# PENNON — New Generation

Michal Kočvara

School of Mathematics, The University of Birmingham

in cooperation with

M. Stingl, Institute of Applied Mathematics II
Friedrich-Alexander University of Erlangen-Nürnberg

CLAPDE
Durham July 14–24, 2008

# PENNON collection

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

NEW

- PENNON (NLP + SDP) extended AMPL, MATLAB

# 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 problem

Here

- $x \in \mathbb{R}^n$ is the vector variable
- $Y_1 \in \mathbb{S}^{p_1}, \ldots, Y_k \in \mathbb{S}^{p_k}$ are the matrix variables, $k$ symmetric matrices of dimensions $p_1 \times p_1, \ldots, p_k \times p_k$
- we denote $Y = (Y_1, \ldots, Y_k)$
- $f$, $g_i$ and $h_i$ are $C^2$ functions from $\mathbb{R}^n \times \mathbb{S}^{p_1} \times \ldots \times \mathbb{S}^{p_k}$ to $\mathbb{R}$
- $\underline{\lambda}_i$ and $\overline{\lambda}_i$ are the lower and upper bounds, respectively, on the eigenvalues of $Y_i$, $i = 1, \ldots, k$

# The problem

Any nonlinear SDP problem can be furmulated as NLP-SDP, using slack variables and (NLP) equality constraints:

$$g(X) \succeq 0$$

write as

$$\begin{aligned} &g(X) = S \quad \text{element-wise} \\ &S \succeq 0 \end{aligned}$$

# The algorithm

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

$$g_i(x) \le 0 \iff p_i \varphi_g(g_i(x)/p_i) \le 0, \quad i = 1, \dots, m$$
$$Z \preceq 0 \iff \Phi_P(Z) \preceq 0, \quad Z \in \mathbb{S}^p .$$

Augmented Lagrangian of (NLP-SDP):

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

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:

(i) Find $x^{k+1}$ and $Y^{k+1}$ 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) $p_i^{k+1} < p_i^k,\ i = 1, \ldots, m_g$

$P^{k+1} < P^k$ .

# Interfaces

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

- Matlab interface
- AMPL interface

# Matlab interface

User provides six MATLAB functions:

`f` ...evaluates the objective function

`df` ...evaluates the gradient of objective function

`hf` ...evaluates the Hessian of objective function

`g` ...evaluates the constraints

`dg` ...evaluates the gradient of constraints

`hg` ...evaluates the Hessian of constraints

# Matlab interface

**Matrix variables are treated as vectors**, using the function svec : $\mathbb{S}^m \to \mathbb{R}^{(m+1)m/2}$ defined by

$$\text{svec}\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ & a_{22} & \dots & a_{2m} \\ & & \ddots & \vdots \\ sym & & & a_{mm} \end{pmatrix} = (a_{11}, a_{12}, a_{22}, \dots, a_{1m}, a_{2m}, a_{mm})^T$$

# Matlab interface

**Matrix variables are treated as vectors**, using the function $\text{svec} : \mathbb{S}^m \to \mathbb{R}^{(m+1)m/2}$ defined by

$$\text{svec} \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ & a_{22} & \dots & a_{2m} \\ & & \ddots & \vdots \\ sym & & & a_{mm} \end{pmatrix}$$

$$= (a_{11}, a_{12}, a_{22}, \dots, a_{1m}, a_{2m}, a_{mm})^T$$

Keep a specific order of variables, to recognize which are matrices and which vectors. Add lower/upper bounds on matrix eigenvalues.

Sparse matrices available, sparsity maintained in the user defined functions.

# AMPL interface

AMPL does not support SDP variables and constraints. Use the same trick:
**Matrix variables are treated as vectors**, using the function $\text{svec} : \mathbb{S}^m \to \mathbb{R}^{(m+1)m/2}$ defined by

$$\text{svec}\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ & a_{22} & \dots & a_{2m} \\ & & \ddots & \vdots \\ sym & & & a_{mm} \end{pmatrix} = (a_{11}, a_{12}, a_{22}, \dots, a_{1m}, a_{2m}, a_{mm})^T$$

Need additional input file specifying the matrix sizes and lower/upper eigvalue bounds.

