Interior-point Algorithms: Methods & Tools

Part II: Conic IPMs and ECOS

Alexander Domahidi

Co-founder embotech GmbH

July 30, 2015

TEMPO Summer School on Numerical Optimal Control
University of Freiburg
Germany

Convex Optimization is the Workhorse

- Many problems can be boiled down to solving

$$\text{minimize } 0.5x^T H x + f^T x$$
$$\text{subject to } Ax = b$$
$$Gx \preceq_K h$$

Bounds, polytopes, second-order cones, 2-norm balls, exponential cones, ...

  - Linear constrained optimal control
  - Nonlinear programming: sequential quadratic programming
  - Mixed-integer problems: convex relaxations
  - Stochastic optimization: sampling
- In fact, this is what we *can* solve reliably
- In real-time control: **parametric convex problems**

In Part II: Conic IPMs & ECOS

$$\text{minimize } \cancel{0.5x^T H x} + f^T x$$
$$\text{subject to } Ax = b$$
$$Gx \boxed{\preceq_K} h$$

Bounds, polytopes,
second-order cones,
2-norm balls,
exponential cones, ...

- Conic problems are nonlinear convex problems
- Hence can be efficiently solved by e.g. interior-point methods
- ECOS is a solver implementing a conic IPM with sparse LA

Second-order Cone Programs

- Minimize linear objective over convex pointed cone ∩ affine equality:

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax = b \\ & Gx \preceq_K h \end{array} \qquad \text{(SOCP)}$$

where $\mathbf{K} \triangleq \mathbf{K}_1 \times \mathbf{K}_2 \times \ldots \mathbf{K}_N$ and $\mathbf{K}_i = \begin{cases} \mathbf{R}_+ & \text{(positive orthant)} \\ \mathbf{Q}^{n_i} & \text{(second-order cone)} \end{cases}$

with $\mathbf{Q}^{n_i} \triangleq \{(x_0, x_1) \in \mathbf{R} \times \mathbf{R}^{n_i - 1} \mid x_0 \geq \|x_1\|_2\}$

- LPs, QPs and QCQPs can be formulated as SOCPs

Applications of SOCPs

- **Signal processing**, e.g.
  - robust beamforming [Vorobyov et al., 2003]
  - error correction [Candes & Randall, 2008]
- **Power grids**, e.g. optimal power flow [Sojoudi & Lavaei, 2012]
- **Finance**, e.g. robust portfolio selection [Goldfarb & Iyengar, 2003]
- **Machine learning**, e.g. group LASSO [Meier et al., 2008]
- **Control**, e.g.
  - Robust MPC via affine feedback policies [Goulart et al., 2006]
  - **Minimum-fuel powered descent for spacecraft** [Acikmese & Ploen, 2007]
  - Soft-constrained MPC with stability guarantees [Zeilinger et al., 2013]
  - **Minimum-time trajectories for robots** [Verscheure et al., 2013]
- **MedTec**: Radiation therapy planning [Chu et al, 2005]

Example: Minimum Time Path Tracking

- Goal: follow given trajectory with robot arm as quickly as possible
- Optimization problem:

minimize	time
subject to	robot tip on given trajectory
	system dynamics
	maximum torque at joints

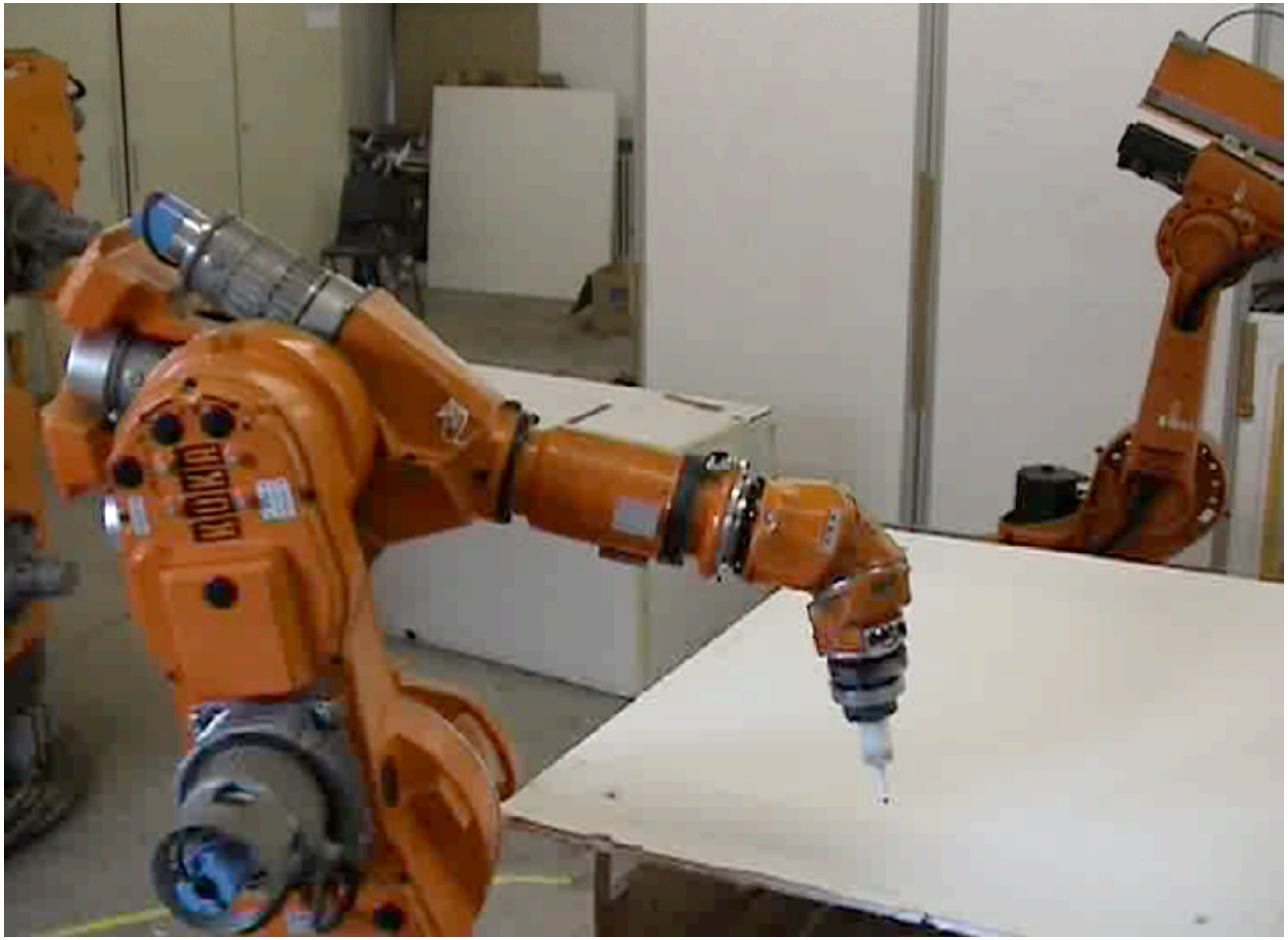
Example: Minimum Time Path Tracking

- Goal: follow given trajectory with robot arm as quickly as possible
- Optimization problem:

minimize	time
subject to	robot tip on given trajectory
	system dynamics
	maximum torque at joints

- Results in convex SOCP [Verscheure, Demeulenaere, Swevers, De Schutter, Diehl 2009]
  - there is no faster way of tracking a path
  - constraints are satisfied
  - optimum can be computed efficiently

Example: Minimum Time Path Tracking

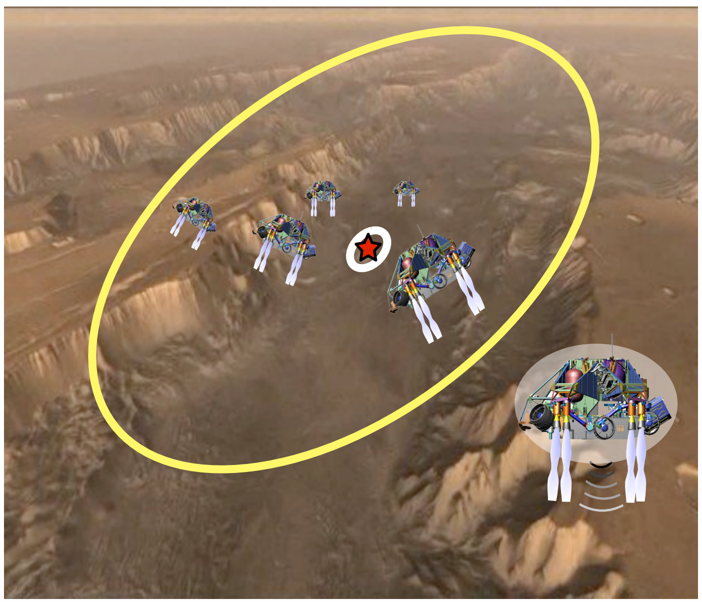


[Link to Video]

(source: Verscheure et al., 2009)

