 - x_n^2$ $(j = 1, \ldots, n-1)$.

$\boldsymbol{H} f_0(\boldsymbol{x})$: the $n \times n$ Hes. mat. of $f_0(\boldsymbol{x})$,

$\boldsymbol{J} \boldsymbol{f}_*(\boldsymbol{x})$: the $m \times n$ Jacob. mat. of $\boldsymbol{f}_*(\boldsymbol{x}) = (f_1(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x}))^T$,

$\boldsymbol{R}$: the csp matrix, the $n \times n$ sparsity pattern matrix of $\boldsymbol{I} + \boldsymbol{H} f_0(\boldsymbol{x}) + \boldsymbol{J} \boldsymbol{f}_*(\boldsymbol{x})^T \boldsymbol{J} \boldsymbol{f}_*(\boldsymbol{x})$ (no cancellation in '$+$').

$[\boldsymbol{J} \boldsymbol{f}_*(\boldsymbol{x})^T \boldsymbol{J} \boldsymbol{f}_*(\boldsymbol{x})]_{ij} \neq 0$ iff x_i and x_j are in a common constraint.

POP : c-sparse (correlatively sparse) $\Leftrightarrow$ The $n \times n$ csp matrix $\boldsymbol{R} = (R_{ij})$ allows a symbolic sparse Cholesky factorization (under a row & col. ordering like a symmetric min. deg. ordering).

POP min. $f_0(\boldsymbol{x})$ s.t. $f_j(\boldsymbol{x}) \geq 0$ or $= 0$ $(j = 1, \ldots, m)$.

Example: $f_0(\boldsymbol{x}) = \sum_{k=1}^{n} (-x_k^2)$ ——— c-sparse
$f_j(\boldsymbol{x}) = 1 - x_j^2 - 2x_{j+1}^2 - x_n^2 \ (j = 1, \ldots, n-1)$.

$\boldsymbol{H} f_0(\boldsymbol{x})$: the $n \times n$ Hes. mat. of $f_0(\boldsymbol{x})$,

$\boldsymbol{J} \boldsymbol{f}_*(\boldsymbol{x})$: the $m \times n$ Jacob. mat. of $\boldsymbol{f}_*(\boldsymbol{x}) = (f_1(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x}))^T$,

$\boldsymbol{R}$: the csp matrix, the $n \times n$ sparsity pattern matrix of
$\boldsymbol{I} + \boldsymbol{H} f_0(\boldsymbol{x}) + \boldsymbol{J} \boldsymbol{f}_*(\boldsymbol{x})^T \boldsymbol{J} \boldsymbol{f}_*(\boldsymbol{x})$ (no cancellation in '$+$').

$[\boldsymbol{J} \boldsymbol{f}_*(\boldsymbol{x})^T \boldsymbol{J} \boldsymbol{f}_*(\boldsymbol{x})]_{ij} \neq 0$ iff x_i and x_j are in a common constraint.

Example with n = 6:

$$\text{the csp matrix } \boldsymbol{R} = \begin{pmatrix} \star & \star & 0 & 0 & 0 & \star \\ \star & \star & \star & 0 & 0 & \star \\ 0 & \star & \star & \star & 0 & \star \\ 0 & 0 & \star & \star & \star & \star \\ 0 & 0 & 0 & \star & \star & \star \\ \star & \star & \star & \star & \star & \star \end{pmatrix}$$

Sparse (SDP) relaxation = Lasserre (2001) + c-sparsity

POP min. $f_0(\boldsymbol{x})$ s.t. $f_j(\boldsymbol{x}) \geq 0$ or $= 0$ $(j = 1, \ldots, m)$, c-sparse.

$$\Downarrow$$

A sequence of c-sparse SDP relaxation problems depending on the relaxation order $r = 1, 2, \ldots$;

(a) Under a moderate assumption,
opt. sol. of SDP $\rightarrow$ opt sol. of POP as $r \rightarrow \infty$.

(b) $r = \lceil$"the max. deg. of poly. in POP"$/2\rceil + 0 \sim 3$ is usually large enough to attain opt sol. of POP in practice.

(c) Such an r is unknown in theory except $\exists$ special cases.