# Example NLPSDP

Example in matrix variable $X \in \mathbb{S}^3$:

$$\min_X \sum_{i,j=1}^{3} (X_{ij} - H_{ij})^2$$

subject to

$$\operatorname{Tr} X = 6$$

$$X \succeq 0$$

where

$$H = \begin{pmatrix} 2.2 & -1.1 & 0 \\ -1.1 & 1.9 & -1.15 \\ 0 & -1.15 & 2.1 \end{pmatrix}$$

Treat the matrix variable as a sparse matrix.

# Example NLPSDP — AMPL

`nlpsdp.mod`

```
var x{1..5} default 0;
param h{1..5};

minimize Obj: sum{i in 1..5} (x[i]-h[i])^2;
subject to
   l1:
    x[1]+x[3]+x[5] = 6;

data;
param h:=
1 2.2 2 -1.1 3 1.9 4 -1.15 5 2.1;
```

# Example NLPSDP — AMPL

`nlpsdp.sdp`

```
# Nr. of sdp blocks
    1
# Nr. of non-lin. sdp blocks
    0
# Nr. of lin. sdp blocks
    1
# Block sizes
    3
# lower eigenvalue bounds
        0.
# upper eigenvalue bounds
    1.0E38
# Constraint types
    0
# nonzeroes per block
    5
```

# Example NLPSDP — Matlab

`f.m`

```
function [fx] = f(x)
h = [2.2; -1.1; 1.9; -1.1; 2.1]./6;
x=reshape(x,length(x),[]);
fx = (x-h)'*(x-h);
```

`g.m`

```
function [gx] = g(i, x)
gx = x(1)+x(3)+x(5)-1.0;
```

# Example NLPSDP — Matlab

```
n = 5;
Infinity = 1.0E38;
pen.nvars = n;
pen.nlin = 1;
pen.nconstr = 1;
pen.nsdp = 1;
pen.blks = [3];
...
```

# Example NLPSDP — Matlab

```
...
pen.nnz_gradient = n;
pen.nnz_hessian = n;
pen.lbv = -Infinity.*ones(n,1);
pen.ubv = Infinity.*ones(n,1);
pen.lbc = [0];
pen.ubc = [0];
pen.lbmv = [0];
pen.ubmv = [Infinity];
pen.mtype = [0];
pen.mnzs = [5];
pen.mrow = [0;0;1;1;2];
pen.mcol = [0;1;1;2;2];
...
```

# Example NLPSDP — Matlab

```
...
pen.xinit=[1;0;1;0;1];
pen.my_f = 'f';
pen.my_f_gradient = 'df';
pen.my_f_hessian = 'hf';
pen.my_g = 'g';
pen.my_g_gradient = 'dg';
pen.my_g_hessian = 'hg';
pen.ioptions = [100 100 2 0 0 0 1 0 0 1 0 0 0 -1 0
pen.doptions = [1.0E-2 1.0E0 1.0E-0 1.0E-2 5.0E-1 5
                  1.0E-6 1.0E-12 1.0e-7 0.05 1.0 1.

[w1,w2]=pennonm(pen);
```

# 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

For

$$H_{\text{ext}} = \begin{pmatrix} 1 & -0.44 & -0.20 & 0.81 & -0.46 & -0.05 \\ -0.44 & 1 & 0.87 & -0.38 & 0.81 & -0.58 \\ -0.20 & .87 & 1 & -0.17 & 0.65 & -0.56 \\ 0.81 & -0.38 & -0.17 & 1 & -0.37 & -0.15 \\ -0.46 & 0.81 & 0.65 & -0.37 & 1 & -0.08 \\ -0.05 & -0.58 & -0.56 & -0.15 & 0.08 & 1 \end{pmatrix}$$

the eigenvalues of the correlation matrix are

```
eigen =
    0.0000 0.1163 0.2120 0.7827 1.7132 3.1757
```

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

# Example: nearest correlation matrix

For

$$H_{\text{ext}} = \begin{pmatrix} 1 & -0.44 & -0.20 & 0.81 & -0.46 & -0.05 \\ -0.44 & 1 & 0.87 & -0.38 & 0.81 & -0.58 \\ -0.20 & .87 & 1 & -0.17 & 0.65 & -0.56 \\ 0.81 & -0.38 & -0.17 & 1 & -0.37 & -0.15 \\ -0.46 & 0.81 & 0.65 & -0.37 & 1 & -0.08 \\ -0.05 & -0.58 & -0.56 & -0.15 & 0.08 & 1 \end{pmatrix}$$

the eigenvalues of the correlation matrix are

```
eigen =
    0.0000 0.1163 0.2120 0.7827 1.7132 3.1757
```