SOCPs for Min-Fuel Powered Descent

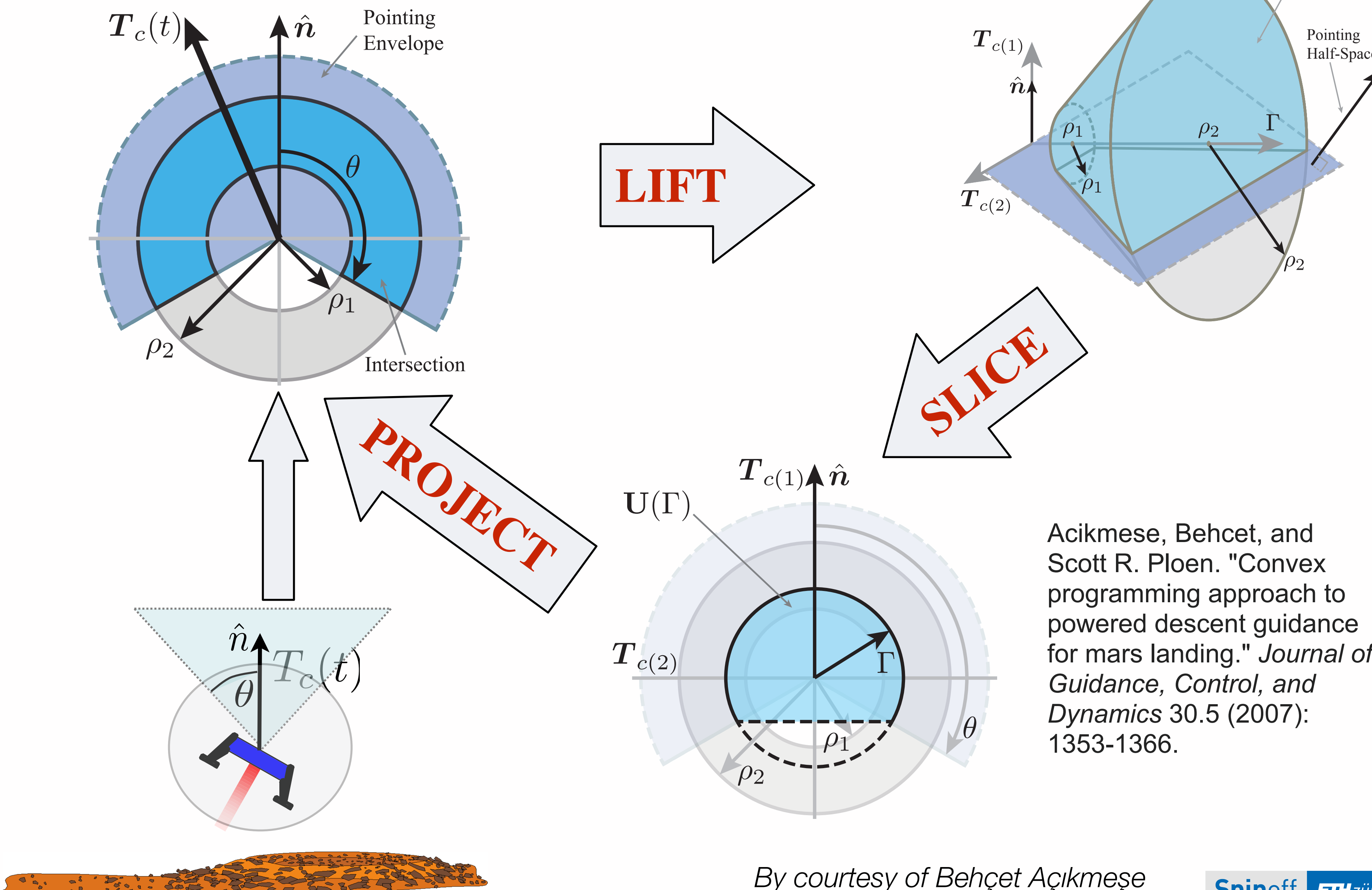
Real-time Optimization for Advanced Automation



Behçet Açıkmeşe

Department of Aerospace Engineering and Engineering Mechanics
University of Texas at Austin

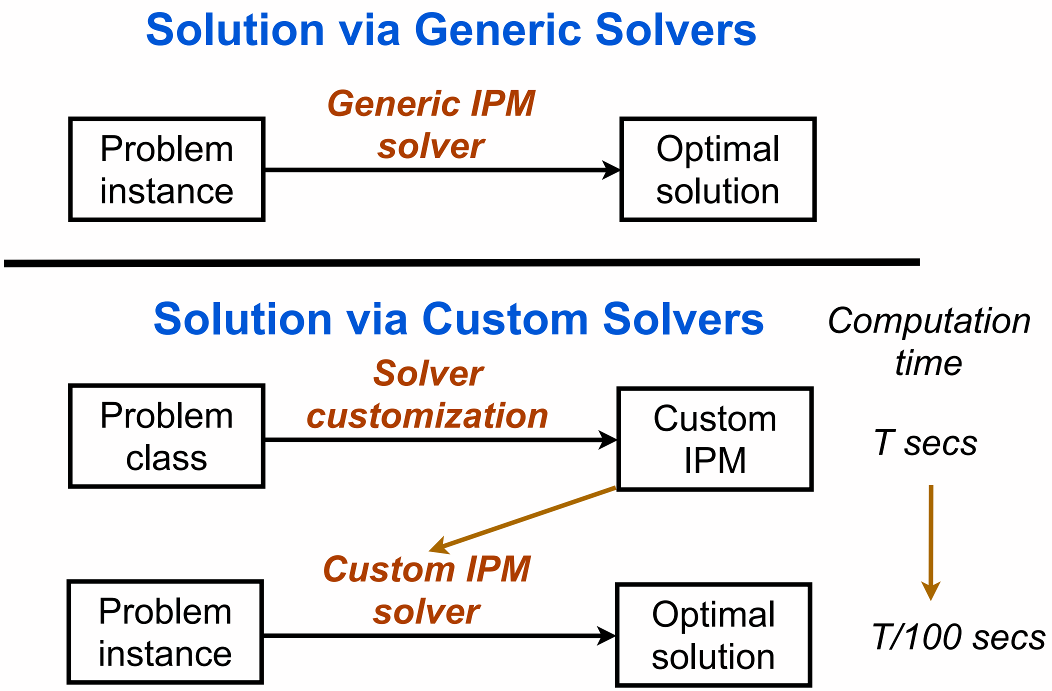
Convexification Method



Acikmese, Behcet, and Scott R. Ploen. "Convex programming approach to powered descent guidance for mars landing." *Journal of Guidance, Control, and Dynamics* 30.5 (2007): 1353-1366.

By courtesy of Behçet Açıkmeşe

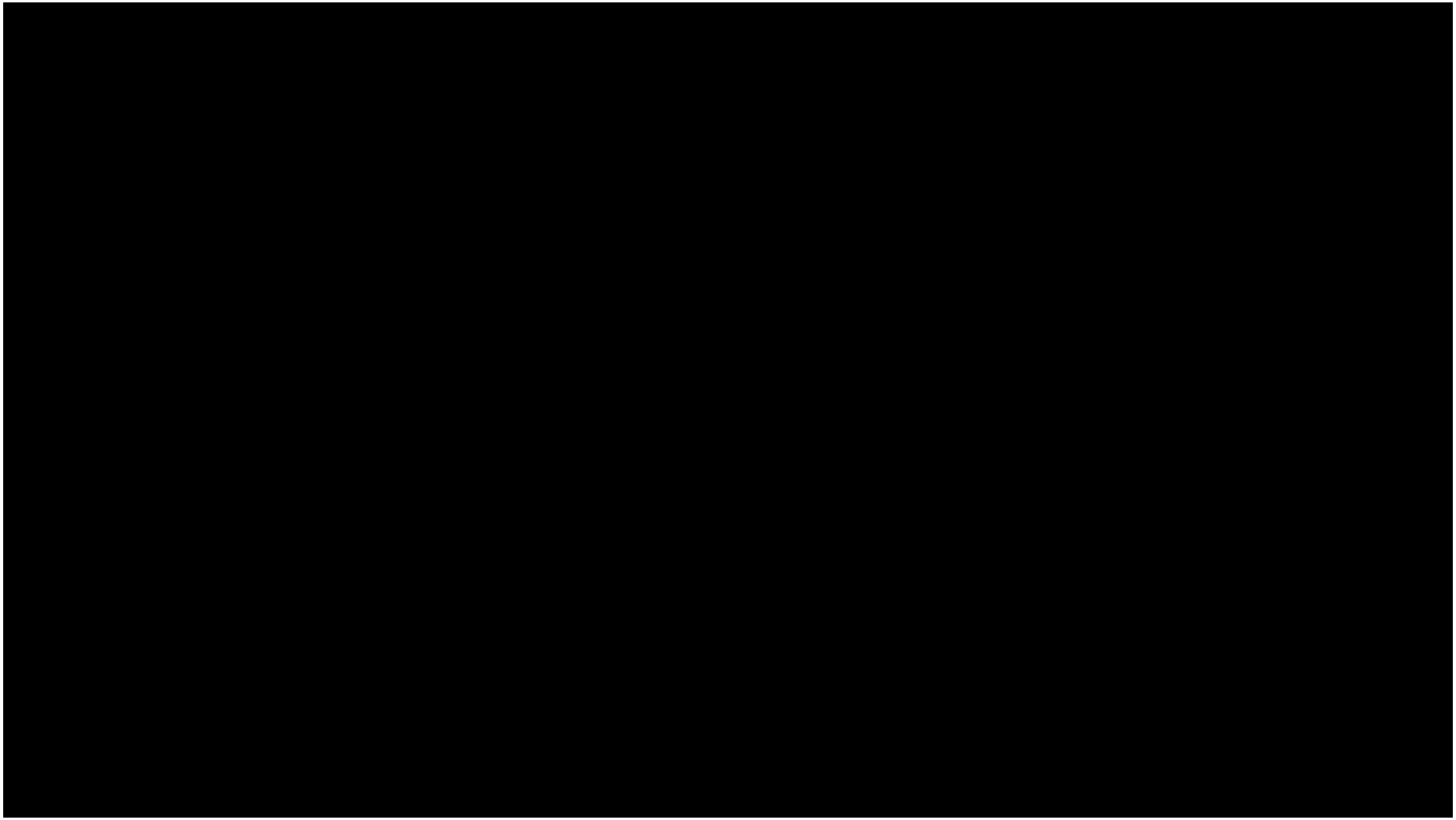
Solve Times for SOCPs



Method	NLP-based	Generic IPM for SOCP	Custom IPM for SOCP
CPU time (ms) on a laptop	20,000	1,000	10 - 15
Reliability	< 80%	> 99%	> 99%

