Introducing PenLab
a MATLAB code for NLP-SDP

Michal Kočvara

School of Mathematics, The University of Birmingham

jointly with

Jan Fiala
Numerical Algorithms Group

Michael Stingl
University of Erlangen-Nürnberg

Berlin, August, 2012

PENNON collection

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

– *one algorithm to rule them all* –

READY

- PENNLP AMPL, MATLAB, C/Fortran
- PENSDP MATLAB/YALMIP, SDPA, C/Fortran
- PENBMI MATLAB/YALMIP, C/Fortran

(relatively) NEW

- PENNON (NLP + SDP) extended AMPL, MATLAB, C/Fortran

The problem

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

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

The algorithm

Based on penalty/barrier functions $\varphi_g : \mathbb{R} \to \mathbb{R}$ and $\Phi_P : \mathbb{S}^p \to \mathbb{S}^p$:

$$\begin{aligned} g_i(x) \leq 0 &\iff p_i \varphi_g(g_i(x)/p_i) \leq 0, \quad i = 1, \ldots, m \\ Z \preceq 0 &\iff \Phi_P(Z) \preceq 0, \quad Z \in \mathbb{S}^p . \end{aligned}$$

Augmented Lagrangian of (NLP-SDP):

$$\begin{aligned} F(x,Y,u,\underline{U},\overline{U},p) = f(x,Y) + \textstyle\sum_{i=1}^{m_g} u_i p_i \varphi_g(g_i(x,Y)/p_i) \\ + \textstyle\sum_{i=1}^{k} \langle \underline{U}_i, \Phi_P(\underline{\lambda}_i I - Y_i) \rangle + \textstyle\sum_{i=1}^{k} \langle \overline{U}_i, \Phi_P(Y_i - \overline{\lambda}_i I) \rangle \, ; \end{aligned}$$

here $u \in \mathbb{R}^{m_g}$ and $\underline{U}_i, \overline{U}_i$ are Lagrange multipliers.

The algorithm

A generalized Augmented Lagrangian algorithm (based on R. Polyak '92, Ben-Tal–Zibulevsky '94, Stingl '05):

Given $x^1, Y^1, u^1, \underline{U}^1, \overline{U}^1; p_i^1 > 0,\ i = 1, \ldots, m_g$ and $P > 0$.
For $k = 1, 2, \ldots$ repeat till a stopping criterium is reached:

$$
\begin{array}{ll}
(i) & \text{Find } x^{k+1} \text{ and } Y^{k+1} \text{ s.t. } \|\nabla_x F(x^{k+1}, Y^{k+1}, u^k, \underline{U}^k, \overline{U}^k, p^k)\| \leq K \\
(ii) & u_i^{k+1} = u_i^k \varphi_g'(g_i(x^{k+1})/p_i^k), \quad i = 1, \ldots, m_g \\
 & \underline{U}_i^{k+1} = D_{\mathcal{A}} \Phi_P((\underline{\lambda}_i I - Y_i); \underline{U}_i^k), \quad i = 1, \ldots, k \\
 & \overline{U}_i^{k+1} = D_{\mathcal{A}} \Phi_P((Y_i - \overline{\lambda}_i I); \overline{U}_i^k), \quad i = 1, \ldots, k \\
(iii) & p_i^{k+1} < p_i^k,\ i = 1, \ldots, m_g \\
 & P^{k+1} < P^k\,.
\end{array}
$$

Interfaces

How to enter the data – the functions and their derivatives?

- Matlab interface
- AMPL interface
- c/Fortran interface

Key point: Matrix variables are treated as vectors

What's new

PENNON being implemented in NAG (The Numerical Algorithms Group) library

The first routines should appear in the NAG Fortran Library, Mark 24 (Autumn 2012)

By-product:
PenLab — free, open, fully functional version of PENNON coded in MATLAB

PenLab

PenLab — free, open, fully functional version of PENNON coded in Matlab

- Open source, all in MATLAB (one MEX function)
- The basic algorithm is identical
- Some data handling routines not (yet?) implemented
- PenLab runs just as PENNON but is slower

Pre-programmed procedures for

- standard NLP (with AMPL input!)
- linear SDP (reading SDPA input files)
- bilinear SDP (=BMI)
- easy to add more (QP, PMI, robust QP, ...)

