Convex Optimization in Sensor Network Localization and Multitask Learning

Ting Kei Pong
Mathematics, University of Washington
Seattle

Thesis Defense
May 2011

(Joint work with João Gouveia, Shuiwang Ji, Paul Tseng, Jieping Ye)

Sensor Network Localization

Basic Problem:

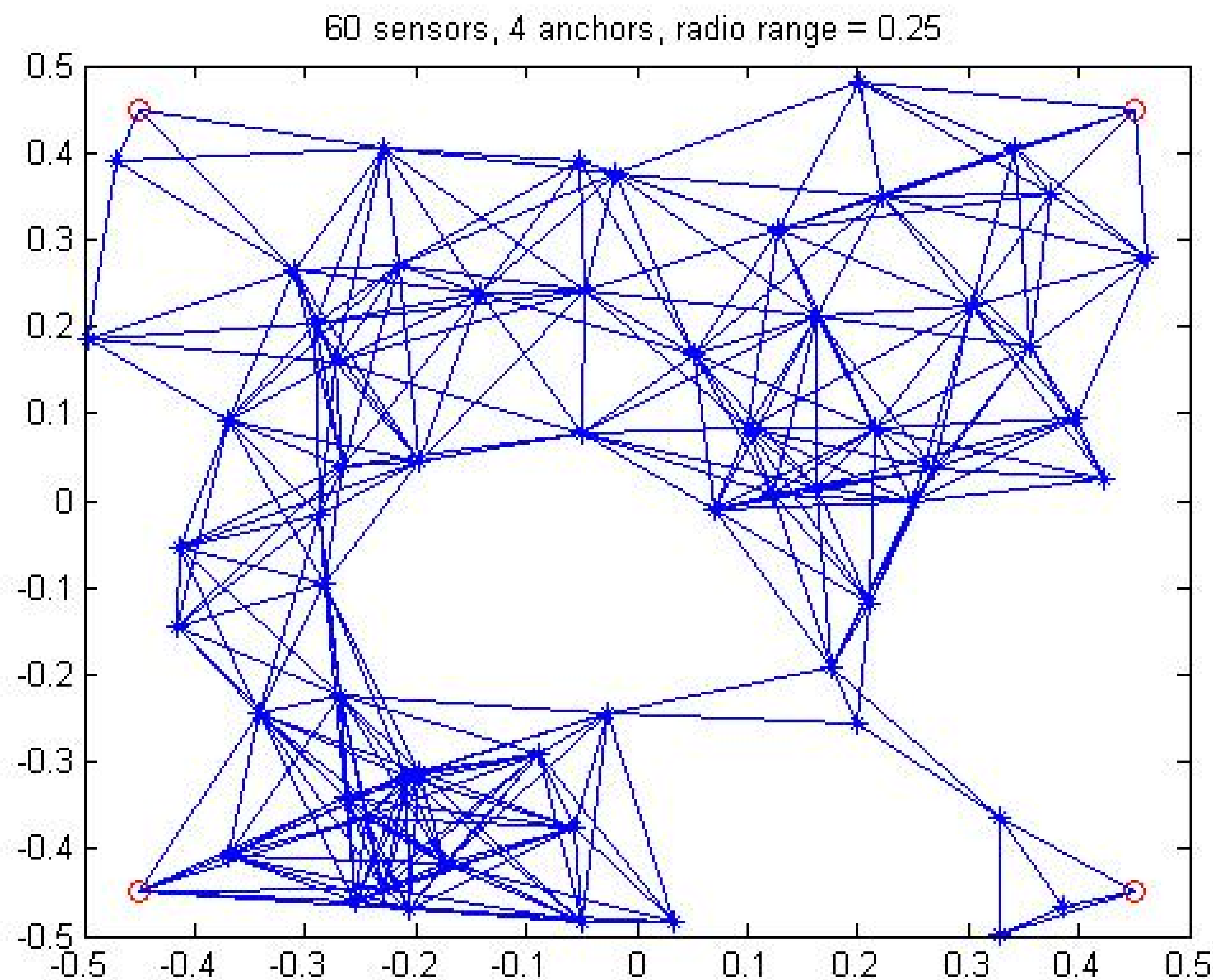
- n pts $\underbrace{x_1, ..., x_m}_{\text{sensors}}, \underbrace{x_{m+1}, ..., x_n}_{\text{anchors}}$ in $\Re^2$.

- Know last $n - m$ pts ('anchors') $x_{m+1}, ..., x_n$ and Eucl. dist. estimate for some pairs of 'neighboring' pts (i.e. within 'radio range')

$$d_{ij} \geq 0 \quad \forall (i,j) \in \mathcal{A},$$

  with $\mathcal{A} \subseteq \{(i,j) : 1 \leq i < j \leq n\}$.

- Estimate the first m pts ('sensors') $x_1, ..., x_m$.