 By courtesy of Behçet Açıkmeşe

Example: Min-Fuel Powered Descent



Source: Youtube ("Xombie 750m Mars EDL Divert Trajectory")

[Watch online]

Conic Programming

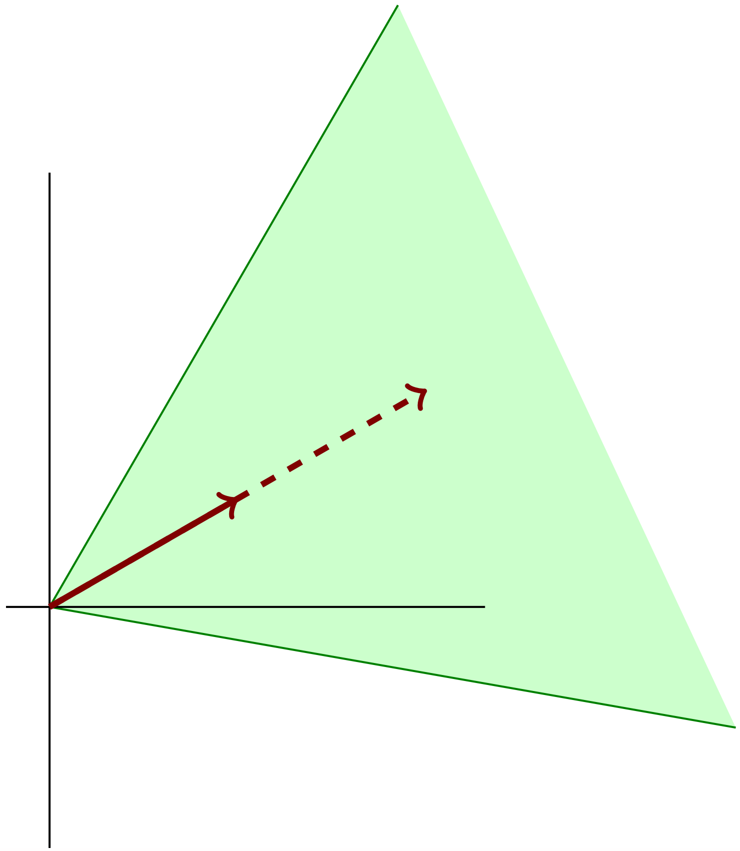
Cone LP

$$\begin{aligned} &\text{minimize } c^T x \\ &\text{subject to } Ax = b \\ &\qquad x \in \mathcal{K} \end{aligned}$$

By courtesy of Santiago Akle, Stanford University

What is a Proper Cone?

- If $x \in \mathcal{K}$ then all positive scalings $\alpha x \in \mathcal{K}$
- Closed
- Convex
- Pointed (if $x \in \mathcal{K}$ then $-x \notin \mathcal{K}$)
- With nonempty interior

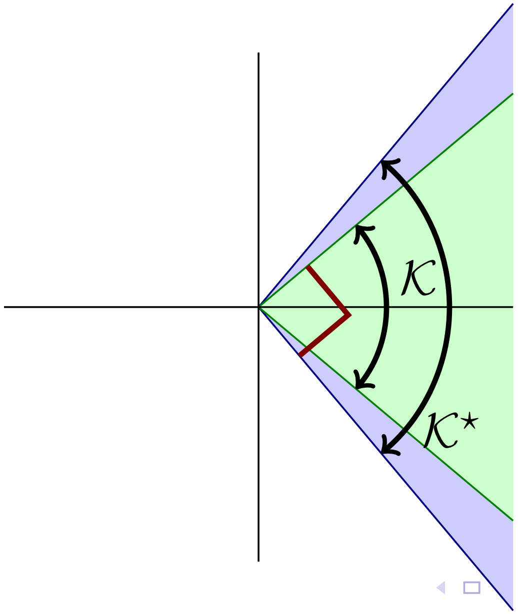


By courtesy of Santiago Akle, Stanford University

Dual Cone and Dual Problem

$$\begin{aligned} \text{minimize } & c^T x \\ \text{subject to } & Ax = b \\ & x \in \mathcal{K} \end{aligned} \qquad \begin{aligned} \text{minimize } & -b^T y \\ \text{subject to } & A^T y + s = c \\ & s \in \mathcal{K}^\star \end{aligned}$$

$$\mathcal{K}^\star = \left\{ s \mid x^T s \geq 0 \text{ for all } x \in \mathcal{K} \right\}$$



By courtesy of Santiago Akle, Stanford University

Cartesian Product of Cones

The product

$$\mathcal{K} = \mathcal{K}_1 \times \mathcal{K}_2$$

is a cone, and has dual

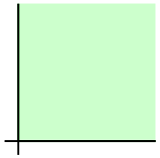
$$\mathcal{K}^\star = \mathcal{K}_1^\star \times \mathcal{K}_2^\star$$

By courtesy of Santiago Akle, Stanford University

The Most Important Cones

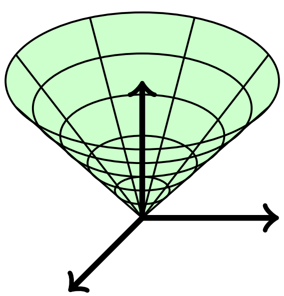
Positive orthant

$$\mathbb{R}^n_+ = \{x \mid 0 \leq x_i \; \forall i\}$$



Second-order cone

$$\mathcal{L} = \{x, \tau \mid \|x\|_2 \leq \tau\}$$

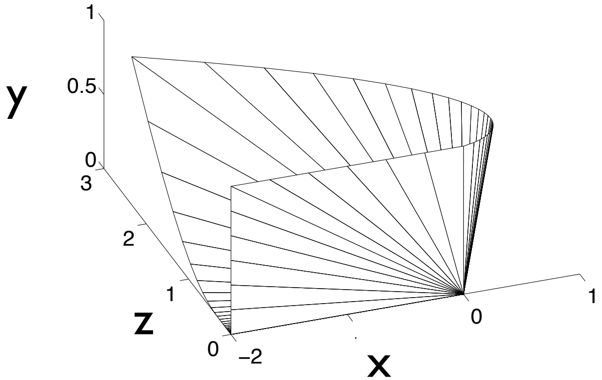


Positive semi-definite matrices

$$\mathcal{S}^n_+ = \left\{X \mid X = X^T, \; X \succeq 0\right\}$$

Exponential cone

$$\mathcal{K}_e = \mathbf{cl}\left\{\begin{pmatrix} x \\ y \\ z \end{pmatrix} \mid \exp\left(\frac{x}{z}\right) \leq \frac{y}{z}, \;\; z > 0\right\}$$

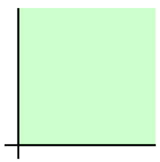


By courtesy of Santiago Akle, Stanford University & Lieven Vandenberghe, UCLA

Cones Supported by ECOS

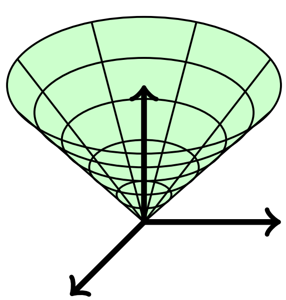
Positive orthant

$$\mathbb{R}^n_+ = \{x \mid 0 \le x_i \ \forall i\}$$



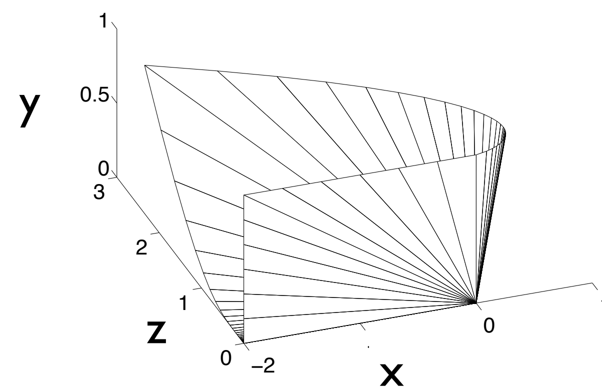
Second-order cone

$$\mathcal{L} = \{x, \tau \mid \|x\|_2 \le \tau\}$$



Exponential cone

$$\mathcal{K}_e = \mathbf{cl}\left\{ \begin{pmatrix} x \\ y \\ z \end{pmatrix} \mid \exp\left(\frac{x}{z}\right) \le \frac{y}{z}, \ \ z > 0 \right\}$$



By courtesy of Santiago Akle, Stanford University & Lieven Vandenberghe, UCLA

Examples for SOCP-representable f(x)

- convex quadratic

$$f(x) = x^T P x + q^T x + r \qquad (P \succeq 0)$$

- quadratic-over-linear function

$$f(x, y) = \frac{x^T x}{y} \quad \text{with } \mathbf{dom}\, f = \mathbf{R}^n \times \mathbf{R}_+ \qquad (\text{assume } 0/0 = 0)$$

- convex powers with rational exponent

$$f(x) = |x|^\alpha, \qquad f(x) = \begin{cases} x^\beta & x > 0 \\ +\infty & x \leq 0 \end{cases}$$

for rational $\alpha \geq 1$ and $\beta \leq 0$

- p-norm $f(x) = \|x\|_p$ for rational $p \geq 1$

Material from Lieven Vandenberghe, UCLA

Examples for SOCP-representable f(x)