(d) Additional method for all opt. sol. of POP, but expensive.

(e) The size of SDP increases rapidly as $r \rightarrow \infty$.

$\boxed{\text{POP min. } f_0(\boldsymbol{x}) \text{ s.t. } f_j(\boldsymbol{x}) \geq 0 \text{ or } = 0 \ (j = 1, \ldots, m)\text{, c-sparse.}}$

Two steps to derive a sparse SDP relaxation of POP

(a) Convert POP to an equivalent poly.SDP with the same c-sparsity.

(b) Linearize poly.SDP $\Rightarrow$ SDP with a similar c-sparsity to poly.SDP.

Example of Sparse SDP Relaxation

POP: min $\sum_{i=1}^{4}(-x_i^3)$ s.t. $-a_i \times x_i^2 - x_4^2 + 1 \geq 0 \ (i = 1, 2, 3)$.

the csp matrix $\boldsymbol{R} = \begin{pmatrix} \star & 0 & 0 & \star \\ 0 & \star & 0 & \star \\ 0 & 0 & \star & \star \\ \star & \star & \star & \star \end{pmatrix}$

No fill-in in the Cholesky factorization $\Rightarrow$ c-sparse.

Example of Sparse SDP Relaxation

POP: min $\sum_{i=1}^{4}(-x_i^3)$ s.t. $-a_i \times x_i^2 - x_4^2 + 1 \geq 0 \ (i = 1, 2, 3)$.

$\Updownarrow$ with the relaxation order $r = 2 \geq r_0 = \lceil 3/2 \rceil = 2$

poly.SDP:

min $\sum_{i=1}^{4}(-x_i^3)$

s.t. $(-a_i \times x_i^2 - x_4^2 + 1)(1, x_i, x_4)^T(1, x_i, x_4) \succeq \boldsymbol{O} \quad i = 1, 2, 3,$

$(1, x_j, x_4, x_j^2, x_j x_4, x_4^2)^T(1, x_j, x_4, x_j^2, x_j x_4, x_4^2) \succeq \boldsymbol{O} \ j = 1, 2, 3.$

$$\text{the csp matrix } \boldsymbol{R} = \begin{pmatrix} \star & 0 & 0 & \star \\ 0 & \star & 0 & \star \\ 0 & 0 & \star & \star \\ \star & \star & \star & \star \end{pmatrix}$$

No fill-in in the Cholesky factorization $\Rightarrow$ c-sparse.

Example of Sparse SDP Relaxation

POP: min $\sum_{i=1}^{4}(-x_i^3)$ s.t. $-a_i \times x_i^2 - x_4^2 + 1 \geq 0 \ (i = 1, 2, 3)$.

$\Updownarrow$ with the relaxation order $r = 2 \geq r_0 = \lceil 3/2 \rceil = 2$

poly.SDP:

min $\sum_{i=1}^{4}(-x_i^3)$

s.t. $(-a_i \times x_i^2 - x_4^2 + 1)(1, x_i, x_4)^T(1, x_i, x_4) \succeq \boldsymbol{O} \quad i = 1, 2, 3,$

$(1, x_j, x_4, x_j^2, x_j x_4, x_4^2)^T(1, x_j, x_4, x_j^2, x_j x_4, x_4^2) \succeq \boldsymbol{O} \ j = 1, 2, 3.$

Represent poly.SDP as

min $\sum_{\boldsymbol{\alpha} \in \mathcal{A}_0} g_0(\boldsymbol{\alpha})\boldsymbol{x}^{\boldsymbol{\alpha}}$ s.t. $\sum_{\boldsymbol{\alpha} \in \mathcal{A}_j} \boldsymbol{G}_j(\boldsymbol{\alpha})\boldsymbol{x}^{\boldsymbol{\alpha}} \succeq \boldsymbol{O} \ j = 1, \ldots, 6,$

where $\mathcal{A}_j \subset \mathbb{Z}_+^4$ and $\boldsymbol{x}^\alpha = x_1^{\alpha_1} x_2^{\alpha_2} x_3^{\alpha_3} x_4^{\alpha_4}$; $\boldsymbol{x}^{(1,2,1,0)} = x_1 x_2^2 x_3$.

