Lagrangian dual interior-point method for semidefinite programs

Mitsuhiro Fukuda[†] *(mituhiro@is.titech.ac.jp)*
Masakazu Kojima[†] *(kojima@is.titech.ac.jp)*
Masayuki Shida[‡] *(shida@cc.nda.ac.jp)*

† **Tokyo Institute of Technology**
‡ **National Defense Academy of Japan**

Contents of this talk

1. Brief overview of existing methods for SDPs
2. Major difficulties in primal-dual interior-point methods for SDPs
3. A Class of Semidefinite Programs (SDPs) solved by LDIPM
4. Lagrangian Dual Interior-Point Method (LDIPM)
   — Primal-dual predictor-corrector path-following method
   - Corrector procedure
     — BFGS quasi-Newton method instead of Newton method
   - Predictor procedure
     — CG method instead of Cholesky factorization
5. Preliminary numerical experiments

Brief overview of existing methods to solve SDPs

(I) Primal-dual path-following interior-point methods

- **for general SDPs**
- **search directions: AHO, NT, H..K..M,** *etc.*
- **use of CG method,** *Nakata et al.'98, Toh et al.'00, etc.*
- **p.definite matrix completion,** *Fukuda et al.'00, Nakata et al.'01*
- **SDPT3, SeDuMi, CSDP, SDPA**

(II) Dual interior-point method, *S.Benson-Ye-Zhang'00*

(III) Spectral Bundle method, *Helmberg-Rendl'00*

(IV) Nonlinear programming formulation

- *Burer-Monteiro-Zhang'99, Vanderbei-H.Benson'00*

- **(II), (III), (IV) : effective for SDPs from comb. optim.**
- **Solving general large scale SDPs in high accuracy is challenging!**

Major difficulties in primal-dual IPMs for SDPs — 1

Large scale fully dense positive definite system of equations

$$\boldsymbol{M}^k \boldsymbol{dy} = \boldsymbol{r}^k$$

to compute a search direction $(\boldsymbol{dX}, \boldsymbol{dS}, \boldsymbol{dy})$ at each iteration k.
Here the size m of $\boldsymbol{M}^k$ = the number of constraints of an SDP to be solved;
m can be more than $200,000$.

$$\Downarrow$$

- **Use iterative methods such as CG and CR methods**
- **However, the condition number of $\boldsymbol{M}^k$ gets worse rapidly as the iterated approx. sol. $(\boldsymbol{X}^k, \boldsymbol{S}^k, \boldsymbol{y}^k)$ approaches to an opt. sol.**

$$\Downarrow$$

- **Effective preconditioner for the fully positive definite dense matrix $\boldsymbol{M}^k$ without storing $\boldsymbol{M}^k$.**

LDIPM tries to resolve this difficulty by using "the BFGS quasi-Newton matrix" as a preconditioner in the CG method.

Major difficulties in primal-dual IPMs for SDPs — 2

Primal matrix variable $\boldsymbol{X}$ becomes fully dense even when data matrices $\boldsymbol{A}_0, \boldsymbol{A}_1, \ldots, \boldsymbol{A}_m$ are sparse. The size of $\boldsymbol{X}$ can be $10,000 \times 10,000$.

But the dual matrix variable $\boldsymbol{S}$ can be sparse because

$$\boldsymbol{S} = \boldsymbol{A}_0 - \sum_{i=1}^{m} \boldsymbol{A}_i y_i.$$

$$\Downarrow$$

- **Dual interior-point methods,** *S.Benson-Ye-Zhang'00*
  — effective for SDPs from max cut and graph partition problems.

- **p.definite matrix completion,** *Fukuda et al.'00, Nakata et al.'01*
  — effective for special sparse cases.

LDIPM tries to resolve this difficulty by evaluating $\boldsymbol{X}$ only when $\boldsymbol{X}\boldsymbol{S} = \mu\boldsymbol{I}$. Instead of $\boldsymbol{X}$ itself, we store and utilize

$$\mu\boldsymbol{X}^{-1} = \boldsymbol{S} = \boldsymbol{L}\boldsymbol{L}^T = \text{ a sparse Cholesky factorization of } \boldsymbol{S}$$