- convex quadratic

$$f(x) = x^T P x + q^T x + r \qquad (P \succeq 0)$$

Many more functions and examples in:

- Ben-Tal and Nemirovski. Lectures in Modern Convex Programming §2.3

- Lobo, Vandenberghe, Boyd, Lebret:
  Applications of Second-order cone programming, 1998

for rational $\alpha \geq 1$ and $\beta \leq 0$

- p-norm $f(x) = \|x\|_p$ for rational $p \geq 1$

Material from Lieven Vandenberghe, UCLA

Functions Representable by Exp Cones

- Logarithms
  - **Geometric programming**:

$$
\begin{array}{ll}
\text{minimize} & x^{-1}y^{-1/2}z^{-1} + 2.3xz + 4xyz \\
\text{subject to} & (1/3)x^{-2}y^{-2} + (4/3)y^{1/2}z^{-1} \leq 1, \\
 & x + 2y + 3z \leq 1, \\
 & (1/2)xy = 1,
\end{array}
$$

- Exponentials:
  - Logistic regression: $f(x) = \dfrac{1}{1 + e^{\beta_0 + \beta_1 x}}$

- Entropy: $f(x) = x \log x$

- Kullback-Leibler Divergence $KL(p, q) = \sum p_i \log \dfrac{p_i}{q_i}$

Optimization over Symmetric Cones

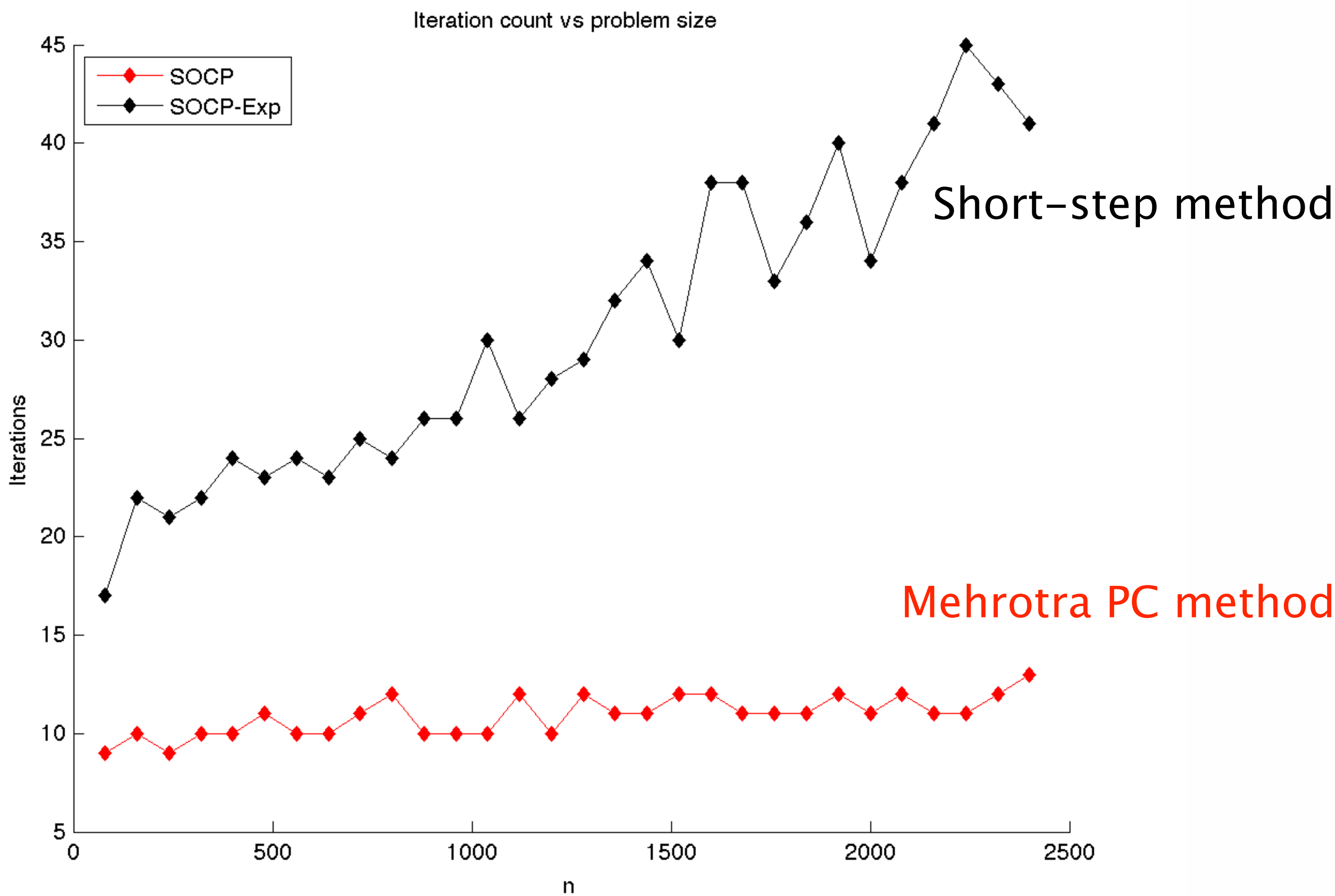
Symmetric Cones: $\mathcal{K} = \mathcal{K}^\star$

- Positive orthant
- Second-order cone
- SDP cone

- Consequence: powerful **long-step** interior-point methods
  - Mehrotra-predictor corrector works extremely well for these problems

- Exponential cones are not symmetric
  - more iterations needed in general (short step methods)

SOCP vs SOCP-Exp - #Iterations



Euclidean Jordan Algebra

- Each element in a symmetric cone can be spectrally decomposed:

$$x \in \mathcal{K} \Leftrightarrow \exists \lambda_i \geq 0, q_i : \boxed{x = \sum_{i=1}^{\theta} \lambda_i q_i}$$

  where vectors q_i form an orthonormal basis with identity element e

- Examples:
  - nonnegative orthant: $\lambda_i = x_i, \qquad q_i = e_i \quad (i\text{th unit vector}), \; i = 1, \ldots, n$
  - **second-order cone** $(K = \mathcal{Q}^p)$

    spectral decomposition of $x = (x_0, x_1) \in \mathbf{R} \times \mathbf{R}^{p-1}$ is

$$\lambda_i = \frac{x_0 \pm \|x_1\|_2}{\sqrt{2}}, \qquad q_i = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ \pm x_1/\|x_1\|_2 \end{bmatrix}, \qquad i = 1, 2$$

Some Interesting Facts

Property	Definition	Cone $\boldsymbol{R}_+$	Cone $\boldsymbol{Q}^n$, $n > 1$
$x \in \boldsymbol{Q}^n$	$\Leftrightarrow \lambda_i \geq 0\,,\ \forall i$	$x \geq 0$	$x_0 \pm \|x_1\|_2 \geq 0$
$x \in \text{int}\ \boldsymbol{Q}^n$	$\Leftrightarrow \lambda_i > 0\,,\ \forall i$	$x > 0$	$x_0 \pm \|x_1\|_2 > 0$
Inverse x^{-1} s.t. $x \circ x^{-1} = \boldsymbol{e}$	$x^{-1} \triangleq \sum_{i=1}^{R} \lambda_i^{-1} q_i$	$x^{-1} = 1/x$	$x^{-1} = (x_0, -x_1)/\det(x)$
Determinant: $\det(x)$	$\det(x) \triangleq \prod_{i=1}^{R} \lambda_i$	$\det(x) = x$	$\det(x) = x_0^2 - x_1^T x_1$

- Can be used to treat symmetric cones with one unified IPM theory

Schmieta, S. H., and Farid Alizadeh. "Extension of primal-dual interior point algorithms to symmetric cones." *Mathematical Programming* 96.3 (2003): 409-438.

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax = b \\ & Gx + s = h, s \in \mathbf{K} \end{array}$$

Central Path

- Use log-det barrier function $\Phi(x) = -\log\det x$ for $x \in \operatorname{int} \mathbf{K}$
- Property: $\nabla\Phi(x) = -x^{-1}$ by a spectral decomposition of x
- Primal-dual central path is the set of points satisfying

Primal-dual central path

$$\begin{bmatrix} 0 \\ 0 \\ s \end{bmatrix} = \begin{bmatrix} 0 & A^T & G^T \\ -A & 0 & 0 \\ -G & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} c \\ b \\ h \end{bmatrix}$$

$$z = -\mu\nabla\Phi(s)$$

$$(s, z) \succ_K 0$$

with path parameter $\mu > 0$

- $z = -\mu\nabla\Phi(s)$ can be written as $s \circ z = \mu\mathbf{e}$ for appropriate vector product $\circ$

Optimality Conditions

- KKT conditions are necessary and sufficient conditions for convex problems

Primal Problem:

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax = b \\ & Gx + s = h, s \in \mathbf{K} \end{array}$$

Dual Problem:

$$\begin{array}{ll} \text{maximize} & -b^T y - h^T z \\ \text{subject to} & A^T y + G^T z = -c \\ & z \in \mathbf{K} \end{array}$$

