Solving Large Scale Semidefinite Problems by Decomposition

with application to

Topology Optimization with Vibration Constraints

Michal Kočvara

School of Mathematics, The University of Birmingham

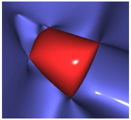
Semidefinite Programming: Theory and Applications
Edinburgh, October 2018

POEMA

Polynomial Optimization, Efficiency through Moments and Algebra

Marie Skłodowska-Curie Innovative Training Network

2019-2022



POEMA network goal is to train scientists at the interplay of algebra, geometry and computer science for polynomial optimization problems and to foster scientific and technological advances, stimulating interdisciplinary and intersectoriality knowledge exchange between algebraists, geometers, computer scientists and industrial actors facing real-life optimization problems.

Partners:

1. Inria, Sophia Antipolis, France (Bernard Mourrain)
2. CNRS, LAAS, Toulouse, France (Didier Henrion)
3. Sorbonne Université, Paris, France (Mohab Safey el Din)
4. NWO-I/CWI, Amsterdam, the Netherlands (Monique Laurent)
5. Univ. Tilburg, the Netherlands (Etienne de Klerk)
6. Univ. Konstanz, Germany (Markus Schweighofer)
7. Univ. degli Studi di Firenze, Italy (Giorgio Ottaviani)
8. Univ. of Birmingham, UK (Mikal Kočvara)
9. F.A. Univ. Erlangen-Nuremberg, Germany (Michael Stingl)
10. Univ. of Tromsoe, Norway (Cordian Riener)
11. Artelys SA, Paris, France (Arnaud Renaud)

Associate partners:

1. IBM Research, Ireland (Martin Mevissen)
2. NAG, UK (Mike Dewar)
3. RTE, France (Jean Maeght)

15 PhD positions available from Sep. 1st 2019

Contact: bernard.mourrain@inria.fr, the partner leaders,
www-sop.inria.fr/members/Bernard.Mourrain/announces/POEMA/

PENNON collection

PENNON (PENalty methods for NONlinear optimization)
a collection of codes for NLP, (linear) SDP and BMI

– one algorithm to rule them all –

C++

- PENNLP AMPL, MATLAB, C/Fortran
- PENSDP MATLAB/YALMIP, SDPA, C/Fortran (FREE)
- PENBMI MATLAB/YALMIP, C/Fortran
- PENNON (NLP + SDP) extended AMPL, MATLAB, C/FORTRAN

MATLAB

- PENLAB (NLP + SDP) open source MATLAB implementation

The NLP-SDP problem

Optimization problems with nonlinear objective subject to nonlinear inequality and equality constraints and semidefinite bound constraints:

$$\begin{aligned}
\min_{x\in\mathbb{R}^n, Y_1\in\mathbb{S}^{p_1},\dots,Y_k\in\mathbb{S}^{p_k}} \quad & f(x,Y) \\
\text{subject to} \quad & g_i(x,Y) \le 0, && i=1,\dots,m_g \\
& h_i(x,Y) = 0, && i=1,\dots,m_h \qquad \text{(NLP-SDP)} \\
& \underline{\lambda}_i I \preceq Y_i \preceq \overline{\lambda}_i I, && i=1,\dots,k
\end{aligned}$$

Notation:

$A \succeq 0$ means A positive semidefinite (all eigenvalues ≥ 0)

$A \succeq B$ means $A - B \succeq 0$

Dimensions in (linear) Semidefinite Optimization

$$\min_{x \in \mathbb{R}^n} c^\top x$$

subject to

$$\sum_{i=1}^{n} x_i A_i^{(k)} - B^{(k)} \succeq 0\,, \quad k = 1, \dots, p$$

where

$$x \in \mathbb{R}^n, \quad A_i^{(k)},\ B^{(k)} \in \mathbb{R}^{m \times m}$$

Majority of SDP software

BAD ... n large, m large   many variables, big matrix

OK ... n small, m large   rare

GOOD ... n large, m small   many variables, small matrix

GOOD ... n large, m small, p large   many small matrix constraints

Solving (very) large scale SDP?