A class of SDPs solved by LDIPM (Lagrangian Dual Interior-Point Method)

$$\textbf{Primal}\begin{cases}\textbf{max.} & \boldsymbol{C} \bullet \boldsymbol{X} \\ \textbf{sub.to} & \boldsymbol{A}_p \bullet \boldsymbol{X} = a_p \ (p = 1, 2, \ldots, m), \ \ \boldsymbol{I} \bullet \boldsymbol{X} = b, \boldsymbol{X} \succeq \boldsymbol{O}\end{cases} \tag{1}$$

$$\textbf{Dual}\begin{cases}\textbf{min.} & \sum_{p=1}^{m} a_p y_p + bw \\ \textbf{sub.to} & \sum_{p=1}^{m} \boldsymbol{A}_p y_p + \boldsymbol{I} w - \boldsymbol{S} = \boldsymbol{C}, \ \ \boldsymbol{S} \succeq \boldsymbol{O}\end{cases}$$

Here

$\mathcal{S}^n$ **: the space of** $n \times n$ **symmetric matrices**

$\mathbb{R}^m$ **: the** n**-dimensional Euclidean space**

$\boldsymbol{C}, \boldsymbol{A}_1, \cdots, \boldsymbol{A}_m \in \mathcal{S}^n,\ \boldsymbol{a} = (1_1, \ldots, a_m) \in \mathbb{R}^m,\ \mathbb{R} \in b > 0$ **are given data.**

$\boldsymbol{I}$ **: the** $n \times n$ **identity matrix**

$\boldsymbol{A} \bullet \boldsymbol{X}$ **: the inner product** $\sum_{p=1}^{n} \sum_{q=1}^{n} A_{pq} X_{pq}$

$\boldsymbol{X} \succeq \boldsymbol{O}$ **:** $\boldsymbol{X}$ **is a symm. positive semidefinite matrix**

$\boldsymbol{X} \in \mathcal{S}^n$ **: primal matrix variable**

$\boldsymbol{S} \in \mathcal{S}^n$ **: dual matrix variable**

Important feature of the SDP above — "the simplex constraint"

$$\boldsymbol{I} \bullet \boldsymbol{X} = b, \ X \succeq \boldsymbol{O}.$$

This is restrictive. But

- **(1) covers various SDPs,**

- **when the feasible region of an SDP to be solved is bounded and its bound is known in advance, we can transform it into (1).**

Assumption

1. $\exists \boldsymbol{X}^0 \succ \boldsymbol{O}$ **feasible for Primal SDP** (Slater c.q.)

2. $\boldsymbol{A}_p \ (p = 1, 2, \ldots, m)$ **and** $\boldsymbol{I}$ **are linearly independent.**

For any $\boldsymbol{y} \in \mathbb{R}^n$**,** $\boldsymbol{S} = \boldsymbol{I}w + \sum_{p=1}^{m} \boldsymbol{A}_p y_p - \boldsymbol{C} \succ \boldsymbol{O}$

whenever w **is sufficiently large;**

hence $(\boldsymbol{y}, w, S)$ **is an interior feasible solution of Dual.**

$\Longrightarrow$ **In LDIPM,** $\boldsymbol{y}$ **can vary over the entire space** $\mathbb{R}^m$**.**

Lagrangian Dual IPM

Given $\boldsymbol{y} \in \mathbb{R}^m$ **and** $\mu > 0$**, consider**

$$g(\boldsymbol{y}, \mu) \equiv \boxed{\widetilde{\mathbf{D}}(\boldsymbol{y}, \mu) \begin{cases} \textbf{min.} & \sum_{p=1}^m a_p y_p + bw - \mu \log \det \boldsymbol{S} \\ \textbf{sub.to} & \boldsymbol{I}w - \boldsymbol{S} = \boldsymbol{C} - \sum_{p=1}^m \boldsymbol{A}_p y_p, \ \ \boldsymbol{S} \succ \boldsymbol{O} \end{cases}}$$

$\Downarrow$ **Unconstrained convex minimization (Lagrangian dual):**

Given $\mu > 0$**,** $\boxed{\widehat{\mathbf{D}}(\mu) : \textbf{ min. } g(\boldsymbol{y}, \mu) \ \ \textbf{sub.to } \boldsymbol{y} \in \mathbb{R}^m}$

$$\boxed{\boldsymbol{y}(\mu) = \underset{\boldsymbol{y} \in \mathbb{R}^m}{\textbf{argmin}} \ g(\boldsymbol{y}, \mu)}$$

Basic idea of LDIPM

• **Trace $\boldsymbol{y}(\mu)$, which converges to an optimal Lagrange multiplier vector of Primal as $\mu \to 0$, by the predictor-corrector method.**