60 sensors, 4 anchors, radio range = 0.25
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
-0.5
-0.4
-0.3
-0.2
-0.1
0
0.1
0.2
0.3
0.4
0.5

Optimization Problem Formulation

$$v_{\mathrm{p}} \;:=\; \min_{x_1,\ldots,x_m} \;\; \sum_{(i,j)\in\mathcal{A}} \left| \|x_i - x_j\|^2 - d_{ij}^2 \right|.$$

Optimization Problem Formulation

$$\upsilon_{\mathrm{p}} \ := \min_{x_1,\ldots,x_m} \sum_{(i,j)\in\mathcal{A}} \left| \|x_i - x_j\|^2 - d_{ij}^2 \right| .$$

- Objective function is nonconvex. m can be large ($m > 1000$).
- Problem is NP-hard (reduction from PARTITION).

Optimization Problem Formulation

$$v_{\mathrm{p}} := \min_{x_1,\ldots,x_m} \sum_{(i,j)\in\mathcal{A}} \left| \|x_i - x_j\|^2 - d_{ij}^2 \right| .$$

- Objective function is nonconvex. m can be large ($m > 1000$).
- Problem is NP-hard (reduction from PARTITION).
- Aim 1: Tractability – use a convex relaxation.
- Aim 2: Identify sensors correctly positioned by relaxation.

Equivalent Reformulation

Let $X = [x_1 \cdots x_m]$. Notice that

$$Y = X^T X \qquad \Leftrightarrow \qquad Z = \begin{bmatrix} Y & X^T \\ X & I \end{bmatrix} \succeq 0, \operatorname{rank}(Z) = 2$$

Equivalent reformulation:

$$\begin{aligned} \upsilon_{\mathrm{p}} := \quad & \min_{Z} \sum_{(i,j)\in\mathcal{A}^a} \left| y_{ii} - 2x_j^T x_i + \|x_j\|^2 - d_{ij}^2 \right| \\ & + \sum_{(i,j)\in\mathcal{A}^s} \left| y_{ii} - 2y_{ij} + y_{jj} - d_{ij}^2 \right| \\ \text{s.t.} \quad & Z = \begin{bmatrix} Y & X^T \\ X & I \end{bmatrix} \succeq 0, \qquad \operatorname{rank}(Z) = 2. \end{aligned}$$

SDP Relaxation

Let $X = [x_1 \cdots x_m]$. Notice that

$$Y = X^T X \qquad \Leftrightarrow \qquad Z = \begin{bmatrix} Y & X^T \\ X & I \end{bmatrix} \succeq 0, \operatorname{rank}(Z) = 2$$

SDP relaxation (Biswas, Ye '03):

$$\begin{aligned} \upsilon_{\rm sdp} := \quad & \min_Z \sum_{(i,j)\in\mathcal{A}^a} \left| y_{ii} - 2x_j^T x_i + \|x_j\|^2 - d_{ij}^2 \right| \\ & \quad + \sum_{(i,j)\in\mathcal{A}^s} \left| y_{ii} - 2y_{ij} + y_{jj} - d_{ij}^2 \right| \\ & \text{s.t.} \quad Z = \begin{bmatrix} Y & X^T \\ X & I \end{bmatrix} \succeq 0. \end{aligned}$$

ESDP Relaxation

ESDP relaxation (Wang, Zheng, Boyd, Ye '08):

$$
\begin{aligned}
v_{\text{esdp}} := \quad & \min_{Z} \sum_{(i,j)\in\mathcal{A}^a} \left| y_{ii} - 2x_j^T x_i + \|x_j\|^2 - d_{ij}^2 \right| \\
& \quad + \sum_{(i,j)\in\mathcal{A}^s} \left| y_{ii} - 2y_{ij} + y_{jj} - d_{ij}^2 \right| \\
\text{s.t.} \quad & Z = \begin{bmatrix} Y & X^T \\ X & I \end{bmatrix} \\
& \begin{bmatrix} y_{ii} & y_{ij} & x_i^T \\ y_{ij} & y_{jj} & x_j^T \\ x_i & x_j & I \end{bmatrix} \succeq 0 \quad \forall (i,j) \in \mathcal{A}^s.
\end{aligned}
$$

Alternative Problem Formulation

$$\min_{x_1,\ldots,x_m} \quad \sum_{(i,j)\in\mathcal{A}} \left(\|x_i - x_j\|^2 - d_{ij}^2\right)^2 .$$

- Objective is a nonconvex degree 4 polynomial;
- Use convex relaxation – sum of squares technique.

Sparse-SOS Relaxation

Idea: Linearization.

Sparse-SOS Relaxation

Idea: Linearization.