Relaxed Optimality Conditions

$$\begin{bmatrix} 0 \\ 0 \\ s \end{bmatrix} = \begin{bmatrix} 0 & A^T & G^T \\ -A & 0 & 0 \\ -G & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} c \\ b \\ h \end{bmatrix}$$

$$s \circ z = 0 \quad \rightarrow \quad s \circ z = \mu e$$

$$(s, z) \succeq_K 0 \quad \rightarrow \quad (s, z) \succ_K 0$$

- Primal-dual interior-point methods: relax KKT conditions & track central path

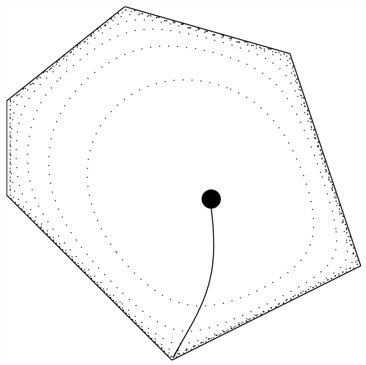
Path-following IPM

- Primal-dual central path (CP) is a continuously differentiable curve defined by the points (x, y, s, z) and $\mu > 0$ s.t. *[Nesterov & Todd, 1997]*

$$\begin{bmatrix} 0 \\ 0 \\ s \end{bmatrix} = \begin{bmatrix} 0 & A^T & G^T \\ -A & 0 & 0 \\ -G & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} c \\ b \\ h \end{bmatrix}$$

$$Wz \circ W^{-T} s = \mu \mathbf{e}$$

$$(s, z) \succ_K 0$$



- Path-following interior point methods track central path to solution:

1. Solve linearized central path equations to obtain search direction $\Delta(x, y, z, s)$
2. Determine step size α (line search)
3. Update W, variables $(x, y, z, s) \leftarrow (x, y, z, s) + \alpha\Delta(x, y, z, s)$ and $\mu \leftarrow s^T z/N$
4. Go to step 1

- 99% of computation time is spent in step 1

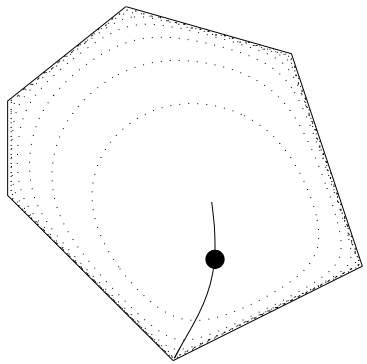
Path-following IPM

- Primal-dual central path (CP) is a continuously differentiable curve defined by the points (x, y, s, z) and $\mu > 0$ s.t. *[Nesterov & Todd, 1997]*

$$\begin{bmatrix} 0 \\ 0 \\ s \end{bmatrix} = \begin{bmatrix} 0 & A^T & G^T \\ -A & 0 & 0 \\ -G & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} c \\ b \\ h \end{bmatrix}$$

$$Wz \circ W^{-T} s = \mu \mathbf{e}$$

$$(s, z) \succ_K 0$$



- Path-following interior point methods track central path to solution:
  1. Solve linearized central path equations to obtain search direction $\Delta(x, y, z, s)$
  2. Determine step size α (line search)
  3. Update W, variables $(x, y, z, s) \leftarrow (x, y, z, s) + \alpha\Delta(x, y, z, s)$ and $\mu \leftarrow s^T z/N$
  4. Go to step 1
- 99% of computation time is spent in step 1

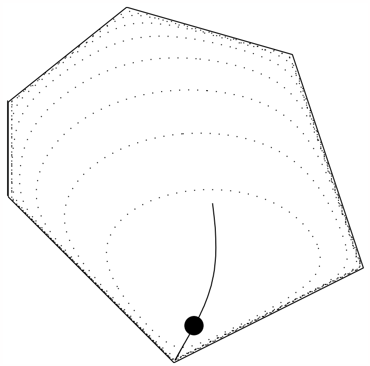
Path-following IPM

- Primal-dual central path (CP) is a continuously differentiable curve defined by the points (x, y, s, z) and $\mu > 0$ s.t. *[Nesterov & Todd, 1997]*

$$\begin{bmatrix} 0 \\ 0 \\ s \end{bmatrix} = \begin{bmatrix} 0 & A^T & G^T \\ -A & 0 & 0 \\ -G & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} c \\ b \\ h \end{bmatrix}$$

$$Wz \circ W^{-T} s = \mu \mathbf{e}$$

$$(s, z) \succ_K 0$$



- Path-following interior point methods track central path to solution:
  1. Solve linearized central path equations to obtain search direction $\Delta(x, y, z, s)$
  2. Determine step size α (line search)
  3. Update W, variables $(x, y, z, s) \leftarrow (x, y, z, s) + \alpha\Delta(x, y, z, s)$ and $\mu \leftarrow s^T z / N$
  4. Go to step 1
- 99% of computation time is spent in step 1

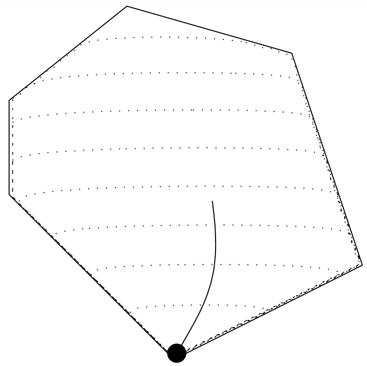
Path-following IPM

- Primal-dual central path (CP) is a continuously differentiable curve defined by the points (x, y, s, z) and $\mu > 0$ s.t. *[Nesterov & Todd, 1997]*

$$\begin{bmatrix} 0 \\ 0 \\ s \end{bmatrix} = \begin{bmatrix} 0 & A^T & G^T \\ -A & 0 & 0 \\ -G & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} c \\ b \\ h \end{bmatrix}$$

$$Wz \circ W^{-T}s = \mu \mathbf{e}$$

$$(s, z) \succ_K 0$$



- Path-following interior point methods track central path to solution:

1. Solve linearized central path equations to obtain search direction $\Delta(x, y, z, s)$
2. Determine step size α (line search)
3. Update W, variables $(x, y, z, s) \leftarrow (x, y, z, s) + \alpha\Delta(x, y, z, s)$ and $\mu \leftarrow s^T z/N$
4. Go to step 1

- 99% of computation time is spent in step 1

Primal-dual System

Duality

$$b^T y \leq d^\star \leq p^\star \leq c^T x$$

$$Ax = b \qquad\qquad x \in \mathcal{K}$$

$$A^T y + s = c \qquad\qquad s \in \mathcal{K}^\star$$

Primal and dual feasibility and complementarity

$$\begin{aligned} Ax &= b \\ A^T y + s &= c \\ c^T x - b^T y = x^T s &= 0 \\ x \in \mathcal{K}, \quad s &\in \mathcal{K}^\star \end{aligned}$$

By courtesy of Santiago Akle, Stanford University

Unboundedness and Infeasibility

To certify that a problem is unbounded

Find $\delta x \in \mathcal{K}$ such that $A\,\delta x = 0$ and $c^T\,\delta x < 0$. Then

$$\begin{aligned} c^T(x + \alpha\,\delta x) &\to -\infty \\ A(x + \alpha\,\delta x) &= b \\ x + \alpha\,\delta x &\in \mathcal{K} \end{aligned}$$

so the primal has to be unbounded

To certify that a problem is infeasible

Find $\delta s \in \mathcal{K}^\star$ and δy such that $A^T\,\delta y + \delta s = 0$ and $b^T\,\delta y > 0$. Then

$$\begin{aligned} b^T(y + \alpha\,\delta y) &\to \infty \\ A^T(y + \alpha\,\delta y) + s + \alpha\,\delta s &= c \\ s + \alpha\,\delta s &\in \mathcal{K}^\star \end{aligned}$$

so the dual has to be unbounded

By courtesy of Santiago Akle, Stanford University

Introduce 2 New Variables

$$
\begin{aligned}
Ax^\star &= \tau^\star b \\
A^T y^\star + s^\star &= \tau^\star c \\
b^T y^\star - c^T x^\star &= \kappa^\star
\end{aligned}
$$

When $\tau^\star > 0$ and $\kappa^\star = 0$ we found a solution
because $(x^\star/\tau^\star, y^\star/\tau^\star, s^\star/\tau^\star)$

$$
\begin{aligned}
Ax^\star/\tau^\star &= b \\
A^T y^\star/\tau^\star + s^\star/\tau^\star &= c \\
c^T x^\star/\tau^\star - b^T y^\star/\tau^\star &= 0
\end{aligned}
$$

By courtesy of Santiago Akle, Stanford University

Detecting Unboundedness & Infeasibility

$$A x^\star = 0$$
$$A^T y^\star + s^\star = 0$$
$$b^T y^\star - c^T x^\star = \kappa^\star > 0$$

When $\kappa^\star > 0$ and $c^T x^\star < 0$ the primal is unbounded

because

$$A\,\delta x = 0$$
$$c^T\,\delta x < 0$$

When $\kappa^\star > 0$ and $b^T y^\star > 0$ the primal is infeasible (dual unbounded)

$$A^T\,\delta y + \delta s = 0$$
$$b^T\,\delta y > 0$$

By courtesy of Santiago Akle, Stanford University

Self-dual Homogeneous Embedding

minimize 0

subject to

$$\begin{pmatrix} & A & -b \\ -A^T & & c \\ b^T & -c^T & \end{pmatrix} \begin{pmatrix} y \\ x \\ \tau \end{pmatrix} - \begin{pmatrix} 0 \\ s \\ \kappa \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

$$x \in \mathcal{K} \text{ and } s \in \mathcal{K}^\star, \ \tau \geq 0, \ \kappa \geq 0$$

Zero is a solution, but it is not the only solution!

Any feasible point satisfies

- $x^T s + \tau\kappa = 0$
- $x^T s = 0$
- $\tau\kappa = 0$

When $\tau > 0$ then $\kappa = 0$

When $\kappa > 0$ then $\tau = 0$

By courtesy of Santiago Akle, Stanford University

Conic Solvers N/A for Embedded Sys.