• **When we compute $\boldsymbol{y}(\mu)$, we can retrieve p. and d. int. feasible solutions $\boldsymbol{X}(\mu), \boldsymbol{y}(\mu), \boldsymbol{w}(\mu), \boldsymbol{S}(\mu)$, which lie on the central trajectory. Therefore they converge to p. and d. opt. solutions as $\mu \to 0$, and $\{\boldsymbol{y}(\mu) : \mu > \boldsymbol{0}\}$ forms the central trajectory in the $\boldsymbol{y}$ space.**

Corrector procedure $\left\{\begin{array}{l}\textbf{Newton}\\ \textbf{BFGS quasi-Newton}\end{array}\right.$

Predictor procedure — CG to compute a predictor direction $-\dot{\boldsymbol{y}}(\mu)$

Algorithm framework of the LDIPM

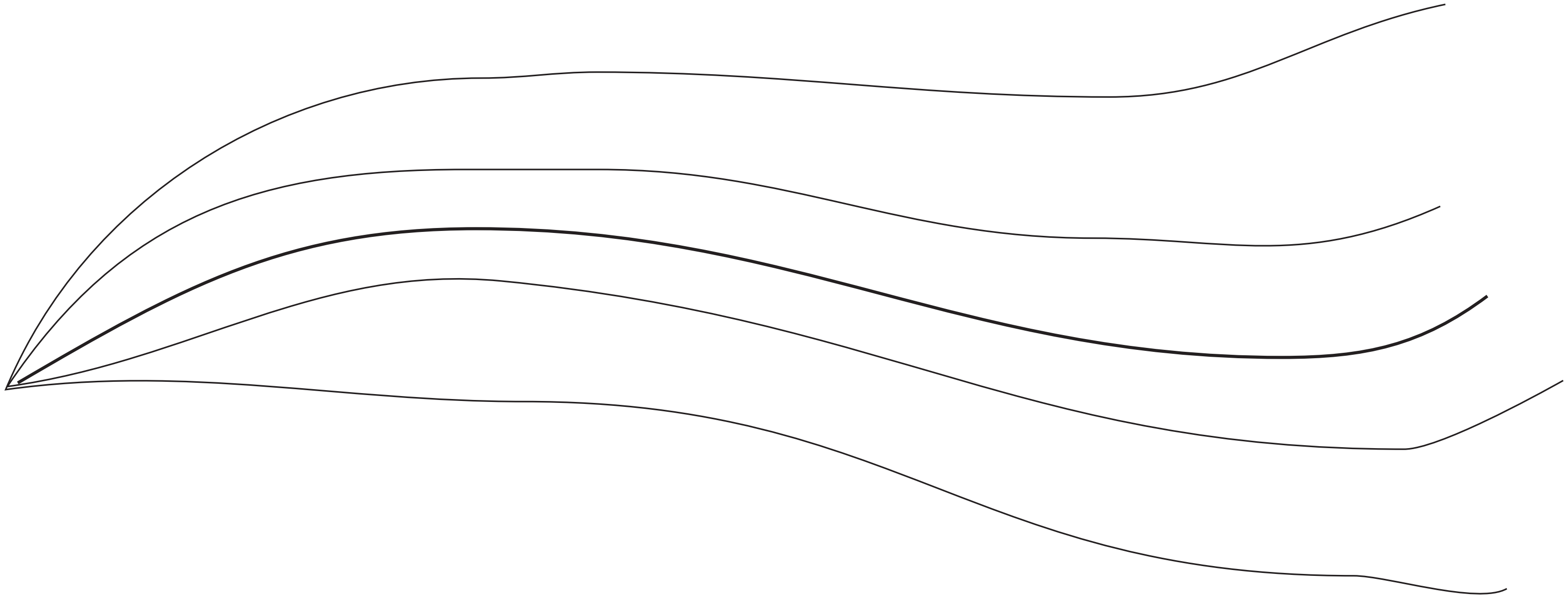
Step 0: **Let** $0 < \epsilon_c < \epsilon_p$**,** $\gamma \in (0, 1)$**. Initial** $\mu^0 \in \mathbb{R}_{++}$, $\bar{\boldsymbol{y}}^0 \in \mathbb{R}^m$, $k = 0$

Step 1: (CORRECTOR procedure) Let $\boldsymbol{z} = \boldsymbol{y}^k$**.**

Solve approximately $\{\min g(\boldsymbol{y}; \mu^k) : \boldsymbol{y} \in \mathbb{R}^m\}$**. Repeat**

- **find a search direction** $\boldsymbol{d}$
- **choose a step length** $\alpha \in (0, 1]$**, update** $\boldsymbol{z} := \boldsymbol{z} + \alpha \boldsymbol{d} \in \mathbb{R}^m$

until $(\boldsymbol{z}, \mu^k) \in N(\epsilon_c)$**. Let** $\boldsymbol{y}^k := \boldsymbol{z}$**.**