$\Downarrow$ Linearize by replacing each $\boldsymbol{x}^{\boldsymbol{\alpha}}$ by an indep. var. $y_{\boldsymbol{\alpha}}$; $\boldsymbol{x}^0$ by 1

SDP min $\displaystyle\sum_{\boldsymbol{\alpha} \in \mathcal{A}_0} g_0(\boldsymbol{\alpha}) y_{\boldsymbol{\alpha}}$ s.t. $\displaystyle\sum_{\boldsymbol{\alpha} \in \mathcal{A}_j} \boldsymbol{G}_j(\boldsymbol{\alpha}) y_{\boldsymbol{\alpha}} \succeq \boldsymbol{O} \ j = 1, \ldots, 6,$

which forms an SDP relaxation of POP.

Contents

1. PHoMpara — Parallel implementation of polyhedral homotopy method ([1] Gunji-Kim-Fujisawa-Kojima ’06)
2. SparsePOP — Matlab implementation of SDP relaxation for sparse POPs ([2] Waki-Kim-Kojima-Muramatsu ’05)
3. Numerical comparison between the polyhedral homotopy method and the SDP relaxation ([1]+[2]+[3] Mevissen-Kojima-Nie-Takayama)
4. Concluding remarks

SDP = Semidefinite Program or Programming
POP = Polynomial Optimization Problem

Unconstrained optimization

Unconstrained POP: min. $f(\boldsymbol{x})$, $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$.

Broyden tridiagonal function with min.val.$= 0$

$f(\boldsymbol{x}) = \sum_{i=1}^{n} \left((3 - 2x_i)x_i - x_{i-1} - 2x_{i+1} + 1\right)^2, \; x_0 = x_{n+1} = 0.$

Generalized Rosenbrock function with min.val.$= 0$

$f(\boldsymbol{x}) = \sum_{i=2}^{n} \left(100(x_i - x_{i-1}^2)^2 + (1 - x_i)^2\right)$.

	B. tridiagonal			G. Rosenbrock		
n	r	approx.opt.val	cpu	r	approx.opt.val	cpu
600	2	1.0e-7	9.3	2	3.9e-7	3.4
800	2	2.2e-7	12.6	2	2.1e-7	5.1
1000	2	2.6e-7	16.0	2	4.5e-7	5.9

A POP alkyl from globalib

$$\begin{aligned}
\text{min} \quad & -6.3x_5x_8 + 5.04x_2 + 0.35x_3 + x_4 + 3.36x_6 \\
\text{sub.to} \quad & -0.820x_2 + x_5 - 0.820x_6 = 0,
\end{aligned}$$

$$0.98x_4 - x_7(0.01x_5x_{10} + x_4) = 0, \; -x_2x_9 + 10x_3 + x_6 = 0,$$

$$x_5x_{12} - x_2(1.12 + 0.132x_9 - 0.0067x_9^2) = 0,$$

$$x_8x_{13} - 0.01x_9(1.098 - 0.038x_9) - 0.325x_7 = 0.574,$$

$$x_{10}x_{14} + 22.2x_{11} = 35.82, \; x_1x_{11} - 3x_8 = -1.33,$$

$$\text{lbd}_i \le x_i \le \text{ubd}_i \; (i = 1, 2, \ldots, 14).$$

- 14 variables, 7 poly. equality constraints with deg. 3.

	Sparse			Dense (Lasserre)		
r	ϵ_{obj}	ϵ_{feas}	cpu	ϵ_{obj}	ϵ_{feas}	cpu
2	1.0e-02	7.1e-01	1.8	7.2e-3	4.3e-2	14.4
3	5.6e-10	2.0e-08	23.0	out of	memory	

ϵ_{obj} = approx.opt.val. – lower bound for opt.val.
ϵ_{feas} = the maximum error in the equality constraints

Systems of polynomial equations

- Is the (sparse) SDP relaxation useful to solve systems of polynomial equations?
- The answer depends on:
  - how sparse the system of polynomial equations is.
  - the maximum degree of polynomials.