Given the known restrictions of interior point solvers, how can we solve very large scale SDP problems?

- Use iterative solvers
  SDPT3, PENSDP, Jacek Gondzio's recent work
- Use a different algorithm
  Bundle algorithm (Helmberg), Burer-Monteiro SDPA, ADMM (Wolkowicz), Augmented Lagrangian (Rendl, Malick, Toh-Sun,. . .)
- Reformulate **BAD** problems as **GOOD** problems

PENSDP with an iterative solver

$$\min_{x \in \mathbb{R}^n} c^\top x \quad \text{s.t.} \quad \sum_{i=1}^{n} x_i A_i - B \succeq 0\,, \quad A_i, B \in \mathbb{R}^{m \times m}$$

Problems with large n, small m (Kim Toh)
We have to solve repeatedly a dense $n \times n$ linear system.

problem	n	m	direct CPU	iterative CPU	iterative CG/it
ham_8_3_4	16129	256	17701	30	1
ham_9_5_6	53761	512	mem	330	1
theta10	12470	500	12165	227	10
theta104	87845	500	mem	11953	25
theta12	17979	600	27565	254	8
theta123	90020	600	mem	10538	23
theta162	127600	800	mem	13197	13
sanr200-0.7	6 033	200	1146	30	12

mem. . . insufficient memory

PENSDP with hybrid strategy

Use PCG till it works, then switch to Cholesky and return to PCG, using the Ch-factor as a preconditioner.

Collection of chemical problems by M. Fukuda . . .

Average Dimacs error $\approx 1.0e - 7$

problem	n	Cg-it	Chol-it	Nwt-it	CPU-hy	CPU-ch
NH2-.r14	1,743	921	4	69	526	4033
NH3+.r16	2,964	1529	3	72	2427	26634
NH4+.r18	4,239	1607	3	77	8931	$>$ 100000
AlH.r20	7,230	2283	2	102	21720	???

Solving (very) large scale SDP?

Given the known restrictions of interior point solvers, how can we solve very large scale SDP problems?

- Use iterative solvers
  SDPT3, PENSDP, Jacek Gondzio's recent work
- Use a different algorithm
  Bundle algorithm (Helmberg), Burer-Monteiro SDPA, ADMM (Wolkowicz), Augmented Lagrangian (Rendl, Malick, Toh-Sun,...)
- Reformulate **BAD** problems as **GOOD** problems

Solving (very) large scale SDP?

Given the known restrictions of interior point solvers, how can we solve very large scale SDP problems?

- Use iterative solvers
  SDPT3, PENSDP, Jacek Gondzio's recent work
- Use a different algorithm
  Bundle algorithm (Helmberg), Burer-Monteiro SDPA, ADMM (Wolkowicz), Augmented Lagrangian (Rendl, Malick, Toh-Sun,...)
- Reformulate **BAD** problems as **GOOD** problems

Dimensions in (linear) Semidefinite Optimization

$$\min_{x \in \mathbb{R}^n} c^\top x$$

subject to

$$\sum_{i=1}^{n} x_i A_i^{(k)} - B^{(k)} \succeq 0\,, \quad k = 1, \ldots, p$$

where

$$x \in \mathbb{R}^n, \quad A_i^{(k)},\ B^{(k)} \in \mathbb{R}^{m \times m}$$

So we may want to replace

BAD . . . n large, m large, p=1

by

GOOD . . . n large, m small, p large   many small matrix constraints

Chordal decomposition

S. Kim, M. Kojima, M. Mevissen and M. Yamashita, Exploiting Sparsity in Linear and Nonlinear Matrix Inequalities via Positive Semidefinite Matrix Completion, Mathematical Programming, 2011

Based on:

A. Griewank and Ph. Toint, On the existence of convex decompositions of partially separable functions, MPA 28, 1984

J. Agler, W. Helton, S. McCulough and L. Rodnan, Positive semidefinite matrices with a given sparsity pattern, LAA 107, 1988

See also:

L. Vandenberghe and M. Andersen, Chordal graphs and semidefinite optimization. Foundations and Trends in Optimization 1:241–433, 2015

Chordal decomposition

$G(N, E)$ – graph with $N = \{1, \dots, n\}$ and max. cliques $C_1, \dots, C_p$.

$$\mathbb{S}^n(E) = \{Y \in \mathbb{S}^n : Y_{ij} = 0 \; (i,j) \notin E \cup \{(\ell,\ell), \; \ell \in N\}$$

$$\mathbb{S}_+^{C_k} = \{Y \succeq 0 : Y_{ij} = 0 \text{ if } (i,j) \notin C_k \times C_k\}$$

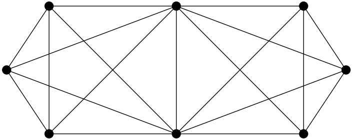
> Theorem 1: $G(N, E)$ is chordal if and only if
> for every $A \in \mathbb{S}^n(E)$, $A \succeq 0$, it holds that
> $\exists Y^k \in \mathbb{S}_+^{C_k}$ $(k = 1, \dots, p)$ s.t. $A = Y^1 + Y^2 + \dots + Y^p$.

Every psd matrix is a sum of psd matrices that are non-zero only on maximal cliques.

So constraint $A(x) \succeq 0$ replaced by:
find matrices $Y^k(x) \succeq 0$, $k = 1, \dots, p$ that sum up to A.

Graph representation of matrix sparsity

Chordal sparsity graph, overlapping blocks



$$\begin{bmatrix} \times & \times & \times & \times & \times & & & \\ \times & \times & \times & \times & \times & & 0 & \\ \times & \times & \times & \times & \times & & & \\ \times & \times & \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times & \times & \times \\ & & & \times & \times & \times & \times & \times \\ & 0 & & \times & \times & \times & \times & \times \\ & & & \times & \times & \times & \times & \times \end{bmatrix}$$

Chordal decomposition

Theorem 1: $G(N,E)$ is chordal if and only if
for every $A \in \mathbb{S}^n(E)$, $A \succeq 0$, it holds that
$\exists Y^k \in \mathbb{S}_+^{C_k}$ $(k = 1,\ldots,p)$ s.t. $A = Y^1 + Y^2 + \ldots + Y^p$.

Let $K = \begin{pmatrix} K_{1,1}^{(1)} & K_{1,2}^{(1)} & 0 \\ K_{2,1}^{(1)} & \boxed{K_{2,2}^{(1)} + K_{1,1}^{(2)}} & K_{1,2}^{(2)} \\ 0 & K_{2,1}^{(2)} & K_{2,2}^{(2)} \end{pmatrix}$ with $K^{(1)}, K^{(2)}$ dense.

Then $K \succeq 0 \Leftrightarrow K = Y^1 + Y^2$ such that

$$Y^1 = \begin{pmatrix} K_{1,1}^{(1)} & K_{1,2}^{(1)} & 0 \\ K_{2,1}^{(1)} & K_{2,2}^{(1)} + S & 0 \\ 0 & 0 & 0 \end{pmatrix} \succcurlyeq 0, \quad Y^2 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & K_{2,2}^{(2)} - S & K_{1,2}^{(2)} \\ 0 & K_{2,1}^{(2)} & K_{2,2}^{(2)} \end{pmatrix} \succcurlyeq 0$$

Even if $K^{(1)}, K^{(2)}$ not dense, we just assume that S is dense.