Direction $\boldsymbol{d}$ in Step 1 (Newton Method)

$$\boldsymbol{d} = - \left[\boldsymbol{\nabla}^2 g(\boldsymbol{z}; \mu^k) \right]^{-1} \boldsymbol{\nabla} g(\boldsymbol{z}; \mu^k)$$

- **Need to compute the Hessian $\boldsymbol{\nabla}^2 g(\boldsymbol{y}; \mu^k)$**
- **Neighborhood**

$$N(\epsilon) = \{(\boldsymbol{y}, \mu^k) \in \mathbb{R}^m \times \mathbb{R}_{++} : \boldsymbol{\nabla} g(\boldsymbol{y}; \mu^k)^T \boldsymbol{\nabla}^2 g(\boldsymbol{y}; \mu^k)^{-1} \boldsymbol{\nabla} g(\boldsymbol{y}; \mu^k) \leq \mu\epsilon\}$$

A neighborhood based on the self-concordant theory by Nesterov and Nemirovskii

Direction $\boldsymbol{d}$ in Step 1 (BFGS Quasi-Newton Method)

- **Initialize $\boldsymbol{H} = \boldsymbol{I}$ at the beginning**

$$\begin{aligned} \boldsymbol{d} &= -\boldsymbol{H}\boldsymbol{\nabla} g(\boldsymbol{z};\mu^k), \\ \boldsymbol{H}^+ &= \boldsymbol{H} - \frac{\boldsymbol{H}\boldsymbol{\eta}\boldsymbol{\sigma}^T + \boldsymbol{\sigma}(\boldsymbol{H}\boldsymbol{\eta})^T}{\boldsymbol{\sigma}^T\boldsymbol{\eta}} + \left(1 + \frac{\boldsymbol{\eta}^T\boldsymbol{H}\boldsymbol{\eta}}{\boldsymbol{\sigma}^T\boldsymbol{\eta}}\right)\frac{\boldsymbol{\sigma}\boldsymbol{\sigma}^T}{\boldsymbol{\sigma}^T\boldsymbol{\eta}}, \end{aligned}$$

where $\boldsymbol{\sigma} = \boldsymbol{z}^+ - \boldsymbol{z}$ and $\boldsymbol{\eta} = \boldsymbol{\nabla} g(\boldsymbol{z}^+;\mu^k) - \boldsymbol{\nabla} g(\boldsymbol{z};\mu^k)$

- **Neighborhood**

$$\tilde{N}(\epsilon) = \{(\boldsymbol{y},\mu^k) \in \mathbb{R}^m \times \mathbb{R}_{++} : \boldsymbol{\nabla} g(\boldsymbol{y};\mu^k)^T \boldsymbol{H}\boldsymbol{\nabla} g(\boldsymbol{y};\mu^k) \le \mu\epsilon\}$$

Step 2: (PREDICTOR procedure) Compute $\dot{\boldsymbol{y}}(\mu^k)$. Let $\gamma \in (0,1)$, $\delta = 1/\gamma$. Repeat

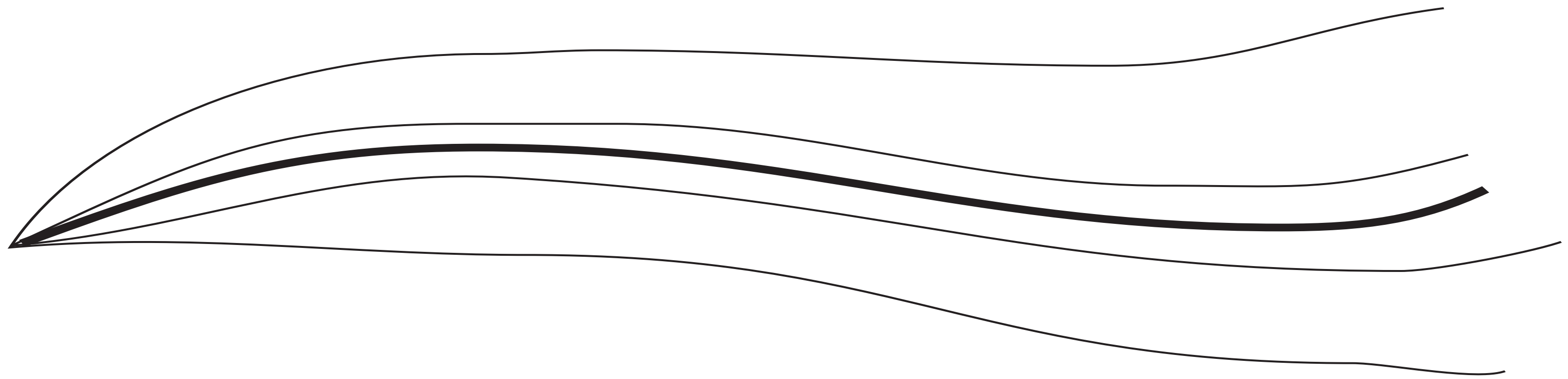
- $\delta = \gamma\delta,\ \ \bar{\mu} = (1-\delta)\mu^k$
- $\bar{\boldsymbol{y}} = \boldsymbol{y}^k + (\bar{\mu} - \mu^k)\dot{\boldsymbol{y}}^k = \boldsymbol{y}^k - \delta\dot{\boldsymbol{y}}^k$ **(the 1st order)**

