Solving Nonconvex SDP Problems of Structural Optimization by PENNON

Michal Kočvara and Michael Stingl

UTIA Prague and University of Erlangen

Outline

- Structural design with stability, vibration control
- FMO—a particular case of structural design
- Solving nonconvex SDP by PENNON
- Examples

MPEC:

$$\min_{\rho, u} F(\rho, u)$$

s.t.

$$\rho \in U_{\mathrm{ad}}$$

$$u \text{ solves } \mathcal{E}(\rho, u)$$

$F(\rho, u)$	...	cost functional (weight, stiffness, peak stress...)
ρ	...	design variable (thickness, material properties, shape...)
u	...	state variable (displacements, stresses)
U_{ad}	...	admissible designs

WEIGHT versus STIFFNESS:

- W weight $\sum \rho_i$
- C stiffness (compliance) $f^T u$

Equilibrium constraint: u solves $\mathcal{E}(\rho, u) \longrightarrow \sum(\rho_i K_i)u = f$

WEIGHT versus STIFFNESS:

- W weight $\sum \rho_i$
- C stiffness (compliance) $f^T u$

Equilibrium constraint: u solves $\mathcal{E}(\rho, u) \longrightarrow \sum(\rho_i K_i) u = f$

$$\min C$$

s.t.

$$W \leq \widehat{W}$$

equilibrium

Structural design problems

WEIGHT versus STIFFNESS:

- W weight $\quad \sum \rho_i$
- C stiffness (compliance) $\quad f^T u$

Equilibrium constraint: u solves $\mathcal{E}(\rho, u) \longrightarrow \sum(\rho_i K_i)u = f$

$\min C$ s.t. $W \leq \widehat{W}$ equilibrium	$\min W$ s.t. $C \leq \widehat{C}$ equilibrium

S. Timoshenko:

> *Experience showed that structures like bridges or aircrafts may fail in some cases not on account of high stresses but owing to insufficient elastic stability.*

Structural design with free vibration control

Three quantities to control:

- W weight $\quad \sum \rho_i$
- C stiffness (compliance) $\quad f^T u$
- λ min. eigenfrequency $\quad K(\rho)u = \lambda M(\rho)u$

Structural design with free vibration control

Three quantities to control:

- W weight $\sum \rho_i$
- C stiffness (compliance) $f^T u$
- λ min. eigenfrequency $K(\rho)u = \lambda M(\rho)u$

$\min C$

s.t.

$W \leq \widehat{W}$

$\lambda \geq \widehat{\lambda}$

equilibrium

Three quantities to control:

- W weight $\quad \sum \rho_i$
- C stiffness (compliance) $\quad f^T u$
- λ min. eigenfrequency $\quad K(\rho)u = \lambda M(\rho)u$

$\min C$ s.t. $W \leq \widehat{W}$ $\lambda \geq \widehat{\lambda}$ equilibrium	$\min W$ s.t. $C \leq \widehat{C}$ $\lambda \geq \widehat{\lambda}$ equilibrium

Three quantities to control:

- W weight $\sum \rho_i$
- C stiffness (compliance) $f^T u$
- λ min. eigenfrequency $K(\rho)u = \lambda M(\rho)u$

$\min C$ s.t. $W \leq \widehat{W}$ $\lambda \geq \widehat{\lambda}$ equilibrium	$\min W$ s.t. $C \leq \widehat{C}$ $\lambda \geq \widehat{\lambda}$ equilibrium	$\max \lambda$ s.t. $W \leq \widehat{W}$ $C \leq \widehat{C}$ equilibrium

Three quantities to control:

- W weight $\quad \sum \rho_i$
- C stiffness (compliance) $\quad f^T u$
- λ critical buckling force $\quad K(\rho)u = \lambda G(\rho, u)u$

$\min C$	$\min W$	$\max \lambda$
s.t.	s.t.	s.t.
$W \le \widehat{W}$	$C \le \widehat{C}$	$W \le \widehat{W}$
$\lambda \ge 1$	$\lambda \ge 1$	$C \le \widehat{C}$
equilibrium	equilibrium	equilibrium

Three quantities to control:

- W weight $\quad \sum \rho_i$
- C stiffness (compliance) $\quad f^T u$
- λ critical buckling force $\quad K(\rho)u = \lambda G(\rho, u)u$

$\min C$	$\min W$	~~$\max \lambda$~~
s.t.	s.t.	~~s.t.~~
$W \le \widehat{W}$	$C \le \widehat{C}$	~~$W \le \widehat{W}$~~
$\lambda \ge 1$	$\lambda \ge 1$	~~$C \le \widehat{C}$~~
equilibrium	equilibrium	~~equilibrium~~

Three quantities to control:

- W weight $\sum \rho_i$
- C stiffness (compliance) $f^T u$
- λ critical buckling force $K(\rho)u = \lambda G(\rho, u)u$

~~$\min C$ s.t. $W \le \widehat{W}$, $\lambda \ge 1$, equilibrium~~	$\min W$ s.t. $C \le \widehat{C}$, $\lambda \ge 1$, equilibrium	~~$\max \lambda$ s.t. $W \le \widehat{W}$, $C \le \widehat{C}$, equilibrium~~

Structural design with stability control

Lowest (positive) eigenvalue of

$$K(\rho)u = \lambda G(\rho, u)u$$

(critical force) should be bigger than 1.

$$\min_{\rho,u} W(\rho)$$

s.t.