For $(i,j) \in \mathcal{A}^s$,

$$\beta_{ij}^4 := \{ \; 1 \quad x_i^1 \quad x_i^2 \quad x_j^1 \quad x_j^2 \quad (x_i^1)^2 \quad \cdots \quad (x_j^2)^2 \quad x_i^1 x_i^2 \quad \cdots \quad x_i^1 x_i^2 x_j^1 x_j^2 \; \}$$

$$\{ \; 1 \quad u_{x_i^1} \quad u_{x_i^2} \quad u_{x_j^1} \quad u_{x_j^2} \quad u_{(x_i^1)^2} \quad \cdots \quad u_{(x_j^2)^2} \quad u_{x_i^1 x_i^2} \quad \cdots \quad u_{x_i^1 x_i^2 x_j^1 x_j^2} \; \}$$

Sparse-SOS Relaxation

Idea: Linearization.

For $(i,j) \in \mathcal{A}^s$,

$$\beta_{ij}^4 := \{ \ 1 \quad x_i^1 \quad x_i^2 \quad x_j^1 \quad x_j^2 \quad (x_i^1)^2 \quad \cdots \quad (x_j^2)^2 \quad x_i^1 x_i^2 \quad \cdots \quad x_i^1 x_i^2 x_j^1 x_j^2 \ \}$$

$$\{ \ 1 \quad u_{x_i^1} \quad u_{x_i^2} \quad u_{x_j^1} \quad u_{x_j^2} \quad u_{(x_i^1)^2} \quad \cdots \quad u_{(x_j^2)^2} \quad u_{x_i^1 x_i^2} \quad \cdots \quad u_{x_i^1 x_i^2 x_j^1 x_j^2} \ \}$$

For $(i,j) \in \mathcal{A}^a$,

$$\beta_{ij}^4 := \{ \ 1 \quad x_i^1 \quad x_i^2 \quad (x_i^1)^2 \quad (x_i^2)^2 \quad x_i^1 x_i^2 \quad \cdots \quad (x_i^1)^2 (x_i^2)^2 \ \}$$

$$\{ \ 1 \quad u_{x_i^1} \quad u_{x_i^2} \quad u_{(x_i^1)^2} \quad u_{(x_i^2)^2} \quad u_{x_i^1 x_i^2} \quad \cdots \quad u_{(x_i^1)^2 (x_i^2)^2} \ \}$$

Moment Matrix

Idea: Linearization of the outer product matrix by monomials up to degree 2, β^2_{ij}. Here shows $M_{\beta^2_{ij}}(u)$ for $(i,j) \in \mathcal{A}^a$:

$$
\begin{array}{c} \\ 1 \\ x_i^1 \\ x_i^2 \\ (x_i^1)^2 \\ x_i^1x_i^2 \\ (x_i^2)^2 \end{array}
\begin{array}{c}
\begin{array}{cccccc} 1 & x_i^1 & x_i^2 & (x_i^1)^2 & x_i^1x_i^2 & (x_i^2)^2 \end{array} \\
\begin{bmatrix}
1 & u_{x_i^1} & u_{x_i^2} & u_{(x_i^1)^2} & u_{x_i^1x_i^2} & u_{(x_i^2)^2} \\
u_{x_i^1} & u_{(x_i^1)^2} & u_{x_i^1x_i^2} & u_{(x_i^1)^3} & u_{(x_i^1)^2x_i^2} & u_{x_i^1(x_i^2)^2} \\
u_{x_i^2} & u_{x_i^1x_i^2} & u_{(x_i^2)^2} & u_{(x_i^1)^2x_i^2} & u_{x_i^1(x_i^2)^2} & u_{(x_i^2)^3} \\
u_{(x_i^1)^2} & u_{(x_i^1)^3} & u_{(x_i^1)^2x_i^2} & u_{(x_i^1)^4} & u_{(x_i^1)^3x_i^2} & u_{(x_i^1)^2(x_i^2)^2} \\
u_{x_i^1x_i^2} & u_{(x_i^1)^2x_i^2} & u_{x_i^1(x_i^2)^2} & u_{(x_i^1)^3x_i^2} & u_{(x_i^1)^2(x_i^2)^2} & u_{x_i^1(x_i^2)^3} \\
u_{(x_i^2)^2} & u_{x_i^1(x_i^2)^2} & u_{(x_i^2)^3} & u_{(x_i^1)^2(x_i^2)^2} & u_{x_i^1(x_i^2)^3} & u_{(x_i^2)^4}
\end{bmatrix}
\end{array}
$$

Sparse-SOS Relaxation

Sparse-SOS relaxation (Nie '09):

$$\begin{aligned} v_{\text{spsos}} := \quad & \min_{u} \sum_{(i,j)\in\mathcal{A}} \sum_{\sigma\in\beta_{ij}^4} p_\sigma^{ij} u_\sigma \\ & \text{s.t. } \; M_{\beta_{ij}^2}(u) \succeq 0 \quad \forall (i,j) \in \mathcal{A}^s, \end{aligned}$$

where

$$(\|x_i - x_j\|^2 - d_{ij}^2)^2 =: \sum_{\sigma\in\beta_{ij}^4} p_\sigma^{ij} \sigma(x) \quad \forall (i,j) \in \mathcal{A}.$$

Properties of Relaxations

Assume that every connected component contains an anchor. Let $\mathrm{pos}(\cdot)$ denote the set of sensor positions ($\subseteq \Re^2$) obtained by solving the relaxation $(\cdot)$.