- Free solvers such as SeDuMi, SDPT3 and CVXOPT
  - require runtime environments (MATLAB or Python)
  - require external libraries (LAPACK/BLAS)
  - slow for "small" problems

- Commercial solvers such as Gurobi and MOSEK
  - do not run on embedded platforms (proprietary binaries)
  - incur licensing costs
  - code size (binary): Gurobi: 2.7 MB, MOSEK: 7.9 MB

- Performance of first order solvers (e.g. FiOrdOs) problem dependent

Conic Solvers N/A for Embedded Sys.

- Free solvers such as SeDuMi, SDPT3 and CVXOPT
  - require runtime environments (MATLAB or Python)
  - require external libraries (LAPACK/BLAS)
  - slow for "small" problems
- Commercial solvers such as Gurobi and MOSEK
  - do not run on embedded platforms (proprietary binaries)
  - incur licensing costs
  - code size (binary): Gurobi: 2.7 MB, MOSEK: 7.9 MB
- Performance of first order solvers (e.g. FiOrdOs) problem dependent

ECOS fills this gap

Recall: Path-following IPMs

Primal & Dual SOCP Problem

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax = b \\ & Gx + s = h, \; s \in \mathbf{K} \end{array}$$

$$\begin{array}{ll} \text{maximize} & -b^T y - h^T z \\ \text{subject to} & A^T y + G^T z = -c \\ & z \in \mathbf{K} \end{array}$$

[Vandenberghe et al., 2010], [Nesterov & Todd, 1997], [Schmieta & Alizadeh, 2003]

Recall: Path-following IPMs

Primal & Dual SOCP Problem

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax = b \\ & Gx + s = h,\, s \in \mathbf{K} \end{array}$$

$$\begin{array}{ll} \text{maximize} & -b^T y - h^T z \\ \text{subject to} & A^T y + G^T z = -c \\ & z \in \mathbf{K} \end{array}$$

▼

Optimality Conditions & Central Path

$$\begin{bmatrix} 0 \\ 0 \\ s \end{bmatrix} = \begin{bmatrix} 0 & A^T & G^T \\ -A & 0 & 0 \\ -G & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} c \\ b \\ h \end{bmatrix}$$

$$s \circ z = 0$$

$$(s, z) \succeq_K 0$$

[Vandenberghe et al., 2010], [Nesterov & Todd, 1997], [Schmieta & Alizadeh, 2003]

Recall: Path-following IPMs

Primal & Dual SOCP Problem

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax = b \\ & Gx + s = h,\, s \in \mathbf{K} \end{array}$$

$$\begin{array}{ll} \text{maximize} & -b^T y - h^T z \\ \text{subject to} & A^T y + G^T z = -c \\ & z \in \mathbf{K} \end{array}$$

▼

Optimality Conditions & Central Path

$$\begin{bmatrix} 0 \\ 0 \\ s \end{bmatrix} = \begin{bmatrix} 0 & A^T & G^T \\ -A & 0 & 0 \\ -G & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} c \\ b \\ h \end{bmatrix}$$

$$s \circ z = 0 \quad \rightarrow \quad s \circ z = \mu \mathbf{1}$$

$$(s, z) \succeq_K 0 \quad \rightarrow \quad (s, z) \succ_K 0$$

[Vandenberghe et al., 2010], [Nesterov & Todd, 1997], [Schmieta & Alizadeh, 2003]

Recall: Path-following IPMs

Primal & Dual SOCP Problem

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax = b \\ & Gx + s = h, s \in \mathbf{K} \end{array}$$

$$\begin{array}{ll} \text{maximize} & -b^T y - h^T z \\ \text{subject to} & A^T y + G^T z = -c \\ & z \in \mathbf{K} \end{array}$$

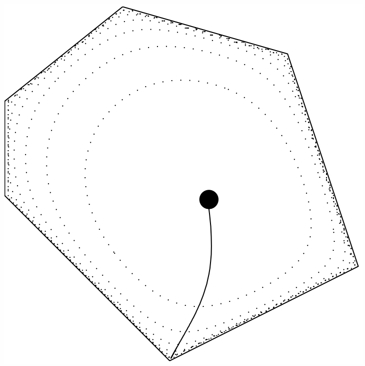
▼

Optimality Conditions & Central Path

$$\begin{bmatrix} 0 \\ 0 \\ s \end{bmatrix} = \begin{bmatrix} 0 & A^T & G^T \\ -A & 0 & 0 \\ -G & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} c \\ b \\ h \end{bmatrix}$$

$$s \circ z = 0 \quad \rightarrow \quad s \circ z = \mu \mathbf{1}$$

$$(s, z) \succeq_K 0 \quad \rightarrow \quad (s, z) \succ_K 0$$



$\mu = 100$

[Vandenberghe et al., 2010], [Nesterov & Todd, 1997], [Schmieta & Alizadeh, 2003]

Recall: Path-following IPMs

Primal & Dual SOCP Problem

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax = b \\ & Gx + s = h, s \in \mathbf{K} \end{array}$$

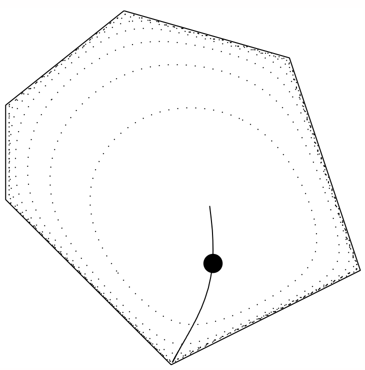
$$\begin{array}{ll} \text{maximize} & -b^T y - h^T z \\ \text{subject to} & A^T y + G^T z = -c \\ & z \in \mathbf{K} \end{array}$$

Optimality Conditions & Central Path

$$\begin{bmatrix} 0 \\ 0 \\ s \end{bmatrix} = \begin{bmatrix} 0 & A^T & G^T \\ -A & 0 & 0 \\ -G & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} c \\ b \\ h \end{bmatrix}$$

$$s \circ z = 0 \quad \rightarrow \quad s \circ z = \mu \mathbf{1}$$

$$(s, z) \succeq_K 0 \quad \rightarrow \quad (s, z) \succ_K 0$$



$\mu = 1$

embotech*

[Vandenberghe et al., 2010], [Nesterov & Todd, 1997], [Schmieta & Alizadeh, 2003]

Recall: Path-following IPMs

Primal & Dual SOCP Problem

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax = b \\ & Gx + s = h,\, s \in \mathbf{K} \end{array}$$

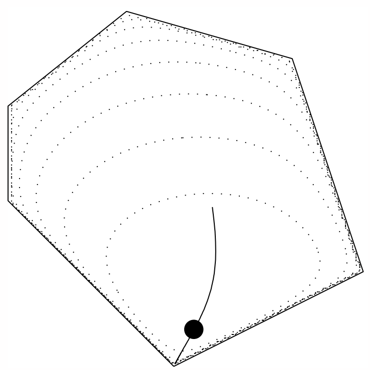
$$\begin{array}{ll} \text{maximize} & -b^T y - h^T z \\ \text{subject to} & A^T y + G^T z = -c \\ & z \in \mathbf{K} \end{array}$$

Optimality Conditions & Central Path

$$\begin{bmatrix} 0 \\ 0 \\ s \end{bmatrix} = \begin{bmatrix} 0 & A^T & G^T \\ -A & 0 & 0 \\ -G & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} c \\ b \\ h \end{bmatrix}$$

$$s \circ z = 0 \quad \rightarrow \quad s \circ z = \mu \mathbf{1}$$

$$(s, z) \succeq_K 0 \quad \rightarrow \quad (s, z) \succ_K 0$$



$\mu = 0.1$

[Vandenberghe et al., 2010], [Nesterov & Todd, 1997], [Schmieta & Alizadeh, 2003]

Recall: Path-following IPMs

Primal & Dual SOCP Problem

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax = b \\ & Gx + s = h,\, s \in \mathbf{K} \end{array}$$

$$\begin{array}{ll} \text{maximize} & -b^T y - h^T z \\ \text{subject to} & A^T y + G^T z = -c \\ & z \in \mathbf{K} \end{array}$$

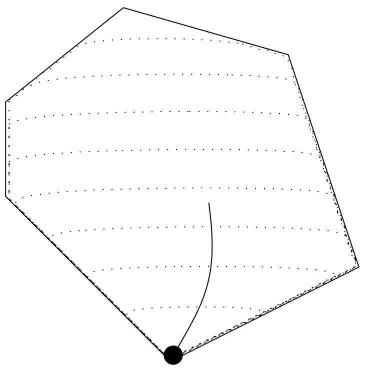
▼

Optimality Conditions & Central Path

$$\begin{bmatrix} 0 \\ 0 \\ s \end{bmatrix} = \begin{bmatrix} 0 & A^T & G^T \\ -A & 0 & 0 \\ -G & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} c \\ b \\ h \end{bmatrix}$$

$$s \circ z = 0 \quad \rightarrow \quad s \circ z = \mu \mathbf{1}$$

$$(s, z) \succeq_K 0 \quad \rightarrow \quad (s, z) \succ_K 0$$



$\mu = 0.01$

[Vandenberghe et al., 2010], [Nesterov & Todd, 1997], [Schmieta & Alizadeh, 2003]

Recall: Path-following IPMs

Primal & Dual SOCP Problem

$$
\begin{array}{ll}
\text{minimize} & c^T x \\
\text{subject to} & Ax = b \\
 & Gx + s = h,\, s \in \mathbf{K}
\end{array}
$$

$$
\begin{array}{ll}
\text{maximize} & -b^T y - h^T z \\
\text{subject to} & A^T y + G^T z = -c \\
 & z \in \mathbf{K}
\end{array}
$$