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

$$f^T u \le \widehat{C}$$

$$\rho_i \ge 0, \quad i = 1, \ldots, m$$

$$\lambda \ge 1$$

Two standard tricks:

$$K(\rho) \succ 0, \qquad u = K(\rho)^{-1} f$$

$$f^T K(\rho)^{-1} f \le \widehat{C} \quad \Longleftrightarrow \quad \begin{pmatrix} \widehat{C} & f^T \\ f & K(\rho) \end{pmatrix} \succeq 0$$

Two standard tricks:

$$K(\rho) \succ 0, \qquad u = K(\rho)^{-1} f$$

$$f^T K(\rho)^{-1} f \le \widehat{C} \quad \Longleftrightarrow \quad \begin{pmatrix} \widehat{C} & f^T \\ f & K(\rho) \end{pmatrix} \succeq 0$$

$$\left. \begin{aligned} K(\rho)u = \lambda G(\rho, u)u \\ \lambda \ge 1 \end{aligned} \right\} \quad \Longleftrightarrow \quad K(\rho) - G(\rho, u) \succeq 0$$

$$\Longleftrightarrow \quad K(\rho) - \widetilde{G}(\rho) \succeq 0$$

$$\widetilde{G}(\rho) = G(\rho, K(\rho)^{-1} f)$$

Formulated as SDP problem:

$$\min_{\rho} W(\rho)$$

subject to

$$K(\rho) - \widetilde{G}(\rho) \succeq 0$$

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

$$\rho_i \geq 0, \quad i = 1, \ldots, m$$

where

$$K(\rho) = \sum \rho_i K_i, \qquad \widetilde{G}(\rho) = \sum \widetilde{G}_i$$

Free Material 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.

The design variables are the material properties at each point of the structure.

Free Material 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.

The design variables are the material properties at each point of the structure.

$$\inf_{\substack{E \succcurlyeq 0 \\ \int tr(E) dx \leq 1}} \sup_{u \in U} -\frac{1}{2} \int_\Omega \langle E e(u), e(u) \rangle \, dx + \int_{\Gamma_2} f \cdot u \, dx$$

Free Material 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.

The design variables are the material properties at each point of the structure.

$$\inf_{\substack{E \succcurlyeq 0 \\ \int tr(E) dx \leq 1}} \sup_{u \in U} -\frac{1}{2} \int_\Omega \langle E e(u), e(u) \rangle \, dx + \int_{\Gamma_2} f \cdot u \, dx$$

$$\inf_{\substack{\rho \geq 0 \\ \int \rho \, dx \leq 1}} \sup_{u \in U} -\frac{1}{2} \int_\Omega \rho \langle e(u), e(u) \rangle \, dx + \int_{\Gamma_2} f \cdot u \, dx$$

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.

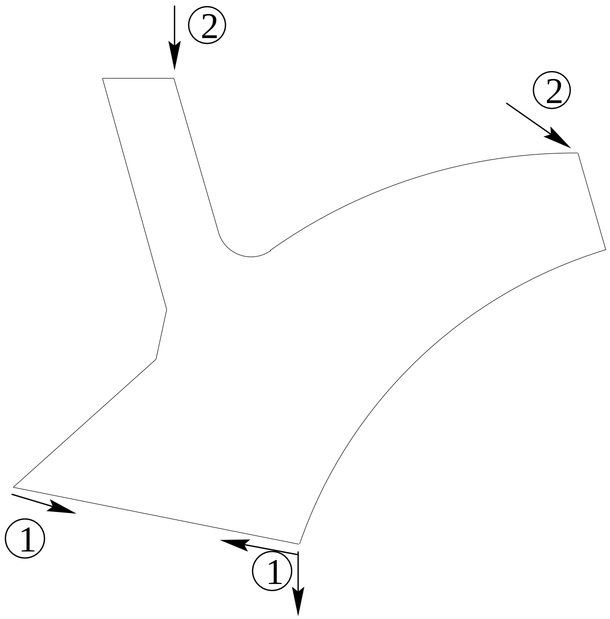
The design variables are the material properties at each point of the structure.

$$\inf_{\substack{E\succcurlyeq 0\\ \int tr(E)dx\le 1}} \sup_{u\in U} -\frac{1}{2}\int_\Omega \langle Ee(u), e(u)\rangle\, dx + \int_{\Gamma_2} f\cdot u\, dx$$

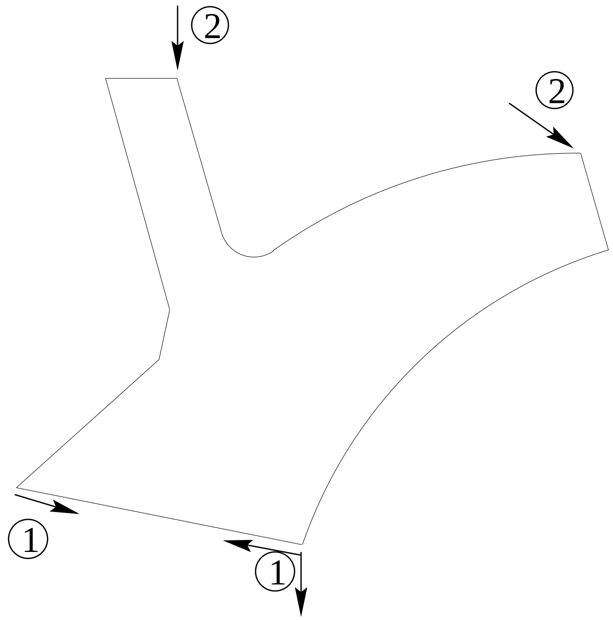
$$\inf_{\substack{\rho\ge 0\\ \int \rho\, dx\le 1}} \sup_{u\in U} -\frac{1}{2}\int_\Omega \rho\langle e(u), e(u)\rangle\, dx + \int_{\Gamma_2} f\cdot u\, dx$$