- 2 types of systems of polynomial equations

(a) Benchmark test problems from Verschelde's homepage; katsura and cyclic — not c-sparse

(b) System of polynomials arising from discretization of ODEs and DAEs (Differential Algebraic Equations) — c-sparse katsura n system of polynomial equations; $n = 8$ case

$$0 = -x_1 + 2x_9^2 + 2x_8^2 + 2x_7^2 + \cdots + 2x_2^2 + x_1^2,$$

$$0 = -x_2 + 2x_9x_8 + 2x_8x_7 + 2x_7x_6 + \cdots + 2x_3x_2 + 2x_2x_1,$$

$$\cdots\cdots$$

not c-sparse

$$0 = -x_8 + 2x_9x_2 + 2x_8x_1 + 2x_7x_2 + 2x_6x_3 + 2x_5x_4,$$

$$1 = 2x_9 + 2x_8 + 2x_7 + 2x_6 + 2x_5 + 2x_4 + 2x_3 + 2x_2 + x_1.$$

- Numerical results on SparsePOP (WKKM 2004)

n	obj.funct.	r	$\boldsymbol{A}$ in SeDuMi	#nz in $\boldsymbol{A}$	cpu
8	$\sum x_i \uparrow$	1	[54, 217]	280	0.08
8	$\sum x_i^2 \downarrow$	2	[714, 6,730]	11,194	7.1
11	$\sum x_i \uparrow$	1	[90, 361]	473	0.14
11	$\sum x_i^2 \downarrow$	2	[1,819, 17,043]	29,431	101.3

- A formulation in terms of a POP

  max $\sum_{i=1}^n x_i$ or min $\sum_{i=1}^n x_i^2$

  sub.to katsura n system , $-5 \leq x_i \leq 5 \ (i = 1, \ldots, n)$.

  - Different objective functions $\Rightarrow$ different solutions.

katsura n system of polynomial equations; $n = 8$ case

$$0 = -x_1 + 2x_9^2 + 2x_8^2 + 2x_7^2 + \cdots + 2x_2^2 + x_1^2,$$

$$0 = -x_2 + 2x_9x_8 + 2x_8x_7 + 2x_7x_6 + \cdots + 2x_3x_2 + 2x_2x_1,$$

$$\cdots\cdots$$

not c-sparse

$$0 = -x_8 + 2x_9x_2 + 2x_8x_1 + 2x_7x_2 + 2x_6x_3 + 2x_5x_4,$$

$$1 = 2x_9 + 2x_8 + 2x_7 + 2x_6 + 2x_5 + 2x_4 + 2x_3 + 2x_2 + x_1.$$

- Numerical results on SparsePOP (WKKM 2004)

n	obj.funct.	r	$\boldsymbol{A}$ in SeDuMi	#nz in $\boldsymbol{A}$	cpu
8	$\sum x_i \uparrow$	1	[54, 217]	280	0.08
8	$\sum x_i^2 \downarrow$	2	[714, 6,730]	11,194	7.1
11	$\sum x_i \uparrow$	1	[90, 361]	473	0.14
11	$\sum x_i^2 \downarrow$	2	[1,819, 17,043]	29,431	101.3

- Numerical results on HOM4PS (Li-Li-Gao 2002)

n	#solutions	cpu sec.
8	256	1.9
11	2048	209.1

cyclic n system of polynomial equations; $n = 5$ case

$$0 = x_1 + x_2 + x_3 + x_4 + x_5,$$
$$0 = x_1x_2 + x_2x_3 + x_3x_4 + x_4x_5 + x_5x_1,$$

not c-sparse

$$0 = x_1x_2x_3 + x_2x_3x_4 + x_3x_4x_5 + x_4x_5x_1 + x_5x_1x_2,$$
$$0 = x_1x_2x_3x_4 + x_2x_3x_4x_5 + x_3x_4x_5x_1 + x_4x_5x_1x_2 + x_5x_1x_2x_3,$$
$$0 = -1 + x_1x_2x_3x_4x_5.$$

- Numerical results on SparsePOP

n	obj.funct.	r	$\boldsymbol{A}$ in SeDuMi	#nz in $\boldsymbol{A}$	cpu
5	$\sum x_i \uparrow$	3	[431, 7,238]	12,403	1.83
6	$\sum x_i \uparrow$	4	[2,891, 122,007]	198,952	753.2