Fact 1:

- pos(ESDP), pos(sSOS) and pos(SDP) are compact convex sets.

- When $d_{ij} = \|x_i^{\mathrm{true}} - x_j^{\mathrm{true}}\|$ for all $(i,j) \in \mathcal{A}$ (noiseless case),

$$\mathrm{pos}(\mathrm{SDP}) \subseteq \mathrm{pos}(\mathrm{ESDP}),$$ (Wang et al. '08)

$$\mathrm{pos}(\mathrm{sSOS}) \subseteq \mathrm{pos}(\mathrm{ESDP}).$$ (Gouveia, P '10)

Fact 2 (local exactness):

- Define $\mathrm{tr}_i(Z) := y_{ii} - \|x_i\|^2$ for SDP and ESDP relaxations, and $\mathrm{Tr}_i(u) := u_{(x_i^1)^2} + u_{(x_i^2)^2} - (u_{x_i^1})^2 - (u_{x_i^2})^2$ for the sSOS relaxation.
- In the noiseless case,

Fact 2 (local exactness):

- Define $\mathrm{tr}_i(Z) := y_{ii} - \|x_i\|^2$ for SDP and ESDP relaxations, and $\mathrm{Tr}_i(u) := u_{(x_i^1)^2} + u_{(x_i^2)^2} - (u_{x_i^1})^2 - (u_{x_i^2})^2$ for the sSOS relaxation.

- In the noiseless case,

  ⋆ If $\mathrm{tr}_i(Z) = 0$ for some $Z \in \mathrm{ri}(\mathrm{Sol}(\mathrm{SDP}))$, then x_i is invariant over pos(SDP) (Tseng '07).

Fact 2 (local exactness):

- Define $\mathrm{tr}_i(Z) := y_{ii} - \|x_i\|^2$ for SDP and ESDP relaxations, and $\mathrm{Tr}_i(u) := u_{(x_i^1)^2} + u_{(x_i^2)^2} - (u_{x_i^1})^2 - (u_{x_i^2})^2$ for the sSOS relaxation.

- In the noiseless case,

  ⋆ If $\mathrm{tr}_i(Z) = 0$ for some $Z \in \mathrm{ri}(\mathrm{Sol}(\mathrm{SDP}))$, then x_i is invariant over pos(SDP) (Tseng '07).

  ⋆ If $\mathrm{tr}_i(Z) = 0$ for some $Z \in \mathrm{ri}(\mathrm{Sol}(\mathrm{ESDP}))$, then x_i is invariant over pos(ESDP) (Wang et al. '08); the converse also holds (P, Tseng '10).

Fact 2 (local exactness):

- Define $\mathrm{tr}_i(Z) := y_{ii} - \|x_i\|^2$ for SDP and ESDP relaxations, and $\mathrm{Tr}_i(u) := u_{(x_i^1)^2} + u_{(x_i^2)^2} - (u_{x_i^1})^2 - (u_{x_i^2})^2$ for the sSOS relaxation.