until $(\bar{\boldsymbol{y}}, \bar{\mu}) \in N(\epsilon_p)$. Let $\mu^{k+1} := \bar{\mu},\ \ \bar{\boldsymbol{y}}^{k+1} := \bar{\boldsymbol{y}}$

Step 3: $k := k+1$, go to Step 1

- **We may be able to use**

$$\bar{\boldsymbol{y}} = \boldsymbol{y}^k + (\bar{\mu} - \mu^k)\dot{\boldsymbol{y}}^k + ((\bar{\mu} - \mu^k)^2/2)\ddot{\boldsymbol{y}}^k \text{ (the 2nd order)}$$



Computation of the 1st order derivative $\dot{\boldsymbol{y}}^k = \dot{\boldsymbol{y}}(\mu^k)$

$$\boldsymbol{\nabla}^2 g(\boldsymbol{y}^k, \mu^k)\dot{\boldsymbol{y}}^k = \exists \boldsymbol{a}^k$$

- **The cholesky factorization, the CG or CR methods**

Computation of the 2nd order derivative $\ddot{\boldsymbol{y}}^k = \ddot{\boldsymbol{y}}(\mu^k)$

$$\boldsymbol{\nabla}^2 g(\boldsymbol{y}^k, \mu^k)\ddot{\boldsymbol{y}}^k = \exists \boldsymbol{r}^k$$

- **Computation of $\boldsymbol{r}^k$ is much more expensive than that of $\boldsymbol{a}^k$!**

• How do we combine the predictor and the corrector procedures effectively?

• How do we utilize the information obtained at the corrector procedure for the succeeding predictor procedure?

Corrector procedure with the use of BFGS quasi-Newton method
$\Downarrow$
An effective preconditioning matrix for the CG method in the succeeding predictor procedure

(Morales-Nocedal '01)

Smooth nonlinear equation system for $\boldsymbol{y}(\mu)$ $(\mu > 0)$:

$$\boldsymbol{y}(\mu) = \operatorname*{\mathbf{argmin}}_{\boldsymbol{y}\in\mathbb{R}^m} g(\boldsymbol{y}, \mu) \iff \boldsymbol{\nabla} g(\boldsymbol{y}, \mu) = \mathbf{0} \tag{2}$$

L. eq. system for a pred. direction $\dot{\boldsymbol{y}}(\mu^k)$ at $\mu = \mu^k$ and $\boldsymbol{y}^k = \boldsymbol{y}(\mu^k)$:

$$\left.\frac{\boldsymbol{\nabla} g(\boldsymbol{y}, \mu)}{d\mu}\right|_{\boldsymbol{y}=\boldsymbol{y}^k,\mu=\mu^k} = \boldsymbol{\nabla}^2 g(\boldsymbol{y}^k, \mu^k)\dot{\boldsymbol{y}}(\mu^k) + \left.\frac{\partial \boldsymbol{\nabla} g(\boldsymbol{y}, \mu)}{\partial \mu}\right|_{\boldsymbol{y}=\boldsymbol{y}^k,\mu=\mu^k} = \mathbf{0}. \tag{3}$$

Solve (3) by CG method. Let $\boldsymbol{z}^0 = \boldsymbol{y}^k + \alpha\dot{\boldsymbol{y}}(\mu^k)$ and $\mu^{k+1} = \mu^k + \alpha$, where $\alpha \in (0, 1]$: a step length.