PenLab

The problem

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

$\mathcal{A}_i(x,Y)\ldots$ nonlinear matrix operators

$h_i(x,Y) = 0$ input as $0 \le g_i(x,Y) \le 0$ but treated as genuine equalities

PenLab

Solving a problem:

- prepare a structure `penm` containing basic problem data
- `>> prob = penlab(penm);` MATLAB class containing all data
- `>> solve(prob);`
- results in class `prob`

The user has to provide MATLAB functions for

- function values
- gradients
- Hessians (for nonlinear functions)

of all $f, g, \mathcal{A}$.

Structure `penm` and f/g/h functions

Example: $\min x_1 + x_2 \quad \text{s.t.} \quad x_1^2 + x_2^2 \leq 1,\ \ x_1 \geq -0.5$

```
penm = [];
penm.Nx = 2;
penm.lbx = [-0.5 ; -Inf];
penm.NgNLN = 1;
penm.ubg = [1];
penm.objfun  = @(x,Y) deal(x(1) + x(2));
penm.objgrad = @(x,Y) deal([1 ; 1]);
penm.confun  = @(x,Y) deal([x(1)^2 + x(2)^2]);
penm.congrad = @(x,Y) deal([2*x(1) ; 2*x(2)]);
penm.conhess = @(x,Y) deal([2 0 ; 0 2]);
% set starting point
penm.xinit = [2,1];
```

Toy NLP-SDP example 1

$$\min_{x\in\mathbb{R}^2} \frac{1}{2}(x_1^2 + x_2^2)$$

$$\text{subject to} \quad \begin{pmatrix} 1 & x_1 - 1 & 0 \\ x_1 - 1 & 1 & x_2 \\ 0 & x_2 & 1 \end{pmatrix} \succeq 0$$

D. Noll, 2007

Toy NLP-SDP example 2

$$
\begin{aligned}
\min_{x \in \mathbb{R}^6} \; & x_1 x_4 (x_1 + x_2 + x_3) + x_3 \\
\text{subject to} \quad & x_1 x_2 x_3 x_4 - x_5 - 25 = 0 \\
& x_1^2 + x_2^2 + x_3^2 + x_4^2 - x_6 - 40 = 0 \\
& \begin{pmatrix} x_1 & x_2 & 0 & 0 \\ x_2 & x_4 & x_2 + x_3 & 0 \\ 0 x_2 + x_3 & x_4 & x_3 & \\ 0 & 0 & x_3 & x_1 \end{pmatrix} \succeq 0 \\
& 1 \leq x_i \leq 5, \; i = 1, 2, 3, 4, \quad x_i \geq 0, \; i = 5, 6
\end{aligned}
$$

Yamashita, Yabe, Harada, 2007
("augmented" Hock-Schittkowski)

Example: nearest correlation matrix

Find a nearest correlation matrix:

$$\min_X \sum_{i,j=1}^{n} (X_{ij} - H_{ij})^2 \tag{1}$$

subject to

$$X_{ii} = 1, \quad i = 1, \ldots, n$$

$$X \succeq 0$$

Example: nearest correlation matrix

The condition number of the nearest correlation matrix must be bounded by κ.

Using the transformation of the variable X:

$$z\widetilde{X} = X$$

The new problem:

$$\min_{z,\widetilde{X}} \sum_{i,j=1}^{n} (z\widetilde{X}_{ij} - H_{ij})^2 \tag{2}$$

subject to

$$z\widetilde{X}_{ii} = 1, \quad i = 1, \ldots, n$$

$$I \preceq \widetilde{X} \preceq \kappa I$$

Example: nearest correlation matrix

For

```
X =
    1.0000 -0.3775 -0.2230  0.7098 -0.4272 -0.0704
   -0.3775  1.0000  0.6930 -0.3155  0.5998 -0.4218
   -0.2230  0.6930  1.0000 -0.1546  0.5523 -0.4914
    0.7098 -0.3155 -0.1546  1.0000 -0.3857 -0.1294
   -0.4272  0.5998  0.5523 -0.3857  1.0000 -0.0576
   -0.0704 -0.4218 -0.4914 -0.1294 -0.0576  1.0000
```

the eigenvalues of the correlation matrix are

```
eigen =
   0.2866   0.2866   0.2867   0.6717   1.6019   2.8664
```