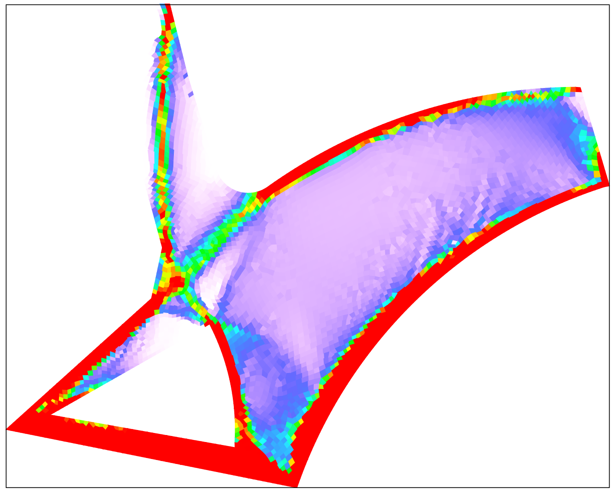
$$\inf_{\alpha\in\mathbb{R}, u\in U}\left\{\alpha - f^T u \,|\, \alpha \ge \frac{m}{2}u^T A_i\, u \quad \text{for} \quad i=1,\ldots,m\right\}$$

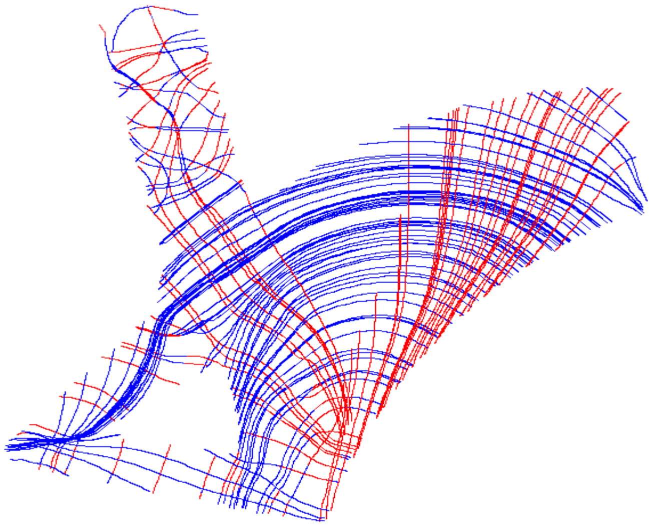
FMO, example

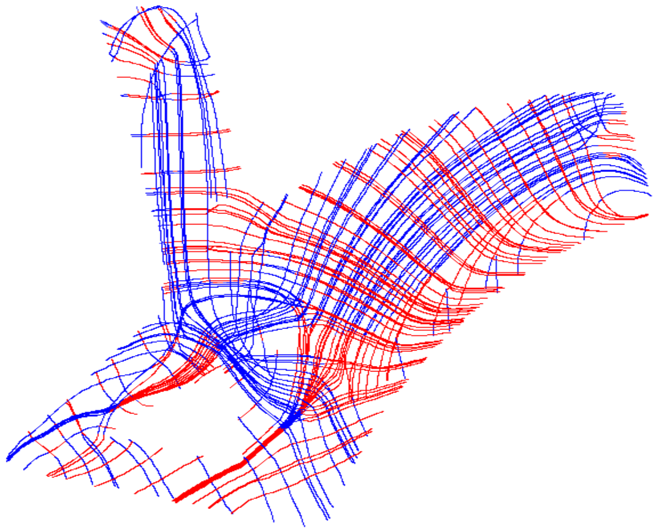


FMO, example









Structural design with stability control

Formulated as SDP problem:

$$\min_{\rho} W(\rho)$$

subject to

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

$$\rho_i \geq 0, \quad i = 1, \ldots, m$$

$$K(\rho) - \widetilde{G}(\rho) \succeq 0$$

where

$$K(\rho) = \sum \rho_i K_i, \qquad \widetilde{G}(\rho) = \sum \widetilde{G}_i$$

PENNON for SDP

Problem:

$$\min_{x \in \mathbb{R}^n} \left\{ b^T x : \mathcal{A}(x) \preccurlyeq 0 \right\}$$

$$\mathcal{A} : \mathbb{R}^n \longrightarrow \mathbb{S}_d$$

Notation:

$\langle A, B \rangle_{\mathbb{S}_d}$ $:=$ tr $\left(A^T B \right)$ *inner product* on $\mathbb{S}_d$

$\mathbb{S}_{d_+}$ $= \{ A \in \mathbb{S}_d \mid A$ *positive semidefinite*$\}$

$U \in \mathbb{S}_{d_+}$ *matrix multiplier (dual variable)*

Φ_p *penalty function* on $\mathbb{S}_d$

PENNON for SDP: algorithm

Generalized augmented Lagrangian algorithm for SDP:

We have

$$\mathcal{A}(x) \preccurlyeq 0 \Longleftrightarrow \Phi_p(\mathcal{A}(x)) \preccurlyeq 0$$

and the corresponding *augmented Lagrangian*

$$F(x, U, p) := f(x) + \langle U, \Phi_p(\mathcal{A}(x)) \rangle_{\mathbb{S}_d}$$