Corrector iterations to minimize $g(\boldsymbol{z}, \mu^{k+1})$ using the Newton method with the initial point $\boldsymbol{z} = \boldsymbol{z}^0$. Let $p = 0$.

$$\boldsymbol{z}^{p+1} = \boldsymbol{z}^p - \left(\boldsymbol{\nabla}^2\ g(\boldsymbol{z}^p, \mu^{k+1})\right)^{-1} \boldsymbol{\nabla} g(\boldsymbol{z}^p, \mu^{k+1}) \tag{4}$$

- **Hessian mat. $\boldsymbol{\nabla}^2 g(\boldsymbol{z}, \mu)$ in both pred. and corr. procedures.**
- **If we use BFGS method: $\boldsymbol{z}^{p+1} = \boldsymbol{z}^p - \boldsymbol{H}_p \boldsymbol{D}_{\boldsymbol{z}}\ g(\boldsymbol{z}^p, \mu^{k+1})$, then $\boldsymbol{H}_p \approx \left(\boldsymbol{\nabla}^2\ g(\boldsymbol{z}^p, \mu^{k+1})\right)^{-1}$ works as a preconditioner for CG meth.**

More details about LDIPM

How do we compute

$$g(\boldsymbol{y},\mu) = \min \left\{ \sum_{p=1}^{m} a_p y_p + bw - \mu \log\det \boldsymbol{S} : \begin{array}{l} \boldsymbol{I}w - \boldsymbol{S} = \boldsymbol{C} - \sum_{p=1}^{m} \boldsymbol{A}_p y_p, \\ \boldsymbol{S} \succ \boldsymbol{O} \end{array} \right\},$$

$\boldsymbol{\nabla}\ g(\boldsymbol{y},\mu)$ **(the gradient vector),   and**

$\boldsymbol{\nabla}^2\ g(\boldsymbol{y},\mu)$ **(the Hessian matrix)?**

$\boldsymbol{y} \in \mathbb{R}^m$ **and** $\mu > 0$ **: given.**

$$g(\boldsymbol{y}, \mu) = \min \left\{ \sum_{p=1}^{m} a_p y_p + bw - \mu \log\det \boldsymbol{S} : \begin{array}{l} \boldsymbol{I}w - \boldsymbol{S} = \boldsymbol{C} - \sum_{p=1}^{m} \boldsymbol{A}_p y_p, \\ \boldsymbol{S} \succ \boldsymbol{O} \end{array} \right\}.$$

KKT cond.:
$$\boxed{\begin{array}{c} \boldsymbol{I} \bullet \boldsymbol{X}(\boldsymbol{y}, \mu) = b, \quad \boldsymbol{S}(\boldsymbol{y}, \mu) = \sum_{p=1}^{m} \boldsymbol{A}_p y_p + \boldsymbol{I}w(\boldsymbol{y}, \mu) - \boldsymbol{C}, \\ \boldsymbol{X}(\boldsymbol{y}, \mu)\boldsymbol{S}(\boldsymbol{y}, \mu) = \mu \boldsymbol{I}, \quad \boldsymbol{X}(\boldsymbol{y}, \mu) \succeq \boldsymbol{O}, \quad \boldsymbol{S}(\boldsymbol{y}, \mu) \succeq \boldsymbol{O} \end{array}}$$

$$\Downarrow \ \boldsymbol{X}(\boldsymbol{y}, \mu) = \mu \boldsymbol{S}(\boldsymbol{y}, \mu)^{-1}$$

$$\boxed{\phi(w; \boldsymbol{y}, \mu) \equiv \mu \boldsymbol{I} \bullet \left(\sum_{p=1}^{m} \boldsymbol{A}_p y_p + \boldsymbol{I}w - \boldsymbol{C} \right)^{-1} = b, \ \left(\sum_{p=1}^{m} \boldsymbol{A}_p y_p + \boldsymbol{I}w - \boldsymbol{C} \right) \succ \boldsymbol{O}}$$

1. **Newton meth. to** $\phi(w; \boldsymbol{y}, \mu) = b$ **to compute** $w = w(\boldsymbol{y}, \mu)$ **using**
$$\frac{d\phi}{dw}(w; \boldsymbol{y}, \mu) = -\mu \boldsymbol{I} \bullet \left(\sum_{p=1}^{m} \boldsymbol{A}_p y_p + \boldsymbol{I}w - \boldsymbol{C} \right)^{-2},$$
Cholesky factorization of $\boldsymbol{S} = \sum_{p=1}^{m} \boldsymbol{A}_p y_p + \boldsymbol{I}w - \boldsymbol{C}$.
2. **Let** $\boldsymbol{S}(\boldsymbol{y}, \mu) = \boldsymbol{I}w(\boldsymbol{y}, \mu) - \boldsymbol{C} + \sum_{p=1}^{m} \boldsymbol{A}_p y_p$**.**
3. **Let** $g(\boldsymbol{y}, \mu) = \sum_{p=1}^{m} a_p y_p + bw(\boldsymbol{y}, \mu) - \mu \log\det \boldsymbol{S}(\boldsymbol{y}, \mu)$