Chordal decomposition

Let $A \in \mathbb{S}^n$, $n \geq 3$, with a sparsity graph $G = (N, E)$.
Let $N = \{1, 2, \ldots, n\}$ be partitioned into $p \geq 2$ overlapping sets

$$N = I_1 \cup I_2 \cup \ldots \cup I_p \,.$$

Define $I_{k,k+1} = I_k \cap I_{k+1} \neq \emptyset\,, \quad k = 1, \ldots, p-1\,.$

Assume $A = \sum_{k=1}^{p} A_k$, with A_k only non-zero on I_k.

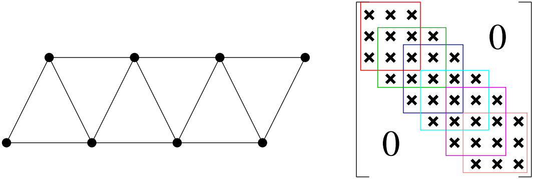
Corollary 1: $A \succeq 0$ if and only if
$\exists S_k \in \mathbb{S}^{I_{k,k+1}}, k = 1, \ldots, p-1$ s.t.

$$A = \sum_{k=1}^{p} \widetilde{A}_k \text{ with } \widetilde{A}_k = A_k - S_{k-1} + S_k \quad (S_0 = S_p = [\,])$$

and $\widetilde{A}_k \succeq 0$ $(k = 1, \ldots, p)$.

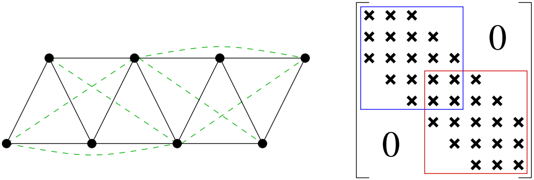
We can <u>choose</u> the partitioning $N = I_1 \cup I_2 \cup \ldots \cup I_p$!

Using the original theorem:



6 max. cliques of size 3, 5 additional 2×2 variables

Using the corollary:



2 “cliques” of size 5, 1 additional 2×2 variable

We can choose the partitioning $N = I_1 \cup I_2 \cup \ldots \cup I_p$!

When we know the sparsity structure of A, we can choose a "regular" partitioning.

Application: Topology optimization

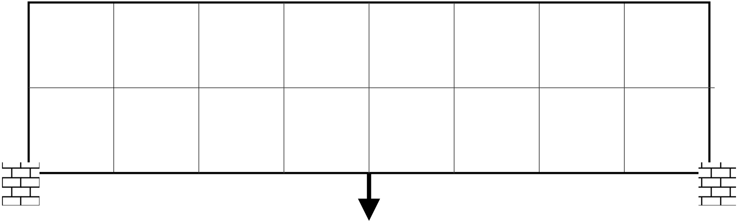
Aim:

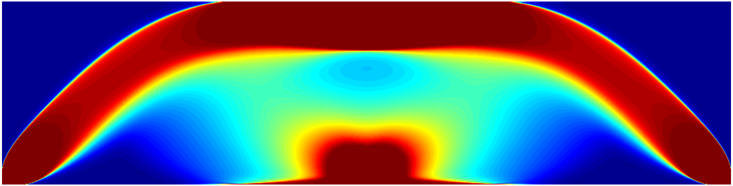
Given an amount of material, boundary conditions and external load f, find the material distribution so that the body is as stiff as possible under f.

$E(x) = \rho(x)E_0$ with $0 \leq \underline{\rho} \leq \rho(x) \leq \overline{\rho}$

E_0 a given (homogeneous, isotropic) material

Topology optimization, example





Pixels—finite elements

Color—value of variable ρ, constant on every element

Equilibrium

Equilibrium equation:

$$K(\rho)u = f, \qquad K(\rho) = \sum_{i=1}^{m} \rho_i K_i := \sum_{i=1}^{m} \sum_{j=1}^{G} B_{i,j} \rho_i E_0 B_{i,j}^\top$$

$$f := \sum_{i=1}^{m} f_i$$

Standard finite element discretization:

Quadrilateral elements

$\rho \ldots$ piece-wise constant

$u \ldots$ piece-wise bilinear (tri-linear)

TO primal formulation

$$\min_{\rho \in \mathbb{R}^m,\, u \in \mathbb{R}^n} f^T u$$

subject to

$$(0 \leq)\ \underline{\rho} \leq \rho_i \leq \overline{\rho}, \quad i = 1, \dots, m$$

$$\sum_{i=1}^{m} \rho_i \leq 1$$

$$K(\rho) u = f$$

... large-scale nonlinear non-convex problem

SDP formulation of TO

The TO problem

$$\min_{\rho\in\mathbb{R}^m,\ u\in\mathbb{R}^n,\ \gamma\in\mathbb{R}} \gamma$$

subject to

$$f^T u \le \gamma, \quad K(\rho)u = f$$

$$\sum \rho_i \le 1, \quad \underline{\rho} \le \rho_i \le \overline{\rho}, \quad i = 1,\dots,m$$

can be equivalently formulated as a linear SDP:

$$\min_{\rho\in\mathbb{R}^m,\ \gamma\in\mathbb{R}} \gamma$$

subject to

$$\begin{pmatrix} \gamma & f^T \\ f & K(\rho) \end{pmatrix} \succeq 0 \qquad \text{(positive semidefinite)}$$

$$\sum \rho_i \le 1, \quad \underline{\rho} \le \rho_i \le \overline{\rho}, \quad i = 1,\dots,m.$$

Helpful when vibration/buckling constraints present

SDP formulation of TO

The TO problem

$$\min_{\rho\in\mathbb{R}^m,\, u\in\mathbb{R}^n,\, \gamma\in\mathbb{R}} \gamma$$

subject to

$$f^T u \leq \gamma, \quad K(\rho)u = f$$

$$\sum \rho_i \leq 1, \quad \underline{\rho} \leq \rho_i \leq \overline{\rho}, \quad i = 1,\dots,m$$

can be equivalently formulated as a linear SDP:

$$\min_{\rho\in\mathbb{R}^m,\, \gamma\in\mathbb{R}} \gamma$$

subject to

$$\begin{pmatrix} \gamma & f^T \\ f & K(\rho) \end{pmatrix} \succeq 0 \qquad \text{(positive semidefinite)}$$

$$\sum \rho_i \leq 1, \quad \underline{\rho} \leq \rho_i \leq \overline{\rho}, \quad i = 1,\dots,m\,.$$

Helpful when vibration/buckling constraints present

SDP formulation of TO

The TO problem

$$\min_{\rho\in\mathbb{R}^m,\, u\in\mathbb{R}^n,\, \gamma\in\mathbb{R}} \gamma$$

subject to

$$f^T u \le \gamma, \quad K(\rho)u = f$$

$$\sum \rho_i \le 1, \quad \underline{\rho} \le \rho_i \le \overline{\rho}, \quad i = 1,\ldots,m$$

can be equivalently formulated as a linear SDP:

$$\min_{\rho\in\mathbb{R}^m,\, \gamma\in\mathbb{R}} \gamma$$

subject to

$$\begin{pmatrix} \gamma & f^T \\ f & K(\rho) \end{pmatrix} \succeq 0 \qquad \text{(positive semidefinite)}$$

$$\sum \rho_i \le 1, \quad \underline{\rho} \le \rho_i \le \overline{\rho}, \quad i = 1,\ldots,m\,.$$

Helpful when vibration/buckling constraints present

TO with a vibration constraint

Self-vibrations of the (discretized) structure—eigenvalues of

$$K(\rho)w = \lambda\, M(\rho)w$$

where the mass matrix $M(\rho)$ has the same sparsity as $K(\rho)$.