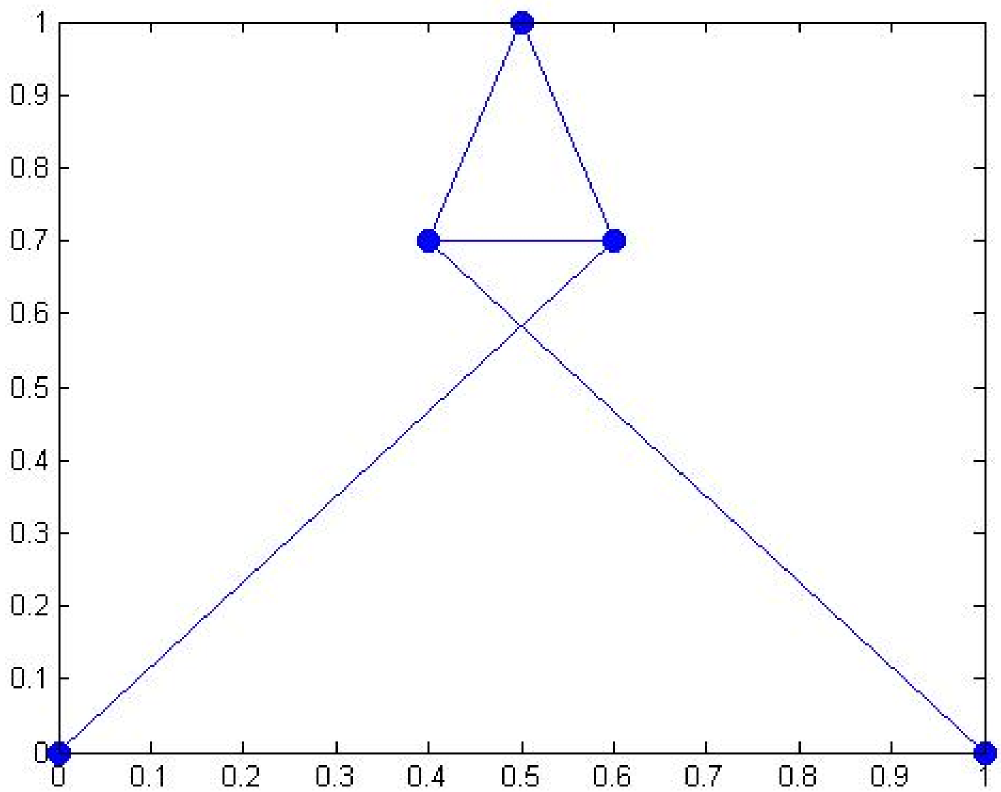
- In the noiseless case,

  ⋆ If $\mathrm{tr}_i(Z) = 0$ for some $Z \in \mathrm{ri}(\mathrm{Sol}(\mathrm{SDP}))$, then x_i is invariant over pos(SDP) (Tseng '07).

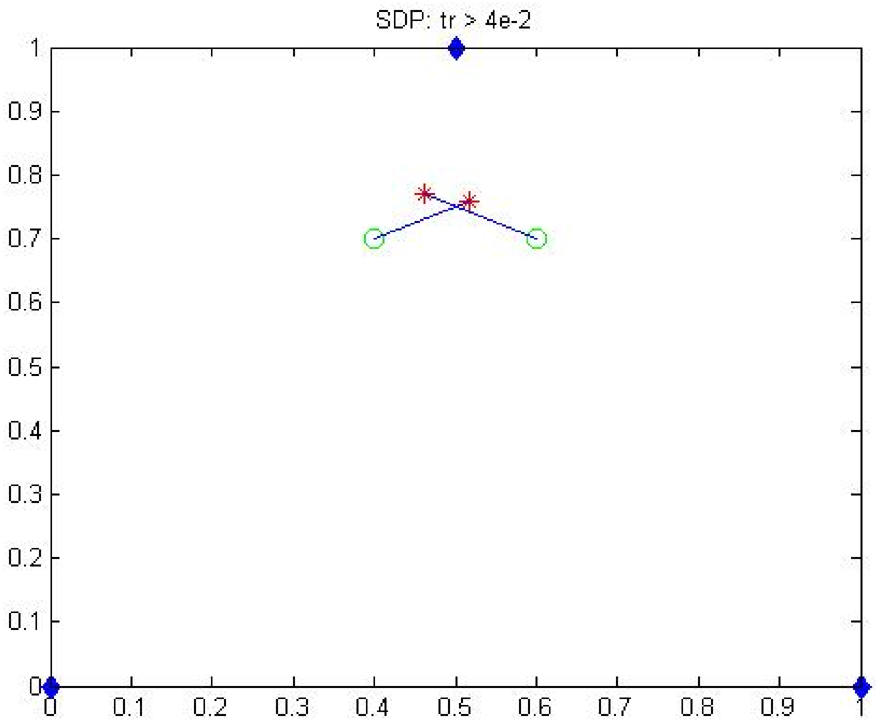
  ⋆ If $\mathrm{tr}_i(Z) = 0$ for some $Z \in \mathrm{ri}(\mathrm{Sol}(\mathrm{ESDP}))$, then x_i is invariant over pos(ESDP) (Wang et al. '08); the converse also holds (P, Tseng '10).

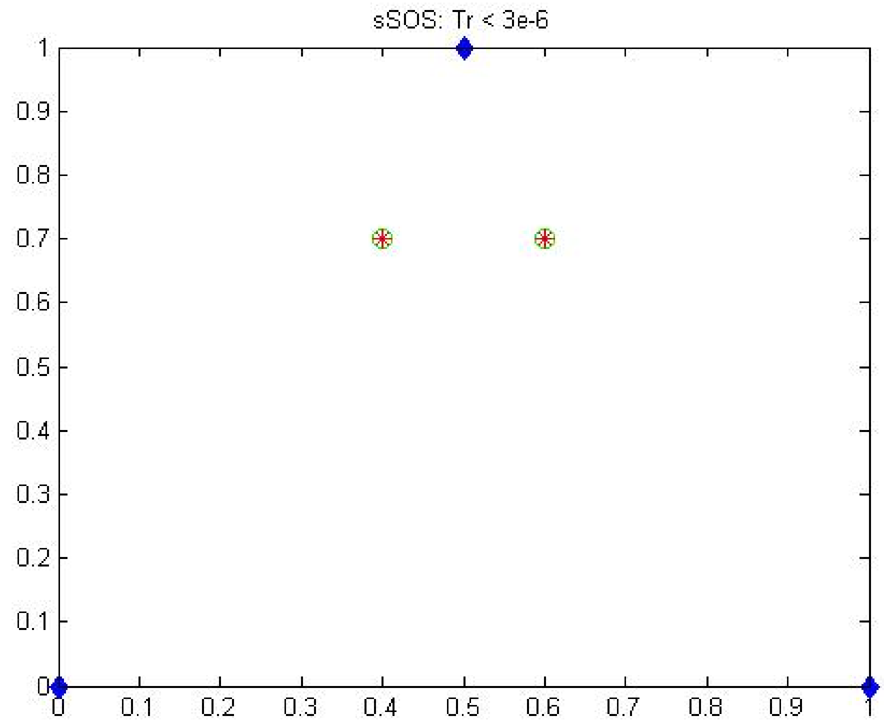
  ⋆ If $\mathrm{Tr}_i(u) = 0$ for some $u \in \mathrm{ri}(\mathrm{Sol}(\mathrm{sSOS}))$, then $(u_{x_i^1}, u_{x_i^2})^T$ is invariant over pos(sSOS) (Gouveia, P '10).