- Numerical results on HOM4PS

n	#solutions	cpu sec.
5	70	0.1
6	156	0.2

Discretization of Mimura's ODE with 2 unknowns $u,\ v : [0,5] \to \mathbb{R}$

$$\begin{aligned} u_{xx} &= -(\delta_1/9)(35+16u-u^2)u+(\delta_1)(kuv), \\ v_{xx} &= (\delta_2)((1+(2/5)v)v-kuv), \end{aligned}$$

$$u_x(0)=u_x(5)=v_x(0)=v_x(5)=0,$$

where $k=1$, $\delta_1=20$ and $\delta_2=1/4$. Discretize:

$$x_i = i\Delta x\ (i=0,1,2,\dots),\ u_x(x_i) \approx (u(x_{i+1})-u(x_{i-1}))/(2\Delta x).$$

Discretized system of polynomials with $\Delta x = 1$:

$$\begin{aligned} f_1(\boldsymbol{u},\boldsymbol{v}) &= 76.8u_1+u_3+35.6u_1^2-20.0u_1v_1-2.22u_2^3, \\ f_2(\boldsymbol{u},\boldsymbol{v}) &= -1.25v_1+v_2+0.25u_1v_1-0.1v_1^2, \\ f_3(\boldsymbol{u},\boldsymbol{v}) &= u_1+75.8u_2+u_3+35.6u_2^2-20.0u_2v_2-2.22u_2^3, \\ f_4(\boldsymbol{u},\boldsymbol{v}) &= v_1-2.25v_2+v_3+0.25u_2v_2-0.1v_2^2, \\ f_5(\boldsymbol{u},\boldsymbol{v}) &= u_2+75.8u_3+u_4+35.6u_3^2-20.0u_3v_3-2.22u_3^2, \\ f_6(\boldsymbol{u},\boldsymbol{v}) &= v_2-2.25v_3+v_4+0.25u_3v_3-0.1v_3^2, \\ f_7(\boldsymbol{u},\boldsymbol{v}) &= u_3+76.8u_4+35.6u_4^2-20.0u_4v_4-2.22u_4^3, \\ f_8(\boldsymbol{u},\boldsymbol{v}) &= v_3-1.25v_4+0.25u_4v_4-0.1v_4^2. \end{aligned}$$

$\Rightarrow$ c-sparse

Here $u_i = u(x_i)$, $v_i = v(x_i)$ $(i=0,1,2,3,4,5)$,

$u_0-u_1=0$, $u_5-u_4=0$, $v_0-v_1=0$ and $v_5-v_4=0$.

Discretization of Mimura's ODE with 2 unknowns $u,\ v : [0,5] \rightarrow \mathbb{R}$

$$\begin{aligned} u_{xx} &= -(\delta_1/9)(35 + 16u - u^2)u + (\delta_1)(kuv), \\ v_{xx} &= (\delta_2)((1 + (2/5)v)v - kuv), \end{aligned}$$

$$\underline{u_x(0) = u_x(5) = v_x(0) = v_x(5) = 0,}$$

where $k = 1$, $\delta_1 = 20$ and $\delta_2 = 1/4$. Discretize:

$$x_i = i\Delta x\ (i = 0, 1, 2, \ldots),\ u_x(x_i) \approx (u(x_{i+1}) - u(x_{i-1}))/(2\Delta x).$$

- Numerical results on SparsePOP

Δx	n	obj.funct.	r	$\boldsymbol{A}$ in SeDuMi	cpu
1.0	8	$\sum r_i u(x_i) \uparrow$	3	[1,084, 18,732]	11.3
0.5	18	$\sum r_i u(x_i) \uparrow$	3	[3,025, 48,285]	57.8

Here $r_i \in (0,1)$: random numbers.