Low frequencies dangerous $\rightarrow$ constraint $\lambda_{\min} \geq \hat{\lambda}$

Equivalently: $V(\hat{\lambda};\rho) := K(\rho) - \hat{\lambda}M(\rho) \succeq 0$

TO problem with vibration constraint as linear SDP:

$$\min_{\rho\in\mathbb{R}^m,\ \gamma\in\mathbb{R}} \gamma$$

subject to

$$\begin{pmatrix} \gamma & f^T \\ f & K(\rho) \end{pmatrix} \succeq 0$$

$$V(\hat{\lambda};\rho) \succeq 0$$

$$\sum \rho_i \leq 1, \quad \underline{\rho} \leq \rho_i \leq \overline{\rho}, \quad i = 1,\dots,m$$

SDP formulation of TO by decomposition

Both

$$\begin{pmatrix} \gamma & f^T \\ f & \sum \rho_i K_i \end{pmatrix} \succeq 0$$

and

$$V(\hat{\lambda}; \rho) \succeq 0$$

are large matrix constraints dependent on many variables
... bad for existing SDP software

Can we replace them by several smaller constraints equivalently?

Chordal decomposition (recall)

Let $A \in \mathbb{S}^n$, $n \geq 3$, with a sparsity graph $G = (N, E)$.
Let $N = \{1, 2, \ldots, n\}$ be partitioned into $p \geq 2$ overlapping sets

$$N = I_1 \cup I_2 \cup \ldots \cup I_p \,.$$

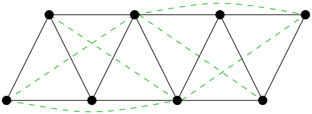
Define $I_{k,k+1} = I_k \cap I_{k+1} \neq \emptyset \,, \quad k = 1, \ldots, p-1 \,.$

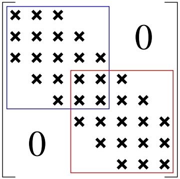
Assume $A = \sum_{k=1}^{p} A_k$, with A_k only non-zero on I_k.

Corollary 1: $A \succeq 0$ if and only if
$\exists S_k \in \mathbb{S}^{I_{k,k+1}}, k = 1, \ldots, p-1$ s.t.
$A = \sum_{k=1}^{p} \widetilde{A}_k$ with $\widetilde{A}_k = A_k - S_{k-1} + S_k \quad (S_0 = S_p = [\,])$
and $\widetilde{A}_k \succeq 0$ $(k = 1, \ldots, p)$.

We can <u>choose</u> the partitioning $N = I_1 \cup I_2 \cup \ldots \cup I_p$!

Using the corollary:





2 "cliques" of size 5, 1 additional 2×2 variable

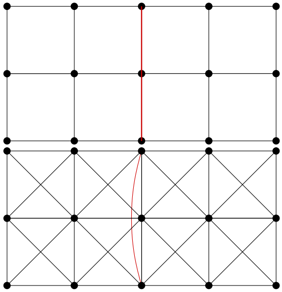
When we know the sparsity structure of A, we can choose a regular partitioning.

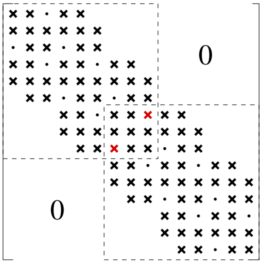
SDP formulation of TO by DD

$$\begin{pmatrix} K(\rho) & f \\ f^\top & \gamma \end{pmatrix} \succeq 0 \qquad \text{and} \qquad V(\hat{\lambda};\rho) \succeq 0$$

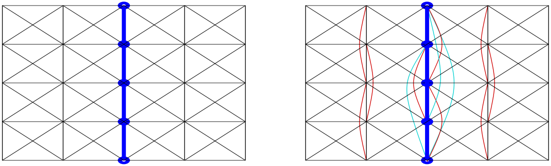
are large matrix constraints dependent on many variables.