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

# Example: nearest correlation matrix

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

Add a new variable $z \in \mathbb{R}$ and use the transformation

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

together with

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

The new problem:

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

subject to

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

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

## Example: nearest correlation matrix

cond.mod

```
param h{1..21};
set indi within {1..21};
var x{1..21} default 0;
var z ;

minimize Obj: sum{i in 1..21} (z*x[i]-h[i])^2;
subject to
   b{i in 1..21}: x[i]*x[i]<=10000;
   bj:            z*z<=10000;
   l1{i in indi}:
        z*x[i] = 1;

data;
param h:=
   1  1.00  2 -0.44  3  1.00  4 -0.20  5  0.87  6
   8 -0.38  9 -0.17 10  1.00 11 -0.46 12  0.81 13
```

# Example: nearest correlation matrix

```
cond.sdp
# Nr. of sdp blocks
        1
# Nr. of non-lin. sdp blocks
        1
# Nr. of lin. sdp blocks
        0
# Block sizes
        6
# lower eigenvalue bounds
        1.0
# upper eigenvalue bounds
        10.
# Constraint types
        0
# nonzeroes per block
        21
```

# Example: Approximation by nonnegative splines

Let $f : [0,1] \to \mathbb{R}$. Given its (noisy) function values $b_i$, $i = 1, \dots, n$ at points $t_i \in (0,1)$.

Find a smooth approximation of $f$ by a cubic spline:

$$P(t) = P^{(i)}(t) = \sum_{k=1}^{3} P_k^{(i)}(t - a_{i-1})^k$$

for a point $t \in [a_{i-1}, a_i]$, where $0 = a_0 < a_1 < \dots < a_m = 1$ are the knots and $P_k^{(i)} (i = 1, \dots, m,\ k = 0, 1, 2, 3)$ the coefficients of the spline.

Spline property: for $i = 1, \dots, m-1$

$$P_0^{(i+1)} - P_0^{(i)} - P_1^{(i)}(a_i - a_{i-1}) - P_2^{(i)}(a_i - a_{i-1})^2 - P_3^{(i)}(a_i - a_{i-1})^3 = 0 \quad (2)$$

$$P_1^{(i+1)} - P_1^{(i)} - 2P_2^{(i)}(a_i - a_{i-1}) - 3P_3^{(i)}(a_i - a_{i-1})^2 = 0 \quad (3)$$

$$2P_2^{(i+1)} - 2P_2^{(i)} - 6P_3^{(i)}(a_i - a_{i-1}) = 0\,. \quad (4)$$

# Example: Approximation by nonnegative splines

The function $f$ will be approximated by $P$ in the least square sense: minimize

$$\sum_{j=1}^{n} (P(t_j) - b_j)^2$$

subject to (2),(3),(4).

Now, $f$ is assumed to be nonnegative, so $P \geq 0$ is required.

# Example: Approximation by nonnegative splines

de Boor and Daniel '74: while approximation of a nonnegative function by nonnegative splines of order $k$ gives errors of order $h^k$, approximation by a subclass of nonnegative splines of order $k$ consisting of all those whose $B$-spline coefficients are nonnegative may yield only errors of order $h^2$.

Nesterov 2000: $P^{(i)}(t)$ nonnegative $\Leftrightarrow$ there exist two symmetric matrices

$$X^{(i)} = \begin{pmatrix} x_i & y_i \\ y_i & z_i \end{pmatrix}, \qquad S^{(i)} = \begin{pmatrix} s_i & v_i \\ v_i & w_i \end{pmatrix}$$