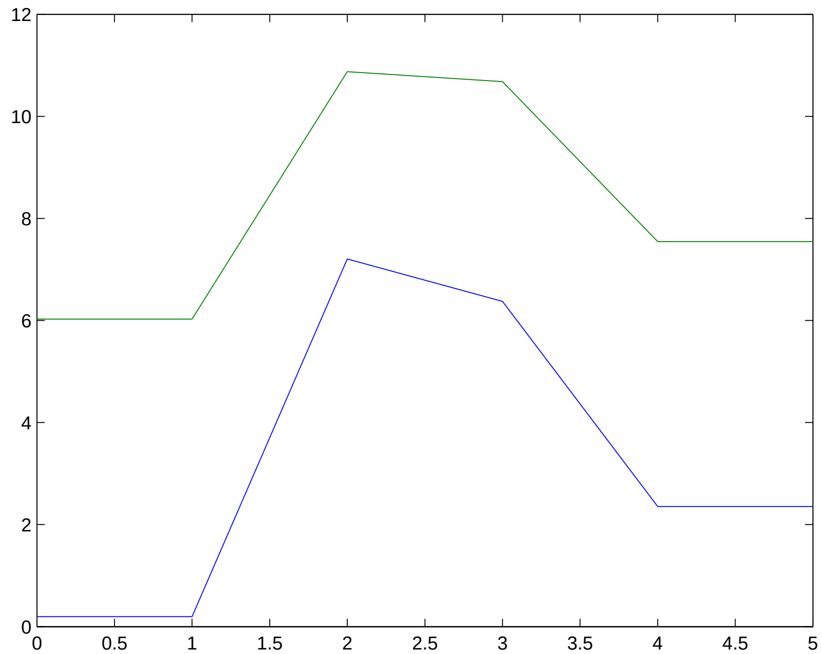
Discretization of Mimura's ODE with 2 unknowns $u,\ v : [0,5] \to \mathbb{R}$

$$
\begin{aligned}
u_{xx} &= -(\delta_1/9)(35 + 16u - u^2)u + (\delta_1)(kuv),\\
v_{xx} &= (\delta_2)((1 + (2/5)v)v - kuv),
\end{aligned}
$$

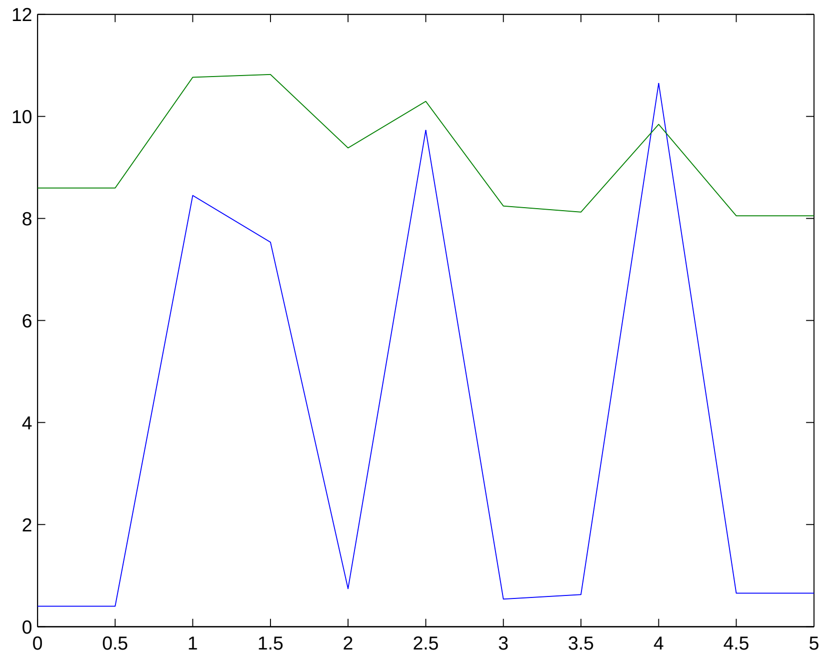
$$\underline{u_x(0) = u_x(5) = v_x(0) = v_x(5) = 0,}$$

where $k = 1$, $\delta_1 = 20$ and $\delta_2 = 1/4$. Discretize:

$$x_i = i\Delta x \ (i = 0, 1, 2, \dots),\ u_x(x_i) \approx (u(x_{i+1}) - u(x_{i-1}))/(2\Delta x).$$