FE mesh, matrix $K(\rho)$ and its sparsity graph:





Chordal decomposition



$$\begin{pmatrix} K_{II}^{(1)} & K_{I\Gamma}^{(1)} & 0 & 0 \\ K_{\Gamma I}^{(1)} & K_{\Gamma\Gamma}^{(1)}+K_{\Gamma\Gamma}^{(2)} & K_{\Gamma I}^{(2)} & 0 \\ 0 & K_{I\Gamma}^{(2)} & K_{II}^{(2)} & f \\ 0 & 0 & f^\top & \gamma \end{pmatrix} = \begin{pmatrix} K_{II}^{(1)} & K_{I\Gamma}^{(1)} & 0 & 0 \\ K_{\Gamma I}^{(1)} & K_{\Gamma\Gamma}^{(1)}+S & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & K_{\Gamma\Gamma}^{(2)}-S & K_{\Gamma I}^{(2)} & 0 \\ 0 & K_{I\Gamma}^{(2)} & K_{II}^{(2)} & f \\ 0 & 0 & f^\top & \gamma \end{pmatrix}$$

Even though $K^{(1)}$ and $K^{(2)}$ are sparse, we need to assume that S is dense.

In this way, we can control the number and size of the maximal cliques and use the chordal decomposition theorem.

New result for arrow-type matrices: For the matrix inequality

$$\begin{pmatrix} K(\rho) & f \\ f^\top & \gamma \end{pmatrix} \succeq 0$$

the additional matrix variables S are rank-one; this further reduces the size of the solved SDP problem.

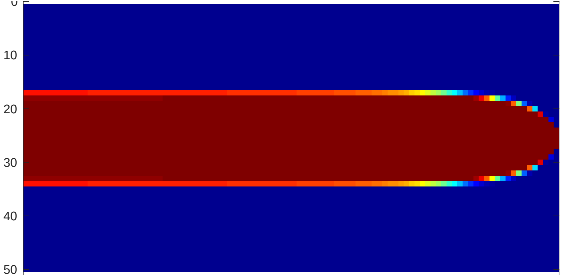
Numerical experiments

SDP codes tested: PENSDP, SeDuMi, SDPT3, Mosek

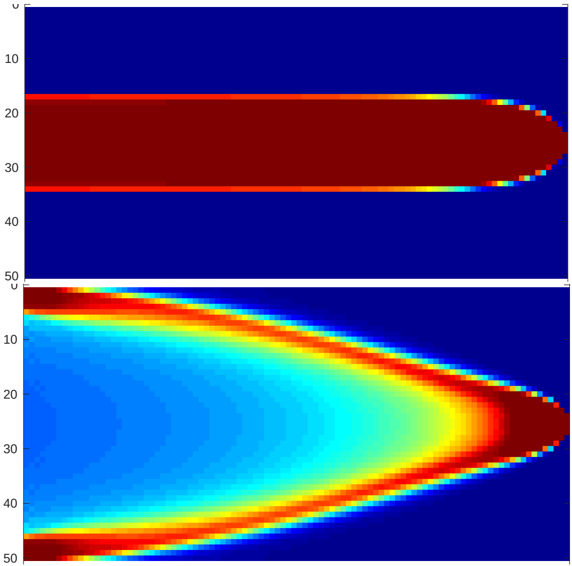
Results shown for Mosek: not the fastest for the original problem but has highest speedup

Mosek:
– version 8 much more reliable than version 7
– called from YALMIP
– difficult (for me) to control any options

Numerical experiments



Numerical experiments



Numerical experiments

Regular decomposition, 40x20 elements, Mosek 8.0
Basic problem (arrow-type matrix, no vibration constraints)

no of doms	no of vars	size of matrix	no of iters	CPU total	CPU per iter	speedup total	speedup /iter
1	801	1681	53	2489	47	1	1
2	844	882	66	778	12	3	4
8	1032	243	57	49	0.86	51	55
32	1492	73	55	11	0.19	235	244
50	1764	51	54	8	0.14	323	329
200	3544	19	45	5	0.10	553	470
34	22997	11...260	42	1206	29	2	2