Algorithm:

(i)	Find x^{k+1} satisfying $\|\nabla_x F(x, U^k, p^k)\| \leq \epsilon^k$
(ii)	$U^{k+1} = D_{\mathcal{A}} \Phi_p(\mathcal{A}(x); U^k)$
(iii)	$p^{k+1} < p^k$

Best choice of Φ: $\Phi(A) = (A - I)^{-1} - I$

PENNON for SDP: theory

Based on Breitfeld-Shanno, 1993; generalized by M. Stingl, 2003

Assume:

1. $f, \mathcal{A} \in C^2$
2. $x \in \Omega$ nonempty, bounded
3. Constraint Qualification

Then $\exists$ an index set $\mathcal{K}$ so that:

- $x_k \longrightarrow \hat{x},\ k \in \mathcal{K}$
- $U_k \longrightarrow \widehat{U},\ k \in \mathcal{K}$
- $(\hat{x}, \widehat{U})$ satisfies first-order optimality conditions

PENNON for SDP: Hessian

The reciprocal barrier function in SDP

$$\Phi(A) = (A - I)^{-1} - I$$

Hessian

$$\frac{\partial^2}{\partial x_i \partial x_j}\Phi(\mathcal{A}(x)) =$$

$$(\mathcal{A}(x) - I)^{-1}\frac{\partial \mathcal{A}(x)}{\partial x_i}(\mathcal{A}(x) - I)^{-1}\frac{\partial \mathcal{A}(x)}{\partial x_j}(\mathcal{A}(x) - I)^{-1}$$

$$+ (\mathcal{A}(x) - I)^{-1}\frac{\partial^2 \mathcal{A}(x)}{\partial x_i \partial x_j}(\mathcal{A}(x) - I)^{-1}$$

$$+ (\mathcal{A}(x) - I)^{-1}\frac{\partial \mathcal{A}(x)}{\partial x_j}(\mathcal{A}(x) - I)^{-1}\frac{\partial \mathcal{A}(x)}{\partial x_i}(\mathcal{A}(x) - I)^{-1}$$

PENNON fo SDP: complexity

FMO with stability constraint (nonconvex SDP)

$$K(\rho) + G(\rho) \succeq 0$$

$$K(\rho) = \sum_{e=1}^{M} \rho_e K_e$$

$$G(\rho) = \sum_{e=1}^{M} G_e \qquad G_e(\rho) = \sum_{k=1}^{K} B_{e,k}^T S_{e,k}(\rho) B_{e,k}$$

$$S_{e,k}(\rho) = \begin{pmatrix} \sigma_1 & \sigma_3 \\ \sigma_3 & \sigma_2 \end{pmatrix} \qquad \sigma_{e,k}(\rho) = \rho_e \widetilde{B}_{e,k}^T (K^{-1}(\rho) f)_e$$

PENNON fo SDP: complexity

FMO with stability constraint (nonconvex SDP)

$$K(\rho) + G(\rho) \succeq 0$$

$$K(\rho) = \sum_{e=1}^{M} \rho_e K_e$$

$$G(\rho) = \sum_{e=1}^{M} G_e \qquad G_e(\rho) = \sum_{k=1}^{K} B_{e,k}^T S_{e,k}(\rho) B_{e,k}$$

$$S_{e,k}(\rho) = \begin{pmatrix} \sigma_1 & \sigma_3 \\ \sigma_3 & \sigma_2 \end{pmatrix} \qquad \sigma_{e,k}(\rho) = \rho_e \widetilde{B}_{e,k}^T (K^{-1}(\rho) f)_e$$

memory: $O(M^2)$ (M = 500 ≈ 64 MB)

PENNON fo SDP: complexity

FMO with stability constraint (nonconvex SDP)

$$K(\rho) + G(\rho) \succeq 0$$

$$K(\rho) = \sum_{e=1}^{M} \rho_e K_e$$

$$G(\rho) = \sum_{e=1}^{M} G_e \qquad G_e(\rho) = \sum_{k=1}^{K} B_{e,k}^T S_{e,k}(\rho) B_{e,k}$$

$$S_{e,k}(\rho) = \begin{pmatrix} \sigma_1 & \sigma_3 \\ \sigma_3 & \sigma_2 \end{pmatrix} \qquad \sigma_{e,k}(\rho) = \rho_e \widetilde{B}_{e,k}^T (K^{-1}(\rho) f)_e$$

memory: $O(M^2)$ (M = 500 ≈ 64 MB)

CPU: $O(K^2 * d^2 * M^3)$ for one Hessian assembling

PENNON fo SDP: complexity

FMO with stability constraint (nonconvex SDP)

$$K(\rho) + G(\rho) \succeq 0$$

$$K(\rho) = \sum_{e=1}^{M} \rho_e K_e$$

$$G(\rho) = \sum_{e=1}^{M} G_e \qquad G_e(\rho) = \sum_{k=1}^{K} B_{e,k}^T S_{e,k}(\rho) B_{e,k}$$

$$S_{e,k}(\rho) = \begin{pmatrix} \sigma_1 & \sigma_3 \\ \sigma_3 & \sigma_2 \end{pmatrix} \qquad \sigma_{e,k}(\rho) = \rho_e \widetilde{B}_{e,k}^T (K^{-1}(\rho) f)_e$$

memory: $O(M^2)$ (M = 500 ≈ 64 MB)