$\boldsymbol{\nabla}\ g(\boldsymbol{y},\mu)$, $\boldsymbol{\nabla}^2 g(\boldsymbol{y},\mu)$ and properties of $g(\boldsymbol{y},\mu)$

1. $$\begin{aligned}\boldsymbol{\nabla} g(\boldsymbol{y},\mu) &= (a_1 - \mu\boldsymbol{A}_1 \bullet \boldsymbol{S}(\boldsymbol{y},\mu)^{-1}, \ldots, a_m - \mu\boldsymbol{A}_m \bullet \boldsymbol{S}(\boldsymbol{y},\mu)^{-1})^T \\ &= (a_1 - \boldsymbol{A}_1 \bullet \boldsymbol{X}(\boldsymbol{y},\mu), \ldots, a_m - \boldsymbol{A}_m \bullet \boldsymbol{X}(\boldsymbol{y},\mu))^T\end{aligned}$$

2. $\boldsymbol{\nabla}^2 g(\boldsymbol{y},\mu) = (\boldsymbol{M} - \boldsymbol{h}\boldsymbol{h}^T/h_{m+1})$, **where**

$$\begin{aligned}M_{qr} &= \mu\boldsymbol{A}_q \bullet \boldsymbol{S}(\boldsymbol{y},\mu)^{-1}\boldsymbol{A}_r\boldsymbol{S}(\boldsymbol{y},\mu)^{-1} \qquad (q,r=1,\ldots,m) \\ \boldsymbol{h} &= (\mu\boldsymbol{A}_1 \bullet \boldsymbol{S}(\boldsymbol{y},\mu)^{-2}, \ldots, \mu\boldsymbol{A}_m \bullet \boldsymbol{S}(\boldsymbol{y},\mu)^{-2})^T \\ h_{m+1} &= \mu\boldsymbol{I} \bullet \boldsymbol{S}(\boldsymbol{y},\mu)^{-2}\end{aligned}$$

(the coefficient matrix of the Schur complement equation)

(the most expensive part to compute)

3. $\boldsymbol{\nabla}^2 g(\boldsymbol{y},\mu)$ **is positive definite** **(**$g(\boldsymbol{y},\mu)$ **is strictly convex)**

4. $\{g(\cdot,\mu)\ :\ \mu \in \mathbb{R}_{++}\}$ **is "a self-concordant family" on** $\mathbb{R}^m$

Preliminary numerical experiments

- **4 variants of the LDIPM**

	Corrector	
Predictor	**Newton**	**BFGS quasi-Newton**
1**st order**		
2**nd order**†		

† : $\boldsymbol{y}^{k+1} = \boldsymbol{y}(\mu^k) + \alpha\dot{\boldsymbol{y}}(\mu^k) + \frac{1}{2}\alpha^2\ddot{\boldsymbol{y}}(\mu^k)$

- **Stopping criterion**

$$\textbf{relative error} = \frac{|\textbf{primal obj. - dual obj.}|}{\max\{\textbf{primal obj.}, 1.0\}} < 1.0e-6$$
$$\textbf{primal feasibility error} = \max_{p=1,\ldots,m}|a_p - \boldsymbol{A}_p \bullet \boldsymbol{X}| < 1.0e-6$$

- **MATLAB Version 5.2**

- **Macintosh with PowerPC 750 400MHz and 360 MB memory**

Box Constrained Quadratic ± 1 Program

- **Average of** 5 **problems** $\{\max \boldsymbol{x}^T \boldsymbol{Q} \boldsymbol{x}$ **sub.to** $x_i^2 = 1, \ (i = 1, 2, \ldots, n)\}$
- **Matrix size** $n = 200$

Corrector **Predictor**	**Newton** **1st-order**	**Newton** **2nd-order**	**BFGS** **1st-order**	**BFGS** **2nd-order**
major # it. **CPU**	**13.4** **3252s**	**10.8** **1529s**	**12.6** **763s**	**10.2** **585s**
Newton # it. **BFGS # it.**	**27.0** **-**	**19.6** **-**	**-** **210.2**	**-** **180.0**
Cholesky of $\boldsymbol{S}$ **CG**	**285.4** **-**	**165.8** **-**	**795.8** **188.4**	**567.8** **177.2**
$\kappa(\boldsymbol{\nabla}^2 \boldsymbol{g}(\boldsymbol{y}, \mu))$ $\kappa(\boldsymbol{H}\boldsymbol{\nabla}^2 \boldsymbol{g}(\boldsymbol{y}, \mu))$	**6.2e+7** **-**	**3.4e+7** **-**	**2.7e+7** **7.8e+1**	**2.2e+7** **8.6e+1**