Numerical Example: SDP Vs Sparse-SOS



Numerical Example: SDP Vs Sparse-SOS





In practice, there are measurement noises:

$$d_{ij}^2 = \|x_i^{\text{true}} - x_j^{\text{true}}\|^2 + \delta_{ij} \quad \forall (i,j) \in \mathcal{A}.$$

What can we say in this case? (P, Tseng '10)

In practice, there are measurement noises:

$$d_{ij}^2 = \|x_i^{\mathrm{true}} - x_j^{\mathrm{true}}\|^2 + \delta_{ij} \quad \forall (i,j) \in \mathcal{A}.$$

What can we say in this case? (P, Tseng '10)

- Individual trace test fails for ESDP relaxation.
- ρ-ESDP is proposed. For a particular solution Z^*,

  $$x_i^* \approx x_i^{\mathrm{true}} \Leftrightarrow \mathrm{tr}_i(Z^*) \approx 0,$$

  when noise is small.
- A fast distributed algorithm is proposed for solving Z^*.

Multi-task Learning

- Given training data set $\{(x_1, y_1^l), ..., (x_p, y_p^l)\} \subset \mathbb{R}^n \times \{-1, 1\}$, $l = 1, ..., m$ (m = number of tasks).

- Find linear predictors $w_l^T x$ by

$$\min \sum_{l=1}^{m} \left(\sum_{i=1}^{p} \|w_l^T x_i - y_i^l\|^2 \right) + \mu \Omega(W),$$

where $W = [w_1 \cdots w_m]$ and Ω is a regularization term (capture relation between predictors).

Nuclear Norm Minimization

- Using nuclear norm as regularization; minimizing rank (Fazel, Hindi, Boyd 01)

$$\sum_{l=1}^{m}\left(\sum_{i=1}^{p}\|w_l^T x_i - y_i^l\|^2\right) + \mu\|W\|_*,$$

where $\|W\|_* := \sum_{i=1}^{\min\{m,n\}} \sigma_i(W)$; or

$$v := \quad \min_W p(W) := \frac{1}{2}\|AW - B\|_F^2 + \mu\|W\|_*,$$

$A \in \mathbb{R}^{p\times n}, B \in \mathbb{R}^{p\times m}, W \in \mathbb{R}^{n\times m}$.

- Typical problem dimension: $50 \leq m \leq 100$, $1000 \leq n, p \leq 3000$.

- Algorithms: IPM, first-order method...

Proximal Gradient Algorithm

- Solves

$$h^* := \min_x h(x) := f(x) + Q(x),$$

with f convex smooth, ∇f Lipschitz continuous, Q "simple" closed convex.

- Initialize $x \in \operatorname{dom} Q$, compute

$$x^{\text{new}} := \arg\min_y \{\langle \nabla f(x), y - x\rangle + Q(y) + \frac{L_f}{2}\|y - x\|^2\}$$

- Complexity: $h(x_k) - h^* = O(\frac{L_f}{k})$.

1st Algorithm

0. Choose any W. Set $L = L_P := \lambda_{\max}(A^T A)$. Go to step 1.

1. Compute the SVD:

$$W - \frac{1}{L}(A^T A W - A^T B) = R D S^T.$$

2. Update

$$W^{\text{new}} = R \max\left\{D - \frac{\mu}{L} I, 0\right\} S^T.$$

1st Algorithm

0. Choose any W. Set $L = L_P := \lambda_{\max}(A^T A)$. Go to step 1.

1. Compute the SVD:

$$W - \frac{1}{L}(A^T AW - A^T B) = RDS^T.$$

2. Update

$$W^{\mathrm{new}} = R \max\left\{D - \frac{\mu}{L} I, 0\right\} S^T.$$

Can we apply Nesterov's acceleration scheme?