CPU: $O(K^2 * d^2 * M^3)$ for one Hessian assembling
All dense matrix-matrix multiplications implemented in BLAS

PENNON fo SDP: complexity

FMO with stability constraint (nonconvex SDP)

$$K(\rho) + G(\rho) \succeq 0$$

$$K(\rho) = \sum_{e=1}^{M} \rho_e K_e$$

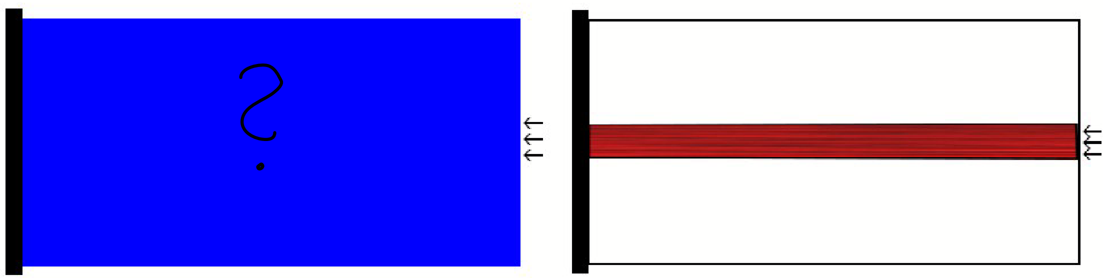
$$G(\rho) = \sum_{e=1}^{M} G_e \qquad G_e(\rho) = \sum_{k=1}^{K} B_{e,k}^T S_{e,k}(\rho) B_{e,k}$$

$$S_{e,k}(\rho) = \begin{pmatrix} \sigma_1 & \sigma_3 \\ \sigma_3 & \sigma_2 \end{pmatrix} \qquad \sigma_{e,k}(\rho) = \rho_e \widetilde{B}_{e,k}^T (K^{-1}(\rho) f)_e$$

memory: $O(M^2)$ (M = 500 ≈ 64 MB)

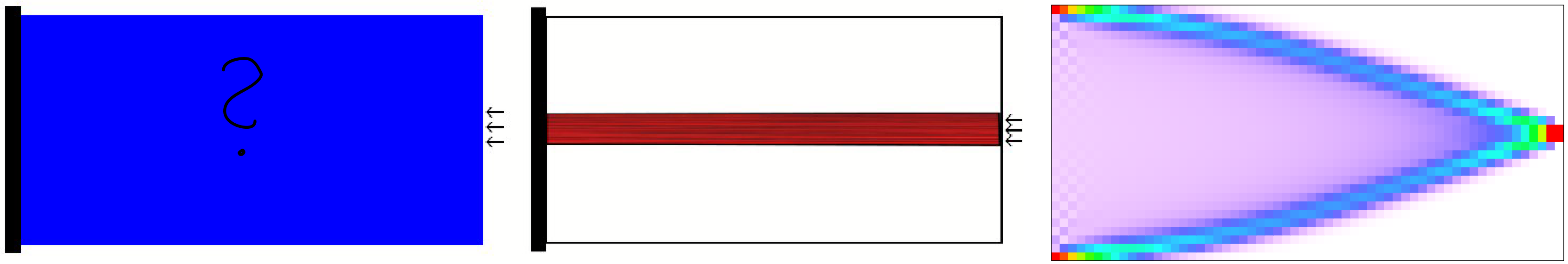
CPU: $O(K^2 * d^2 * M^3)$ for one Hessian assembling
Pentium 4, 2.4GHz, ∼100 Newton steps:
400 elements . . . 8 h 45 min, 1000 elements . . . ∼130 hours

FMO with vibration constraint (linear SDP)



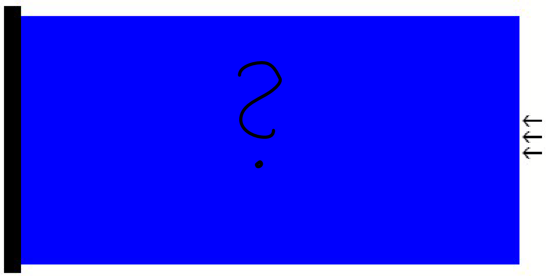
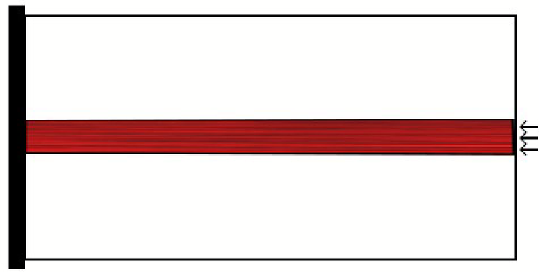
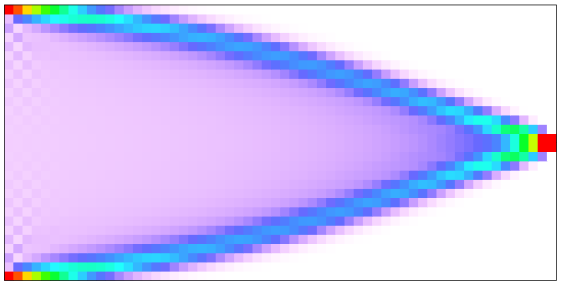
Examples, FMO w. stability constraint

FMO with vibration constraint (linear SDP)