such that

$$P_0^{(i)} = (a_i - a_{i-1})s_i \tag{5}$$

$$P_1^{(i)} = x_i - s_i + 2(a_i - a_{i-1})v_i \tag{6}$$

$$P_2^{(i)} = 2y_i - 2v_i + (a_i - a_{i-1})w_i \tag{7}$$

$$P_3^{(i)} = z_i - w_i \tag{8}$$

$$X^{(i)} \succeq 0, \quad S^{(i)} \succeq 0\,. \tag{9}$$

# Example: Approximation by nonnegative splines

We want to solve an NLP-SDP problem

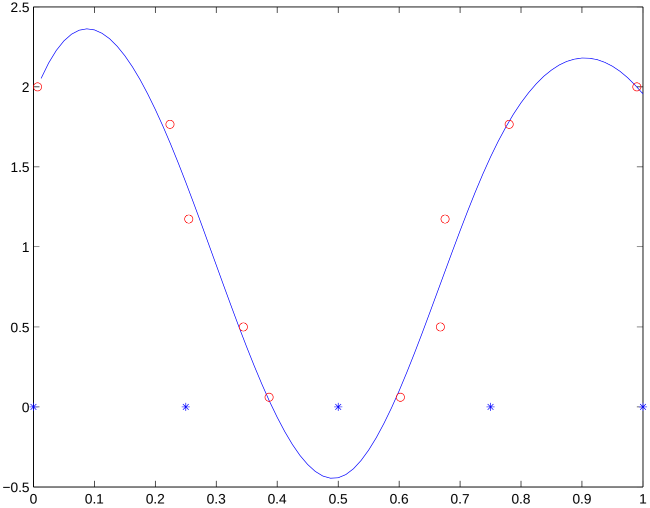
$$\min_{\substack{P_k^{(i)} \in \mathbb{R} \\ i=1,\ldots,m,\ k=0,1,2,3}} \sum_{j=1}^{n} (P(t_j) - b_j)^2 \tag{10}$$

subject to

$$(2), (3), (4), \quad i = 1, \ldots, m$$

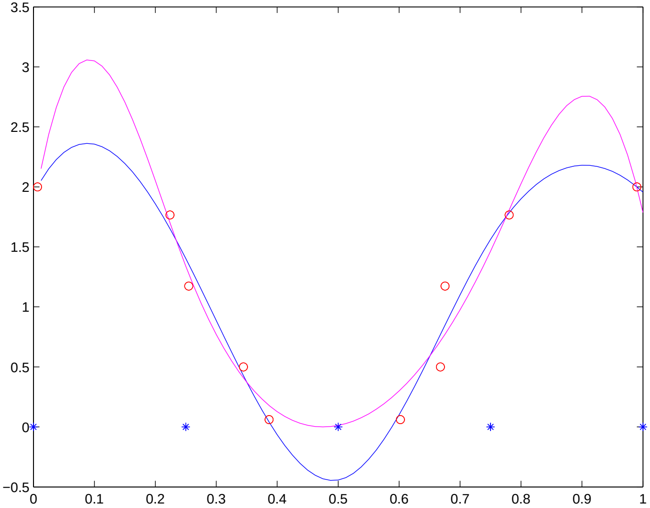
$$(5) - (9), \quad i = 1, \ldots, m$$

# Example: Approximation by nonnegative splines




err = 0.2350

# Example: Approximation by nonnegative splines




err = 0.2350 err = 0.3085

# Example: Approximation by nonnegative splines

Example, $n = 500, m = 7$, noisy data:

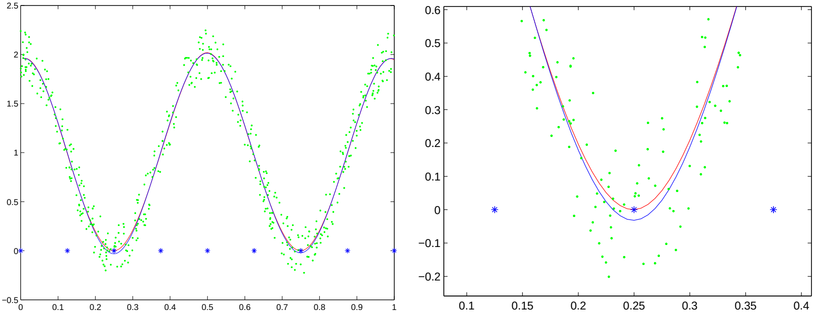



**Figure:** Approximation by nonnegative splines: noisy data given in green, optimal nonnegative spline in red and an optimal spline ignoring the nonnegativity constraint in blue. The right-hand side figure zooms on the left valley.

# Other Applications, Availability

- polynomial matrix inequalities
- financial mathematics
- structural optimization with matrix variables and nonlinear matrix constraints
- approximation by nonnegative splines
- approximation of arrival rate function of a non-homogeneous Poisson process
- sensor network localization
- optimal quasi-Newton matrix

Many other applications. . . . . . any hint welcome

Free academic version of the code available
Free downloadable MATLAB version available soon