PAPG: Primal Accelerated Proximal Gradient Method

0. Choose any W. Initialize $W_- = W$, $\theta_- = \theta = 1$. Set $L = L_P$. Go to step 1.

1. Set
$$Y = W + \left(\frac{\theta}{\theta_-} - \theta\right)(W - W_-).$$

2. Compute the SVD:
$$Y - \frac{1}{L}(A^T AY - A^T B) = RDS^T.$$

3. Update
$$W^{\text{new}} = R\max\left\{D - \frac{\mu}{L}I, 0\right\}S^T, \qquad W_-^{\text{new}} = W,$$
$$\theta^{\text{new}} = \frac{\sqrt{\theta^4 + 4\theta^2} - \theta^2}{2}, \qquad \theta_-^{\text{new}} = \theta.$$

About PAPG:

- Complexity: $p(W_k) - v = O(\frac{L_P}{k^2})$.
- $L_P = \lambda_{\max}(A^T A)$. If $A^T A$ has large eigenvalues, the algorithm is slow.

About PAPG:

- Complexity: $p(W_k) - v = O(\frac{L_P}{k^2})$.
- $L_P = \lambda_{\max}(A^T A)$. If $A^T A$ has large eigenvalues, the algorithm is slow.

Consider dual problem instead?

About PAPG:

- Complexity: $p(W_k) - v = O(\frac{L_P}{k^2})$.
- $L_P = \lambda_{\max}(A^T A)$. If $A^T A$ has large eigenvalues, the algorithm is slow.

Consider dual problem instead?

Fact 1 (P, Tseng, Ji, Ye '09): The problem can be reduced to

$$v = \quad \min_{\tilde{W}} \frac{1}{2}\|\tilde{A}\tilde{W} - R^T B\|_F^2 + \mu\|\tilde{W}\|_*,$$

where $A = R\begin{bmatrix}\tilde{A} & 0\end{bmatrix} S^T$, $\tilde{A} \in \mathbb{R}^{p\times r}$, $R^T R = I$ and $S^T S = I$.

Derivation of the Dual Problem

Let $\mathrm{rank}(A) = n$. We have

$$\begin{aligned}
v &= \min_W \frac{1}{2}\|AW - B\|_F^2 + \mu\|W\|_* \\
&= \min_W \left(\frac{1}{2}\|AW - B\|_F^2 + \max_{\Lambda^T\Lambda \preceq \mu^2 I}\langle -\Lambda, W\rangle\right) \\
&= \min_W \max_{\Lambda^T\Lambda \preceq \mu^2 I}\left(\frac{1}{2}\|AW - B\|_F^2 - \langle \Lambda, W\rangle\right) \\
&= \max_{\Lambda^T\Lambda \preceq \mu^2 I} \min_W \left(\frac{1}{2}\|AW - B\|_F^2 - \langle \Lambda, W\rangle\right) \\
&= -\min_{\Lambda^T\Lambda \preceq \mu^2 I} \frac{1}{2}\langle \Lambda, (A^TA)^{-1}\Lambda\rangle + \langle (A^TA)^{-1}A^TB, \Lambda\rangle + \text{constant}
\end{aligned}$$

$$p(W) := \frac{1}{2}\|AW - B\|_F^2 + \mu\|W\|_*$$

$$d(\Lambda) := \frac{1}{2}\langle\Lambda, (A^T A)^{-1}\Lambda\rangle + \langle(A^T A)^{-1}A^T B, \Lambda\rangle + \text{constant}.$$

Then $p(W) + d(\Lambda) \geq 0$ for any W and $\Lambda^T\Lambda \preceq \mu^2 I$.

$$p(W) := \frac{1}{2}\|AW - B\|_F^2 + \mu\|W\|_*$$
$$d(\Lambda) := \frac{1}{2}\langle\Lambda, (A^TA)^{-1}\Lambda\rangle + \langle(A^TA)^{-1}A^TB, \Lambda\rangle + \text{constant}.$$

Then $p(W) + d(\Lambda) \geq 0$ for any W and $\Lambda^T\Lambda \preceq \mu^2 I$.

Fact 2 (P, Tseng, Ji, Ye ’09):
If W^* is the primal optimal solution, then the dual optimal solution is

$$\Lambda^* = A^T(AW^* - B).$$

If Λ^* is the dual optimal solution, then the primal optimal solution is

$$W^* = (A^TA)^{-1}(\Lambda^* + A^TB).$$

DAGP: Dual Accelerated Gradient Projection Method

0. Choose any Λ satisfying $\Lambda^T \Lambda \preceq \mu^2 I$. Initialize $\Lambda_- = \Lambda$ and $\theta_- = \theta = 1$. Set $L = L_D := \frac{1}{\lambda_{\min}(A^T A)}$. Go to Step 1.