Examples, FMO w. stability constraint

FMO with vibration constraint (linear SDP)

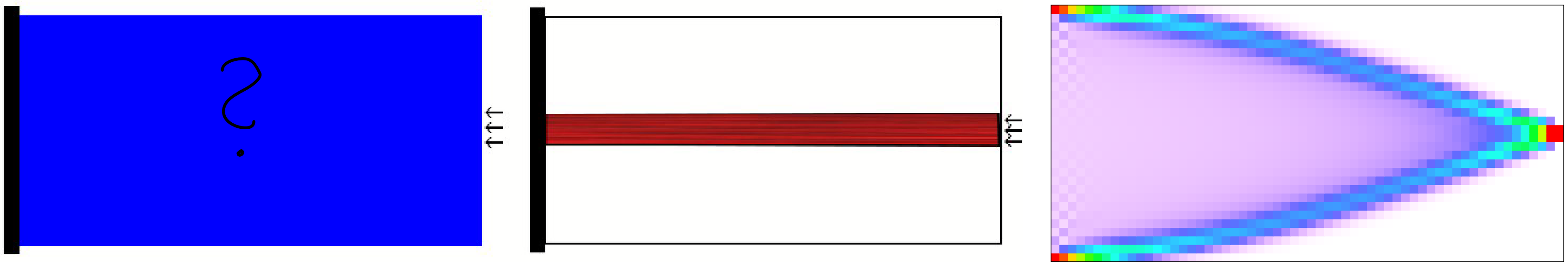
  

Linear SDP, SDPA input file (Pentium 4, 2.5 GHz):

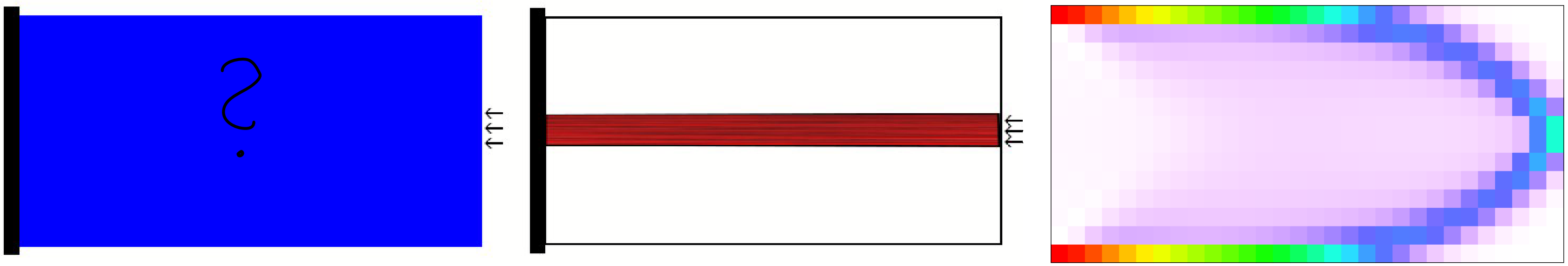
problem	no. of variables	size of matrix
shmup-3	420	1801+840
shmup-4	800	3361+1600
shmup-5	1800	7441+3660

problem	PENNON	SDPT3	SDPA	DSDP	CSDP	SeDuMi
shmup-3	381	417	497	439	1395	23322
shmup-4	2095	2625	2952	2798	5768	$>$127320
shmup-5	14149	23535	m	fail	m	m

FMO with vibration constraint (linear SDP)

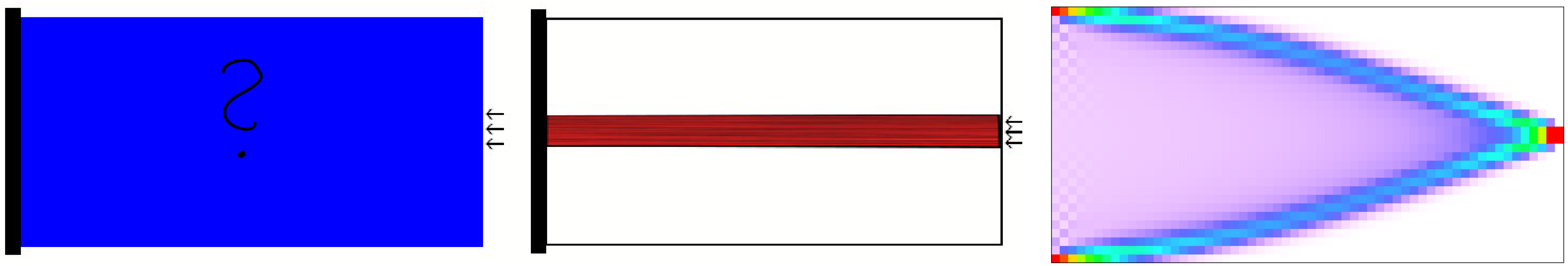


FMO with stability constraint (nonlinear SDP)