$\Delta x = 1.0$



$\Delta x = 0.5$

Discretization of Mimura's ODE with 2 unknowns $u,\ v : [0,5] \to \mathbb{R}$

$$\begin{aligned} u_{xx} &= -(\delta_1/9)(35+16u-u^2)u+(\delta_1)(kuv), \\ v_{xx} &= (\delta_2)((1+(2/5)v)v-kuv), \end{aligned}$$

$$\underline{u_x(0)=u_x(5)=v_x(0)=v_x(5)=0,}$$

where $k=1$, $\delta_1=20$ and $\delta_2=1/4$. Discretize:

$$x_i = i\Delta x\ (i=0,1,2,\dots),\ u_x(x_i) \approx (u(x_{i+1})-u(x_{i-1}))/(2\Delta x).$$

- Numerical results on SparsePOP

Δx	n	obj.funct.	r	$\boldsymbol{A}$ in SeDuMi	cpu
1.0	8	$\sum r_i u(x_i) \uparrow$	3	[1,084, 18,732]	11.3
0.5	18	$\sum r_i u(x_i) \uparrow$	3	[3,025, 48,285]	57.8

Here $r_i \in (0,1)$: random numbers.

- Numerical results on HOM4PS

Δx	n	#solutions	#real solutions	cpu sec.
1.0	8	1296	222	2.2
0.5	18	10,077,696 (M.vol., 168 sec.)	not traced	

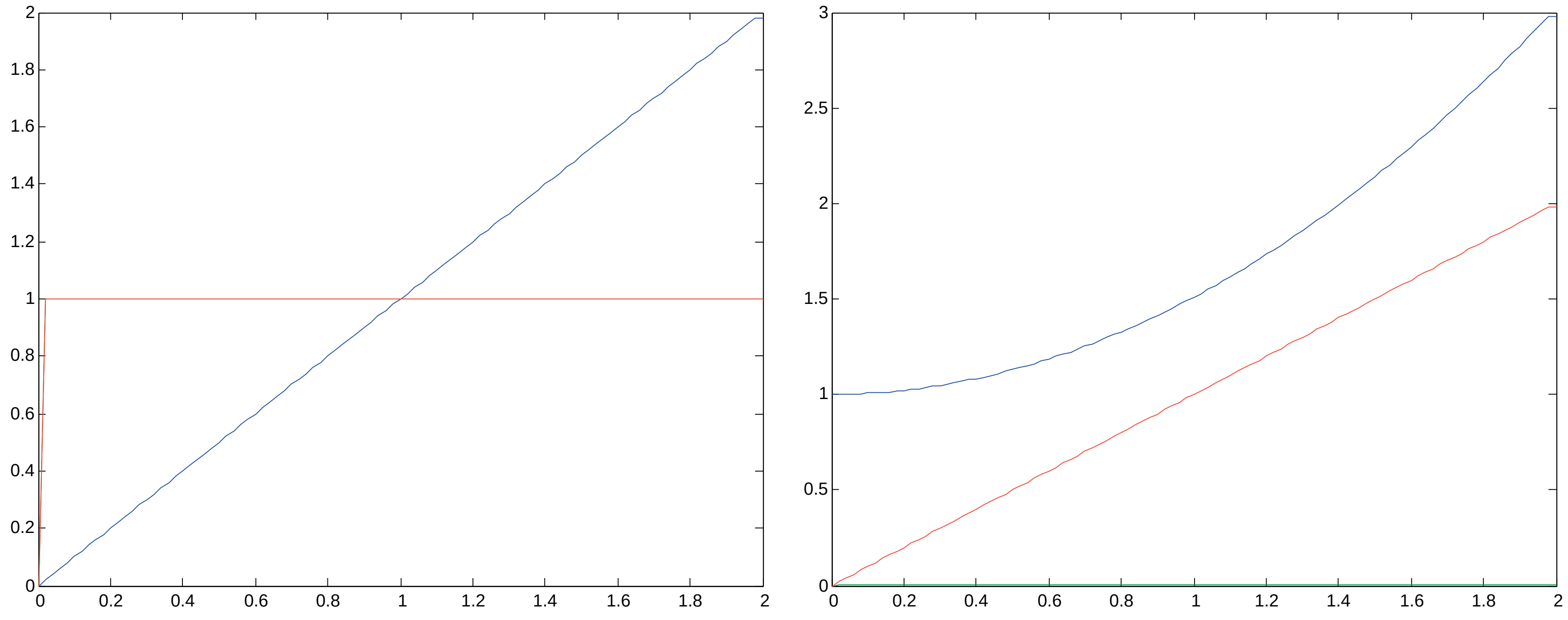
Discretization of DAE with 3 unknowns $y_1, y_2, y_3 : [0,2] \to \mathbb{R}$