NLP with AMPL input

Pre-programmed. All you need to do:

```
>> penm=nlp_define('datafiles/chain100.nl');
>> prob=penlab(penm);
>> prob.solve();
```

Linear SDP with SDPA input

Pre-programmed. All you need to do:

```
>> sdpdata=readsdpa('datafiles/arch0.dat-s');
>> penm=sdp_define(sdpdata);
>> prob=penlab(penm);
>> prob.solve();
```

Bilinear matrix inequalities (BMI)

Pre-programmed. All you need to do:

```
>> bmidata=define_my_problem; %matrices A, K, ...
>> penm=bmi_define(bmidata);
>> prob=penlab(penm);
>> prob.solve();
```

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

s.t.

$$A_0^i + \sum_{k=1}^{n} x_k A_k^i + \sum_{k=1}^{n} \sum_{\ell=1}^{n} x_k x_\ell K_{k\ell}^i \succcurlyeq 0, \quad i = 1, \ldots, m$$

Example:

Robust quadratic programming on unit simplex with constrained uncertainty set

Nominal QP

$$\min_{x\in\mathbb{R}^n} [x^T A x - b^T x] \quad \text{s.t.} \quad \sum x \leq 1,\ x \geq 0$$

Assume A uncertain: $A \in \mathcal{U} := \{A_0 + \varepsilon U,\ \sigma(U) \leq 1\}$

Robust QP

$$\min_{x\in\mathbb{R}^n} \max_{A\in\mathcal{U}} [x^T A x - b^T x] \quad \text{s.t.} \quad \sum x \leq 1,\ x \geq 0$$

equivalent to

$$\min_{x\in\mathbb{R}^n} [x^T (A_0 + \varepsilon I) x - b^T x] \quad \text{s.t.} \quad \sum x \leq 1,\ x \geq 0$$

Example: Robust QP

Optimal solution: x^*, $A^* = A_0 + \varepsilon U^*$ (non-unique)

Constraint: A^* should share some properties with A_0

For instance: $(A_0)_{ii} = A^*_{ii} = 1$, i.e., $U^*_{ii} = 0$ for all i

$\Longrightarrow$ then the above equivalence no longer holds

Remedy: for a given x^* find a feasible solution $\widehat{A}$ to the maximization problem. This is a linear SDP:

$$\begin{aligned} &\max_{U \in \mathbb{S}^m} \; [{x^*}^T (A_0 + \varepsilon U) x^* - b^T x^*] \\ &\text{s.t.} \\ &-I \preccurlyeq U \preccurlyeq I \\ &\quad U_{ii} = 0, \; i = 1, \ldots, m \end{aligned}$$

Example: Robust QP

Alternatively: find feasible $\hat{A}$ nearest to A^* $\rightarrow$ nonlinear SDP

Finally, we need to solve the original QP with the feasible worst-case $\widehat{A}$ to get a feasible robust solution x_{rob}:

$$\min_{x \in \mathbb{R}^n} [x^T \widehat{A} x - b^T x] \quad \text{s.t.} \quad \sum x \leq 1, \; x \geq 0$$

The whole procedure

- solve QP with $A = A_0$ (nominal)
- solve QP with $A = A_0 + \varepsilon I$ (robust)
- solve (non-)linear SDP to get $\widehat{A}$
- solve QP with $\widehat{A}$ (robust feasible)

Availability

PENNON: Free time-limited academic version of the code available

PENLAB: Free open MATLAB version available in Autumn 2012 from NAG

What's missing?

SOCP (Second-Order Conic Programming) - nonlinear, integrated in PENLAB (and PENNON)

Postdoctoral research position in Birmingham
(sponsored by NAG)

- development of NL-SOCP algorithm (compatible with PENNON algorithm)
- implementation in PENNON
- 24 months
- start: THIS AUTUMN
- if interested, contact me

What's missing?

SOCP (Second-Order Conic Programming) - nonlinear, integrated in PENLAB (and PENNON)

Postdoctoral research position in Birmingham
(sponsored by NAG)

- development of NL-SOCP algorithm (compatible with PENNON algorithm)
- implementation in PENNON
- 24 months
- start: THIS AUTUMN
- if interested, contact me