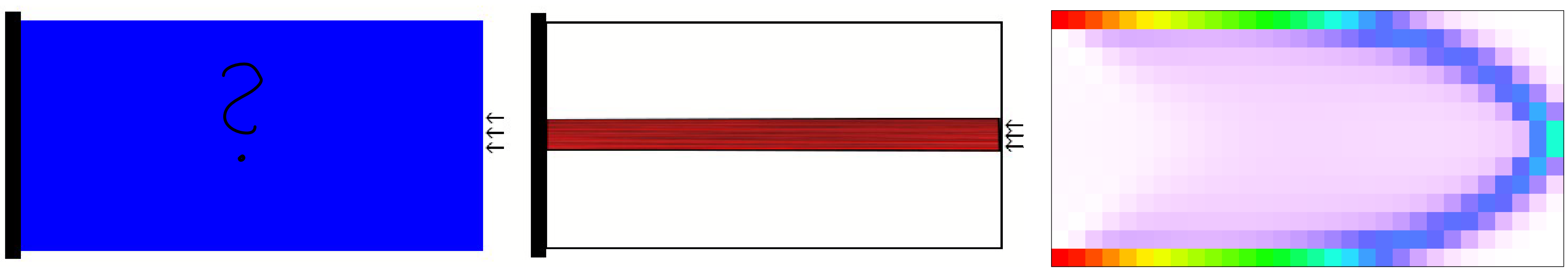
Examples, FMO w. stability constraint

FMO with vibration constraint (linear SDP)



shmup3 (420 elements) . . . 6 min 20 sec

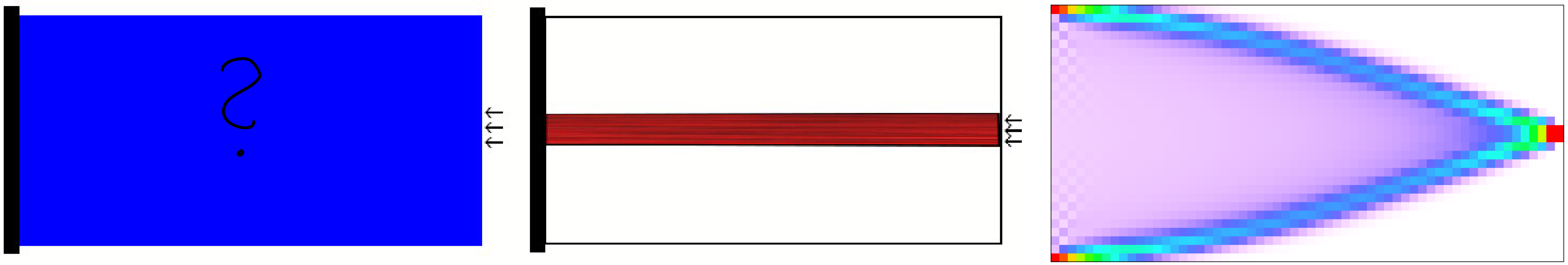
FMO with stability constraint (nonlinear SDP)



shmup3 (420 elements) . . . 8 hours

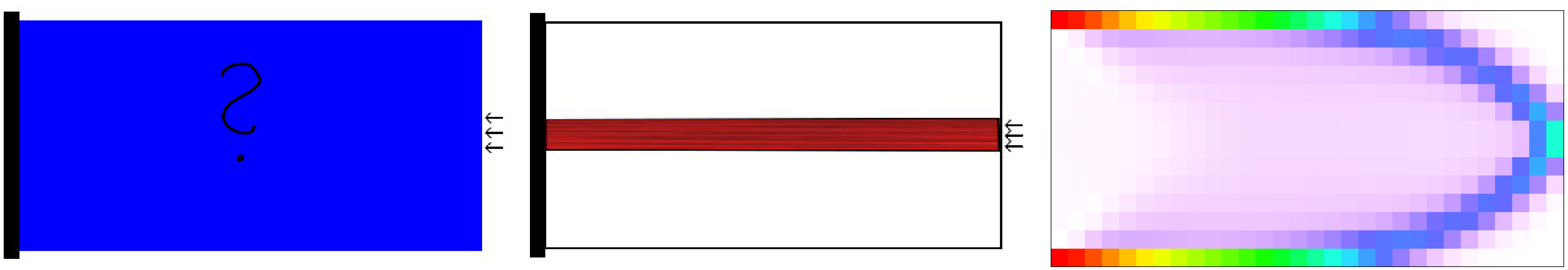
Examples, FMO w. stability constraint

FMO with vibration constraint (linear SDP)



shmup3 (420 elements) . . . 6 min 20 sec

FMO with stability constraint (nonlinear SDP)



shmup3 (420 elements) . . . 8 hours

shmup3 with no SDP constraints (convex NLP) . . . 1 sec

Conclusions (so far)

PENNON algorithm works well for nonconvex SDP
–accurate solution within 60–100 internal iterations–
(more experience from BMI problems and truss design)

Conclusions (so far)

PENNON algorithm works well for nonconvex SDP
–accurate solution within 60–100 internal iterations–
(more experience from BMI problems and truss design)

complexity of second-order method too high
(for "large" problems)

Conclusions (so far)

PENNON algorithm works well for nonconvex SDP
–accurate solution within 60–100 internal iterations–
(more experience from BMI problems and truss design)

complexity of second-order method too high
(for “large” problems)

FIRST-ORDER METHOD

Hessian free methods

Use conjugate gradient method for solving the Newton system

Use finite difference formula for Hessian-vector products:

$$\nabla^2 F(x_k)v \approx \frac{\nabla F(x_k + hv) - \nabla F(x_k)}{h}$$

with $h = (1 + \|x_k\|_2 \sqrt{\varepsilon})$

Complexity: Hessian-vector product = gradient evaluation
need for Hessian-vector-product type preconditioner

Limited accuracy (4–5 digits)

Preconditioners

Should be:

- efficient (obvious but often difficult to reach)
- simple (low complexity)
- only use Hessian-vector product (NOT Hessian elements)