$\boldsymbol{\nabla}^2 \boldsymbol{g}(\boldsymbol{y}, \mu)$ **: the Hessian matrix of** $\boldsymbol{g}(\cdot, \mu)$

$\boldsymbol{H}$ **: the BFGS matrix**

$\kappa(\boldsymbol{A})$ **: the condition number of** $\boldsymbol{A}$**.**

Norm Minimization Problem

- **Average of** 5 **problems**
- **Matrix size** $n = 50$**, constraints** $m = 200$

Corrector Predictor	Newton 1st-order	Newton 2nd-order	BFGS 1st-order	BFGS 2nd-order
major # it.	14.8	12.6	14.2	12.6
CPU	843s	544s	240s	210s
Newton # it.	39.2	28.0	-	-
BFGS # it.	-	-	340.0	319.8
Cholesky of $\boldsymbol{S}$	198.6	107.8	608.2	509.4
CG	-	-	228.2	262.2
$\kappa(\boldsymbol{\nabla}^2\boldsymbol{g}(\boldsymbol{y},\mu))$	7.8e+9	9.2e+9	4.8e+9	1.2e+10
$\kappa(\boldsymbol{H}\boldsymbol{\nabla}^2\boldsymbol{g}(\boldsymbol{y},\mu))$	-	-	3.3e+2	1.7e+3

$\boldsymbol{\nabla}^2\boldsymbol{g}(\boldsymbol{y},\mu)$ **: the Hessian matrix of** $\boldsymbol{g}(\cdot,\mu)$

$\boldsymbol{H}$ **: the BFGS matrix**

$\kappa(\boldsymbol{A})$ **: the condition number of** $\boldsymbol{A}$

Condition and scaled condition numbers along the iterations

- **Box Constrained Quadratic ± 1 Program**
- **Matrix size $n = 200$, constraints $m = 201$**

k	μ^k	rel.error	$\nabla^2 g(\boldsymbol{y}^k, \mu^k)$	$\kappa(\boldsymbol{H}^k \nabla^2 g(\boldsymbol{y}^k, \mu^k))$	#CG
1	1.41e+1	+2.81e+1	2.17e+2	3.03e+3	4
2	3.83e+0	+1.87e+0	3.42e+2	9.00e+2	9
3	2.00e+0	+6.17e−1	7.57e+2	6.75e+2	14
4	8.19e−1	+1.92e−1	1.75e+3	1.07e+3	24
5	2.22e−1	+4.62e−2	2.58e+3	2.86e+1	16
6	4.22e−2	+8.48e−3	3.01e+3	5.94e+1	18
7	4.22e−3	+8.47e−4	1.32e+4	1.93e+4	44
8	4.22e−4	+8.47e−5	1.33e+5	1.62e+2	18
9	4.22e−5	+8.43e−6	1.33e+6	3.37e+1	14
10	4.22e−6	+8.45e−7	1.33e+7	5.18e+2	16

$\nabla^2 \boldsymbol{g}(\boldsymbol{y}, \mu)$ **: the Hessian matrix of** $\boldsymbol{g}(\cdot, \mu)$

$\boldsymbol{H}$ **: the BFGS matrix**

$\kappa(\boldsymbol{A})$ **: the condition number of** $\boldsymbol{A}$**.**

Summary

$\Longrightarrow$ **New type of predictor-corrector dual IP method for SDP**

$$\begin{cases} \textbf{dual feasible, primal infeasible} \\ \boldsymbol{XS} = \mu \boldsymbol{I} \end{cases}$$

$\Longrightarrow$ **(CORRECTOR procedure)**

Quasi-Newton BFGS instead of Newton method

$\Longrightarrow$ **(PREDICTOR Step)**

BFGS matrix $\boldsymbol{H}$ is a good preconditioner for the CG

$\Longrightarrow$ **Can be extended to Linear Optimization Problems over convex cones (LP, SOCP)**

Further Directions

$\Longrightarrow$ **Implementation in C/C++**

$\Longrightarrow$ **Improve numerical convergence**

$\Longrightarrow$ **Limited memory BFGS method for large scale problems**