Optimality Conditions & Central Path

$$
\begin{bmatrix} 0 \\ 0 \\ s \end{bmatrix} = \begin{bmatrix} 0 & A^T & G^T \\ -A & 0 & 0 \\ -G & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} c \\ b \\ h \end{bmatrix}
$$

$$s \circ z = 0 \quad \rightarrow \quad s \circ z = \mu \mathbf{1}$$

$$(s, z) \succeq_K 0 \quad \rightarrow \quad (s, z) \succ_K 0$$

Path-following Method

1. Initialize variables $\xi \triangleq (x, y, z, s)$
2. **Obtain search direction** $\Delta\xi$
3. Line search for step size α
4. Update variables $\xi \leftarrow \xi + \alpha\Delta\xi$
5. Reduce path parameter μ, go to 2.

embotech*

[Vandenberghe et al., 2010], [Nesterov & Todd, 1997], [Schmieta & Alizadeh, 2003]

Spinoff ETHzürich

Recall: Path-following IPMs

Primal & Dual SOCP Problem

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax = b \\ & Gx + s = h,\, s \in \mathbf{K} \end{array}$$

$$\begin{array}{ll} \text{maximize} & -b^T y - h^T z \\ \text{subject to} & A^T y + G^T z = -c \\ & z \in \mathbf{K} \end{array}$$

Search Direction Computation

Solve linearized central path equation:

$$\begin{bmatrix} 0 \\ 0 \\ \Delta s \end{bmatrix} - \begin{bmatrix} 0 & A^T & G^T \\ -A & 0 & 0 \\ -G & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} = \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix}$$

$$W \Delta z + W^{-1} \Delta s = b_s$$

Optimality Conditions & Central Path

$$\begin{bmatrix} 0 \\ 0 \\ s \end{bmatrix} = \begin{bmatrix} 0 & A^T & G^T \\ -A & 0 & 0 \\ -G & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} c \\ b \\ h \end{bmatrix}$$

$$s \circ z = 0 \quad \rightarrow \quad s \circ z = \mu \mathbf{1}$$

$$(s, z) \succeq_K 0 \quad \rightarrow \quad (s, z) \succ_K 0$$

Path-following Method

1. Initialize variables $\xi \triangleq (x, y, z, s)$
2. **Obtain search direction** $\Delta \xi$
3. Line search for step size α
4. Update variables $\xi \leftarrow \xi + \alpha \Delta \xi$
5. Reduce path parameter μ, go to 2.

Recall: Path-following IPMs

Primal & Dual SOCP Problem

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax = b \\ & Gx + s = h,\, s \in \mathbf{K} \end{array}$$

$$\begin{array}{ll} \text{maximize} & -b^T y - h^T z \\ \text{subject to} & A^T y + G^T z = -c \\ & z \in \mathbf{K} \end{array}$$

▼

Optimality Conditions & Central Path

$$\begin{bmatrix} 0 \\ 0 \\ s \end{bmatrix} = \begin{bmatrix} 0 & A^T & G^T \\ -A & 0 & 0 \\ -G & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} c \\ b \\ h \end{bmatrix}$$

$$s \circ z = 0 \quad \rightarrow \quad W^{-1} s \circ W z = \mu \mathbf{1}$$

$$(s, z) \succeq_K 0 \quad \rightarrow \quad (s, z) \succ_K 0$$

►

Search Direction Computation

Solve linearized central path equation:

$$\begin{bmatrix} 0 \\ 0 \\ \Delta s \end{bmatrix} - \begin{bmatrix} 0 & A^T & G^T \\ -A & 0 & 0 \\ -G & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} = \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix}$$

$$W \Delta z + W^{-1} \Delta s = b_s$$

▼▲

Path-following Method

1. Initialize variables $\xi \triangleq (x, y, z, s)$
2. **Obtain search direction** $\Delta\xi$
3. Line search for step size α
4. Update variables $\xi \leftarrow \xi + \alpha \Delta \xi$
5. Reduce path parameter μ, go to 2.

[Vandenberghe et al., 2010], [Nesterov & Todd, 1997], [Schmieta & Alizadeh, 2003]

Search Direction Computation

- Per iteration, solve up to 3 linear systems with indefinite coefficient matrix:

$$\underbrace{\begin{bmatrix} 0 & A^T & G^T \\ A & 0 & 0 \\ G & 0 & -W^2 \end{bmatrix}}_{K} \underbrace{\begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix}}_{x} = \underbrace{\begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix}}_{b} \qquad \text{(SD)}$$

- Common approach: Cholesky factorization of reduced system

$$A(GW^{-2}G^T)^{-1}A^T\Delta y = A(GW^{-2}G^T)^{-1}(b_x + G^TW^{-2}b_z) - b_y$$

  - implemented in FORCES, MOSEK, CVXOPT, SeDuMi, and many others
  - potentially slow if dense columns in A or G are present
  - additional code needed (e.g. low rank modifications)

- **In ECOS**: sparse LDL factorization directly of (SD):
  - sparsity exploitation in A and G
  - small and efficient code

On Sparse LDL Factorization

- Direct approach for solving $Kx = b$ with indefinite K :

$$PKP^T = LDL^T$$

$$u = L\backslash(Pb),\ v = D\backslash u,\ w = L^T\backslash v,\ x = P^T w$$

where P is chosen to obtain sparse triangular L and D is (block-)diagonal

On Sparse LDL Factorization

- Direct approach for solving $Kx = b$ with indefinite K :

$$PKP^T = LDL^T$$
$$u = L\backslash(Pb),\; v = D\backslash u,\; w = L^T\backslash v,\; x = P^T w$$

  where P is chosen to obtain sparse triangular L and D is (block-)diagonal

- Standard LDL requires pivoting for numerical stability *[Bunch & Parlett, 1976]*
  - effective elimination ordering P is data dependent
  - requires complex code and is potentially slow

On Sparse LDL Factorization

- Direct approach for solving $Kx = b$ with indefinite K :

$$PKP^T = LDL^T$$

$$u = L\backslash(Pb),\ \ v = D\backslash u,\ \ w = L^T\backslash v,\ \ x = P^T w$$

where P is chosen to obtain sparse triangular L and D is (block-)diagonal

- Standard LDL requires pivoting for numerical stability *[Bunch & Parlett, 1976]*
  - effective elimination ordering P is data dependent
  - requires complex code and is potentially slow

- Theorem *[Vanderbei, 1994]* If K is **quasi-definite**,

$$PKP^T = LDL^T$$

can be computed stably for all permutations P.

Quasi-definite Matrix

$$\begin{bmatrix} H & F^T \\ F & -E \end{bmatrix}$$ with $H, E \quad 0$

On Sparse LDL Factorization

- Direct approach for solving $Kx = b$ with indefinite K :

$$PKP^T = LDL^T$$
$$u = L\backslash(Pb), \;\; v = D\backslash u, \;\; w = L^T\backslash v, \;\; x = P^T w$$

  where P is chosen to obtain sparse triangular L and D is (block-)diagonal

- Standard LDL requires pivoting for numerical stability *[Bunch & Parlett, 1976]*
  - effective elimination ordering P is data dependent
  - requires complex code and is potentially slow

- Theorem *[Vanderbei, 1994]* If K is **quasi-definite**,

$$PKP^T = LDL^T$$

  can be computed stably for all permutations P.

Quasi-definite Matrix

$$\begin{bmatrix} H & F^T \\ F & -E \end{bmatrix}$$ with $H, E \quad 0$

- Advantage: Ordering P fixed, D diagonal, factorization code ~20 lines of C

On Sparse LDL Factorization

- Direct approach for solving $Kx = b$ with indefinite K :

$$PKP^T = LDL^T$$
$$u = L\backslash(Pb),\ v = D\backslash u,\ w = L^T\backslash v,\ x = P^T w$$

where P is chosen to obtain sparse triangular L and D is (block-)diagonal

- Standard LDL requires pivoting for numerical stability *[Bunch & Parlett, 1976]*
  - effective elimination ordering P is data dependent
  - requires complex code and is potentially slow

- Theorem *[Vanderbei, 1994]* If K is **quasi-definite**,

$$PKP^T = LDL^T$$

can be computed stably for all permutations P.

Quasi-definite Matrix

$$\begin{bmatrix} H & F^T \\ F & -E \end{bmatrix}$$ with $H, E \quad 0$

- Advantage: Ordering P fixed, D diagonal, factorization code ~20 lines of C

Issue: K is not quasi-definite

Obtaining a Quasi-definite KKT Matrix

- Regularization makes K quasi-definite:

$$K = \begin{bmatrix} 0 & A^T & G^T \\ A & 0 & 0 \\ G & 0 & -W^2 \end{bmatrix} \rightarrow \left[\begin{array}{c|cc} +\delta I & A^T & G^T \\ \hline A & -\delta I & 0 \\ G & 0 & -W^2 \end{array}\right] = \tilde{K}$$

where $\delta \approx 10^{-6} \ldots 10^{-8}$. Solving $\tilde{K}\tilde{x} = b$ is stable for any permutation

- Iterative refinement recovers true solution x from $\tilde{x}$ in a few steps:

Set	$x \leftarrow \tilde{x}$
compute	$e = b - Kx$
solve	$\tilde{K}d = e$
update	$x \leftarrow x + d$

[Arioli, Demmel & Duff, 1989]

- This is standard in many solvers (e.g. CVXGEN, PDCO, ...)