Preconditioners

Should be:

- efficient (obvious but often difficult to reach)
- simple (low complexity)
- only use Hessian-vector product (NOT Hessian elements)

Diagonal

$$M = \mathbf{diag}\,(H)$$

simple, not (considered) very efficient

does not satisfy point 3

Preconditioners

Should be:

- efficient (obvious but often difficult to reach)
- simple (low complexity)
- only use Hessian-vector product (NOT Hessian elements)

L-BFGS (Morales-Nocedal, SIOPT 2000)
–start with CG (no precond.)
–use CG iterations as *correction pairs* $\longrightarrow$ build M using L-BFGS
–next Newton step $\longrightarrow$ use M as precondiotiner
–from CG iterations build new M

relatively inexpensive (16–32 correction pairs)

mixed success

Preconditioners

Should be:

- efficient (obvious but often difficult to reach)
- simple (low complexity)
- only use Hessian-vector product (NOT Hessian elements)

A-inv (approximate inverse) (Benzi-Collum-Tuma, SISC 2000)

$$M = ZD^{-1}Z^T \approx H^{-1}$$

Z sparse approximation of Cholesky factor L^{-1}
computed directly from H by incomplete $H-$orthogonalization
small elements dropped to prevent (introduce) sparsity
uses only Hessian-vector products

relatively expensive, dependent on (sensitive to) the dropping parameter

efficient (often)

Linear SDP, dense Hessian:

Complexity of Hessian evaluation

- $O(m_A^3 n + m_A^2 n^2)$ for dense matrices
- $O(m_A^2 n + K^2 n^2)$ for sparse matrices
  (K ... max. number of nonzeros in $A_i,\ i = 1, \ldots, n$)

Complexity of Cholesky algorithm - linear SDP

- $O(n^3)$

Library of examples with large n and small m
(courtesy of Kim Toh)

CG-exact much better than Cholesky
CG-approx much better than CG-exact

Number of Newton steps (yellow) and QMR iterations (white)

problem	pensdp	pen_QMR		pen_approx-QMR	
ham_7_5_6	54	47	109	45	78
ham_9_8	54	57	132	61	91
ham_8_3_4		51	116	50	89
ham_9_5_6				59	108
theta32#	50	48	458	50	453
theta42#	53	52	435	53	718
theta6#	71	61	574	60	362
theta62#				52	404
theta8	61	62	744	62	504
theta82				57	482
theta83				58	647
theta10		68	748	62	473
theta102				58	744
theta103				56	769
theta104				56	834
theta12		63	606	66	518
keller4	47	54	376	52	864
sanr200-0.7	53	55	531	56	698

Total CPU time (white) and time per onew Newton step (yellow)

problem	n	m	pensdp		pen_QMR		pen_approx-QMR	
ham_7_5_6	1793	128	176	3.26	47	1.00	4	0.09
ham_9_8	2305	512	497	9.20	244	4.28	197	3.23
ham_8_3_4	16129	256			6944	136.16	90	1.80
ham_9_5_6	53761	512					1499	25.41
theta32#	150	2286	200	4.00	71	1.48	11	0.22
theta42#	200	5986	2998	56.57	827	15.90	49	0.92
theta6#	4375	300	1714	24.14	490	8.03	60	1.00
theta62#	13390	300					118	2.27
theta8	7905	400	15139	248.18	1975	31.85	350	5.65
theta82	23872	400					971	17.04
theta83	39862	400					3274	56.45
theta10	12470	500		961.28	5842	85.91	703	11.34
theta102	37467	500					3635	62.67
theta103	62516	500					9850	175.89
theta104	87845	500					20329	363.02
theta12	17979	600			14098	223.78	1365	20.68
keller4	5101	171	3236	68.85	587	10.87	86	1.65
sanr200-0.7	6033	200	5790	109.25	916	16.65	103	1.84

Number of QMR iterations per one Newton step

problem	n	m	pen_QMR	pen_appr-QMR
ham_7_5_6	1793	128	2	2
ham_9_8	2305	512	2	1
ham_8_3_4	16129	256	2	2
ham_9_5_6	53761	512		2
theta32	150	2286	10	9
theta42	200	5986	8	14
theta6	4375	300	9	6
theta62	13390	300		8
theta8	7905	400	12	8
theta82	23872	400		8
theta83	39862	400		11
theta10	12470	500	11	8
theta102	37467	500		13
theta103	62516	500		14
theta104	87845	500		15
theta12	17979	600	10	8
keller4	5101	171	7	17

Nonlinear SDP—FMO with stability constraints

Can CG + approx. Hessian help?

Partly. . .

No preconditioning, approx. Hessian:
as many gradient evaluations as CG steps (good)
CG with no preconditioning inefficient (bad)

Nonlinear SDP—FMO with stability constraints

Can CG + approx. Hessian help?

Partly. . .

No preconditioning, approx. Hessian:
as many gradient evaluations as CG steps (good)
CG with no preconditioning inefficient (bad)

Evaluation of exact diagonal as expensive as evaluation of full Hessian
Evaluation of approx. diagonal . . . . . .

Only L-BFGS preconditioner can be used — but it isn't really efficient

		pennon		app-CG(BFGS-N)		
	n	time	Nwt	time	Nwt	CG
shape2	200	1699	63	840	62	3192
shape3	420	18949	77	10622	75	8016

Conclusions, part II

Hessian-free SDP:

- First promising results, more testing (and coding) needed

THE END