$$y_1' = y_3,\ 0 = y_2(1-y_2),\ 0 = y_1 y_2 + y_3(1-y_2) - t,\ y_1(0) = y_1^0.$$

2 solutions : $y(t) = (t,1,1))$ and $y(t) = (y_1^0 + t_2^2, 0, t)$.

- Numerical results on SparsePOP — c-sparse

y_1^0	Δt	n	obj.funct.	r	$\boldsymbol{A}$ in SeDuMi	cpu
0	0.02	297	$\sum y_2(t_i) \uparrow$	2	[3,557, 25,413]	30.9
1	0.02	297	$\sum y_1(t_i) \uparrow$	2	[3,557, 25,413]	33.9



Contents

1. PHoMpara — Parallel implementation of polyhedral homotopy method ([1] Gunji-Kim-Fujisawa-Kojima '06)
2. SparsePOP — Matlab implementation of SDP relaxation for sparse POPs ([2] Waki-Kim-Kojima-Muramatsu '05)
3. Numerical comparison between the polyhedral homotopy method and the SDP relaxation ([1]+[2]+[3] Mevissen-Kojima-Nie-Takayama)
4. Concluding remarks

SDP = Semidefinite Program or Programming
POP = Polynomial Optimization Problem

- Some essential differences between Homotopy Continuation and (sparse) SDP Relaxation — 1:

(a) HC works on $\mathbb{C}^n$ while SDPR on $\mathbb{R}^n$.

(b) HC aims to compute all isolated solutions; in SDPR, computing all isolated solutions is possible but expensive.

(c) SDPR can process inequalities, and SDPR can have an objective function to pick up a specific solution.

- Some essential differences between Homotopy Continuation and (sparse) SDP Relaxation — 2:

(d) SDPR is sensitive to degrees of polynomials of a POP because the SDP relaxed problem becomes larger rapidly as they increase.
$\Rightarrow$ SDPR can be applied to POPs with lower degree polynomials such as degree ≤ 4 in practice.

(e) HC fits parallel computation more than SDPR.

(f) The effectiveness of sparse SDPR depends on the c-sparsity; for example, discretization of ODE, DAE, Optimal control problem and PDE.

Thank you!

Content

1. PHoMpara — Parallel implementation of polyhedral homotopy method ([1] Gunji-Kim-Fujisawa-Kojima ’06)
2. SparsePOP — Matlab implementation of SDP relaxation for sparse POPs ([2] Waki-Kim-Kojima-Muramatsu ’05)
3. Numerical comparison between the polyhedral homotopy method and the SDP relaxation
([1]+[2]+[3] Mevissen-Kojima-Nie-Takayama)
4. Concluding remarks

Appendix.

SOS and SDP relaxations of POPs

$$\boxed{\text{POP:} \quad \min \quad f_0(\boldsymbol{x}) \quad \text{sub.to} \quad f_i(\boldsymbol{x}) \geq 0 \ (i = 1, \ldots, m),}$$

POP $\Rightarrow$ generalized Lagrangian dual

$\Updownarrow$ add valid LMIs — dual — $\Downarrow$

Polynomial SDP — $\Downarrow$ SOS relaxation

$\Downarrow$ linearize (relaxation) — dual — $\Downarrow$

SDP[1] $\Leftrightarrow$ SDP[2]

[1] J.B.Lasserre, "Global optimization with polynomials and the problems of moments", *SIAM J. on Optim.* (2001).

[2] P.A.Parrilo, "Semidefinite programming relaxations for semialgebraic problems". *Math. Prog.* (2003).

(a) Global optimal solutions

(b) Large-scale SDPs require enormous computation

(c) Sparse SDP relaxation (Waki-Kim-Kojima-Muramatsu '06) = SDP[1] + "Exploiting structured sparsity" — c-sparsity