1. Set
$$\Phi = \Lambda + \left(\frac{\theta}{\theta_-} - \theta\right)(\Lambda - \Lambda_-).$$

2. Compute the SVD
$$\Phi - \frac{1}{L}(A^T A)^{-1}(\Phi + A^T B) = RDS^T.$$

3. Update
$$\Lambda^{\text{new}} = R \min\{D, \mu I\} S^T, \qquad \Lambda_-^{\text{new}} = \Lambda,$$
$$\theta^{\text{new}} = \frac{\sqrt{\theta^4 + 4\theta^2} - \theta}{2}, \qquad \theta_-^{\text{new}} = \theta.$$

4. (Termination test) Compute $W = (A^T A)^{-1}(\Lambda^{\text{new}} + A^T B)$. If

$$\frac{p(W) + d(\Lambda^{\text{new}})}{|d(\Lambda^{\text{new}})| + 1} \leq tol,$$

terminate. Else, go to Step 1.

4. (Termination test) Compute $W = (A^T A)^{-1}(\Lambda^{\text{new}} + A^T B)$. If

$$\frac{p(W) + d(\Lambda^{\text{new}})}{|d(\Lambda^{\text{new}})| + 1} \leq tol,$$

terminate. Else, go to Step 1.

Alternative termination criterion (Tseng '09): Initialize $W = 0$ and update $W^+ = (1-\theta)W + \theta(A^T A)^{-1}(\Phi + A^T B)$. Then

$$0 \leq p(W^+) + d(\Lambda^{\text{new}}) \leq \theta^2 L_D \max_{\Gamma^T\Gamma \preceq \mu^2 I} \frac{1}{2}\|\Gamma - \Lambda^{\text{init}}\|_F^2.$$

About DAGP:

- Complexity $d(\Lambda_k) - v = O(\frac{L_D}{k^2})$.
- $L_D = \frac{1}{\lambda_{\min}(A^T A)}$. If $A^T A$ has small eigenvalue, the algorithm is slow.
- Complexity bound on duality gap. No such bounds known for the primal algorithm.
- DAGP requires a reduction. PAPG does not necessarily require a reduction first.

Simulation Results

- Compare PAPG and DAGP.

- Generate A with entries uniformly in $[0,1]$, B with entries uniformly in $\{-1,1\}$.

- Terminate PAPG when
  - ⋆ $\frac{p(W^{\text{new}})+d(\Lambda)}{|d(\Lambda)|+1} \leq 0.001$ (checked every 500 iterations); or
  - ⋆ $\|W^{\text{new}} - W\|_F < 10^{-8}$ (checked every iteration).

- Terminate DAGP when
  - ⋆ $\frac{\min\{p(W), p(W^+)\}+d(\Lambda^{\text{new}})}{|d(\Lambda^{\text{new}})|+1} \leq 0.001$ (checked every 500 iterations); or
  - ⋆ $\|\Lambda^{\text{new}} - \Lambda\|_F < 10^{-8}$ (checked every iteration).

Simulation Results

$m \times n \times p$	L_P	L_D	red	μ	(PAPG)iter/cpu/gap	(DAGP)iter/cpu/gap
$50 \times 2000 \times 1500$	8e5	3e-1	9e1	100	2000/8e2/6e-4	459/3e2/5e-15
$50 \times 2000 \times 1500$	8e5	3e-1	9e1	1	max/2e3/4e-1	12/1e2/2e-13
$50 \times 2000 \times 3500$	2e6	6e-2	1e2	100	2500/2e3/3e-4	85/2e2/2e-15
$50 \times 2000 \times 3500$	2e6	6e-2	1e2	1	max/3e3/3e-3	7/2e2/3e-15
$50 \times 3000 \times 1500$	1e6	5e-2	1e2	100	3500/1e3/8e-4	81/2e2/7e-15
$50 \times 3000 \times 1500$	1e6	5e-2	1e2	1	max/2e3/4e-1	7/2e2/5e-13
$50 \times 3000 \times 3500$	3e6	6e-1	3e2	100	2500/3e3/1e-3	500/1e3/6e-16
$50 \times 3000 \times 3500$	3e6	6e-1	3e2	1	max/6e3/3e-2	10/5e2/2e-15

- Matlab codes run on an HP DL360 workstation, running RedHat Linux 3.5, Matlab 7.2. Time in seconds (CPU), relative duality gap (gap).
- Initialize PAPG at $W = 0$, DAGP at $\Lambda = 0$.
- PAPG works better when L_P is small and μ is large.
- DAGP works better when L_D is small and μ is small.

Other Work & Extensions

Other work:

- Algorithms for Optimal Experimental Design and computing Dantzig selector (with Zhaosong Lu and Yong Zhang).
- Convex reformulation and algorithm for finding minimal condition number (with Zhaosong Lu).

Ongoing/Future work:

- Algorithms for nuclear norm minimization with special linear structure (with Maryam Fazel, Defeng Sun and Paul Tseng).
- Graph structure uniquely localized by solving SOS relaxation (with João Gouveia).

Thanks for coming! ☺