Automatic decomposition using software SparseCoLO
by Kim, Kojima, Mevissen and Yamashita (2011)

Numerical experiments

Regular decomposition, 40x20 elements, Mosek 8.0

Problem with vibration constraints

no of matrices	no of vars	size of matrix	no of iters	CPU total	CPU per iter	speedup total	speedup /iter
2	801	1681	64	3894	61	1	1
16	1746	243	59	127	2.15	31	28
64	3384	73	54	27	0.50	144	122
100	4263	51	55	25	0.45	155	136
400	9258	19	37	18	0.49	216	125

and without again, for comparison:

no of matrices	no of vars	size of matrix	no of iters	CPU total	CPU per iter	speedup total	speedup /iter
1	801	1681	53	2489	47	1	1
8	1032	243	57	49	0.86	51	55
32	1492	73	55	11	0.19	235	244
50	1764	51	54	8	0.14	323	329
200	3544	19	45	5	0.10	553	470

Numerical experiments

Regular decomposition, 120x60 elements, Mosek 8.0
Basic problem (arrow-type matrix, no vibration constraints)

no of doms	no of vars	size of matrix	no of iters	CPU total	CPU per iter	speedup total	speedup /iter
1	7200	14641	178	5089762	28594	1	1
50	9524	339	85	1475	17.4	3541	1648
200	12904	99	72	209	2.9	24355	9851
450	16984	51	67	107	1.6	47568	17905
800	21764	33	61	82	1.3	62070	21271
1800	33424	19	44	77	1.6	66101	18196

estimated; 508976 sec $\approx$ 2 months

Numerical experiments

Regular decomposition, Mosek 8.0

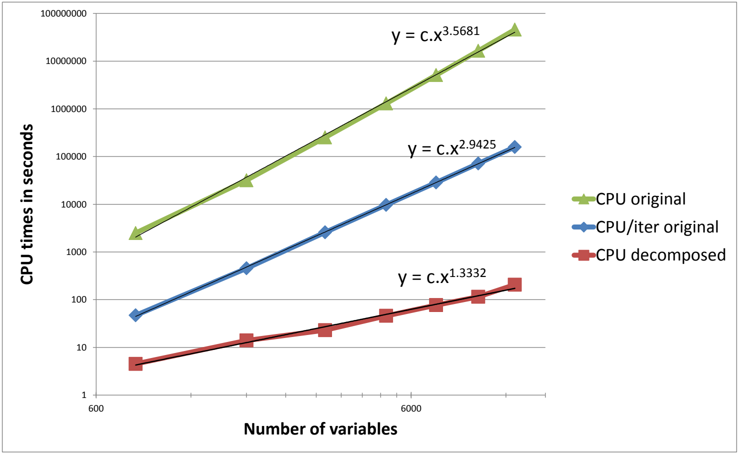
Basic problem (arrow-type matrix, no vibration constraints)

“best” decomposition speedup (subdomain = 4 elements)

problem	ORIGINAL no of vars	ORIGINAL size of matrix	ORIGINAL CPU total	DECOMPOSED no of vars	DECOMPOSED size of matrix	DECOMPOSED CPU total	speedup
40x20	801	1681	2489	3544	19	8	311
60x30	1801	3721	31835	8164	19	25	1273
80x40	3201	6561	252355	14684	19	23	10972
100x50	5001	10201	1298087	23104	19	46	28219
120x60	7201	14641	5091862	33424	19	77	66128
140x70	9801	19881	16436180	45664	19	115	142923
160x80	12801	25921	45804946	59764	19	206	222354
complexity $c \cdot \text{size}^q$			$q = 3.5$			$q = 1.33$	

times estimated; 45804946 sec $\approx$ 18 months

CPU time, original versus decomposed



THE END