Obtaining a Quasi-definite KKT Matrix

- Regularization makes K quasi-definite:

$$K = \begin{bmatrix} 0 & A^T & G^T \\ A & 0 & 0 \\ G & 0 & -W^2 \end{bmatrix} \rightarrow \left[\begin{array}{c|cc} +\delta I & A^T & G^T \\ \hline A & -\delta I & 0 \\ G & 0 & -W^2 \end{array}\right] = \tilde{K}$$

where $\delta \approx 10^{-6} \ldots 10^{-8}$. Solving $\tilde{K}\tilde{x} = b$ is stable for any permutation

- Iterative refinement recovers true solution x from $\tilde{x}$ in a few steps:

Set	$x \leftarrow \tilde{x}$
compute	$e = b - Kx$
solve	$\tilde{K}d = e$
update	$x \leftarrow x + d$

[Arioli, Demmel & Duff, 1989]

- This is standard in many solvers (e.g. CVXGEN, PDCO, ...)

Issue: Matrix W is dense for second-order cones $\Rightarrow$ method slow for "large" cones

Portfolio Minimization Example

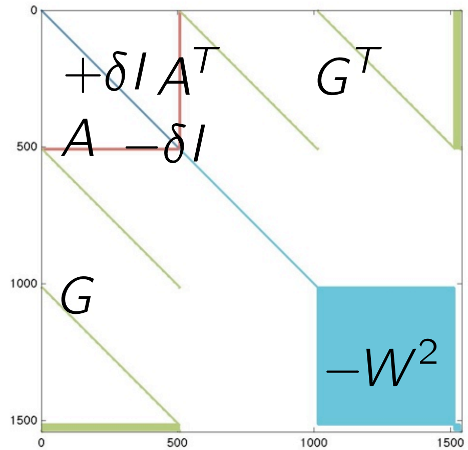
- Maximize risk-adjusted return for portfolio investments x:

$$\begin{aligned} \max \quad & \mu^T x - \gamma(x^T \Sigma x) \\ \text{s.t.} \quad & \mathbf{1}^T x = 1 \\ & x \geq 0 \end{aligned}$$

- Risk covariance matrix is in factor model form: $\Sigma = D + FF^T$ *[Boyd & Vandenberghe, 2004]*
  D diagonal, $F \in \mathbf{R}^{n \times m}$, $m \ll n$ fairly large SOC cones

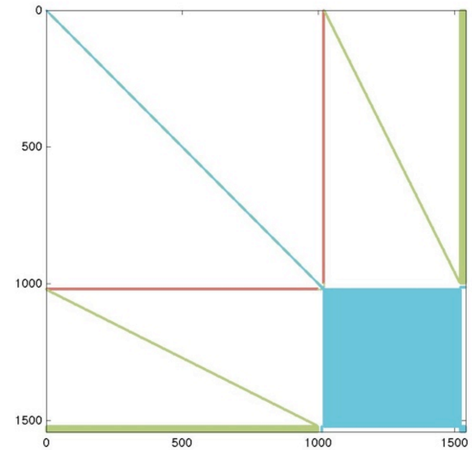
- Sparsity pattern:

K



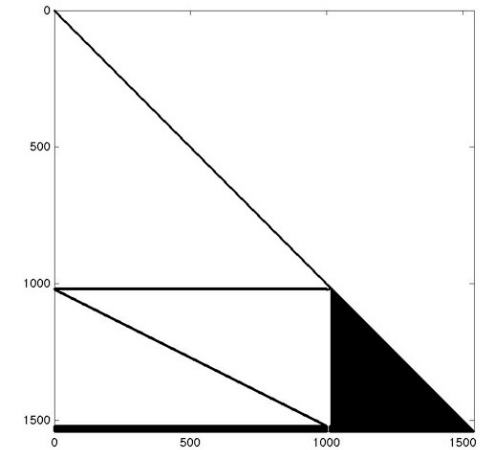
nz = 251’964

PKP^T



nz = 251’964

Factor L



nz = 158’584

Nesterov-Todd Scalings for Large Steps

- An invertible matrix W is called **scaling** if it preserves the conic inequalities:

$$s \in \operatorname{int} \mathbf{K} \Leftrightarrow Ws \in \operatorname{int} \mathbf{K} \Leftrightarrow W^T s \in \operatorname{int} \mathbf{K} \quad \forall s \in \operatorname{int} \mathbf{K}$$

- Scaling for product cone is block-diagonal:

$$W = \operatorname{blkdiag}(W_1, \dots, W_N) \text{ for } s, z \in \mathbf{K} = \mathbf{K}_1 \times \cdots \times \mathbf{K}_N$$

- Nesterov-Todd scaling yields **large step sizes** - used in most solvers:
  - $s, z \in \mathbf{R}^n_{++}: \; W_{\mathrm{R}_{++}} = \operatorname{diag}(s)/\operatorname{diag}(z) = SZ^{-1}$ (standard directions for LPs/QPs)
  - $s, z \in \operatorname{int} \mathbf{Q}^n: \; W_{\mathrm{Q}} = \eta(qq^T - J), \; J = \begin{bmatrix} 1 & 0 \\ 0 & -I \end{bmatrix}$ for scalar $\eta(s,z)$ and vector $q(s,z)$

[Nesterov & Todd, 1997]

Nesterov-Todd Scalings for Large Steps

- An invertible matrix W is called **scaling** if it preserves the conic inequalities:

$$s \in \operatorname{int} \mathbf{K} \Leftrightarrow Ws \in \operatorname{int} \mathbf{K} \Leftrightarrow W^T s \in \operatorname{int} \mathbf{K} \quad \forall s \in \operatorname{int} \mathbf{K}$$

- Scaling for product cone is block-diagonal:

$$W = \operatorname{blkdiag}(W_1, \ldots, W_N) \text{ for } s, z \in \mathbf{K} = \mathbf{K}_1 \times \cdots \times \mathbf{K}_N$$

- Nesterov-Todd scaling yields **large step sizes** - used in most solvers:
  - $s, z \in \mathbf{R}^n_{++}$: $W_{\mathrm{R}_{++}} = \operatorname{diag}(s)/\operatorname{diag}(z) = SZ^{-1}$ (standard directions for LPs/QPs)
  - $s, z \in \operatorname{int} \mathbf{Q}^n$: $W_{\mathrm{Q}} = \eta(qq^T - J)$, $J = \begin{bmatrix} 1 & 0 \\ 0 & -I \end{bmatrix}$ for scalar $\eta(s, z)$ and vector $q(s, z)$

[Nesterov & Todd, 1997]

Approach: Exploit diagonal + rank 1 structure of SOC scaling

Stable Sparse Expansion of KKT Matrix

- **Main Result** The square of scaling matrix, W^2, can be rewritten as

$$W^2 = D + uu^T - vv^T$$

for carefully chosen diagonal D and vectors u and v such that the matrix

$$\left[\begin{array}{cc|c} D & v & u \\ v^T & 1 & 0 \\ \hline u^T & 0 & -1 \end{array}\right]$$

is quasi-definite.

- Consequence: SOC blocks in K can be safely expanded into sparse form:

$$\begin{bmatrix} +\delta I & A^T & G^T \\ A & -\delta I & 0 \\ G & 0 & -W^2 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} = \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} \Leftrightarrow \begin{bmatrix} +\delta I & A^T & G^T & 0 & 0 \\ A & -\delta I & 0 & 0 & 0 \\ G & 0 & -D & -v & -u \\ 0 & 0 & -v^T & -1 & 0 \\ 0 & 0 & -u^T & 0 & 1 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \\ t_1 \\ t_2 \end{bmatrix} = \begin{bmatrix} b_x \\ b_y \\ b_z \\ 0 \\ 0 \end{bmatrix}$$

- New KKT matrix is quasi-definite

Stable Sparse Expansion of KKT Matrix

- **Main Result** The square of scaling matrix, W^2, can be rewritten as

$$W^2 = D + uu^T - vv^T$$

for carefully chosen diagonal D and vectors u and v such that the matrix

$$\left[\begin{array}{cc|c} D & v & u \\ v^T & 1 & 0 \\ \hline u^T & 0 & -1 \end{array}\right]$$

is quasi-definite.

- Consequence: SOC blocks in K can be safely expanded into sparse form:

$$\begin{bmatrix} +\delta I & A^T & G^T \\ A & -\delta I & 0 \\ G & 0 & -W^2 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} = \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} \Leftrightarrow \left[\begin{array}{cc|ccc} +\delta I & 0 & A^T & G^T & 0 \\ 0 & 1 & 0 & -u^T & 0 \\ \hline A & 0 & -\delta I & 0 & 0 \\ G & -u & 0 & -D & -v \\ 0 & 0 & 0 & -v^T & -1 \end{array}\right] \begin{bmatrix} \Delta x \\ t_2 \\ \Delta y \\ \Delta z \\ t_1 \end{bmatrix} = \begin{bmatrix} b_x \\ 0 \\ b_y \\ b_z \\ 0 \end{bmatrix}$$

- New KKT matrix is quasi-definite

embotech*

Spinoff ETH zürich

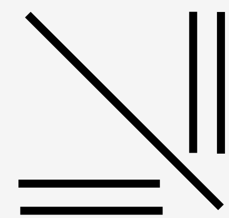
Stable Sparse Expansion of KKT Matrix

- **Main Result** The square of scaling matrix, W^2, can be rewritten as

$$W^2 = D + uu^T - vv^T$$

for carefully chosen diagonal D and vectors u and v such that the matrix

$$\left[\begin{array}{cc|c} D & v & u \\ v^T & 1 & 0 \\ \hline u^T & 0 & -1 \end{array}\right]$$



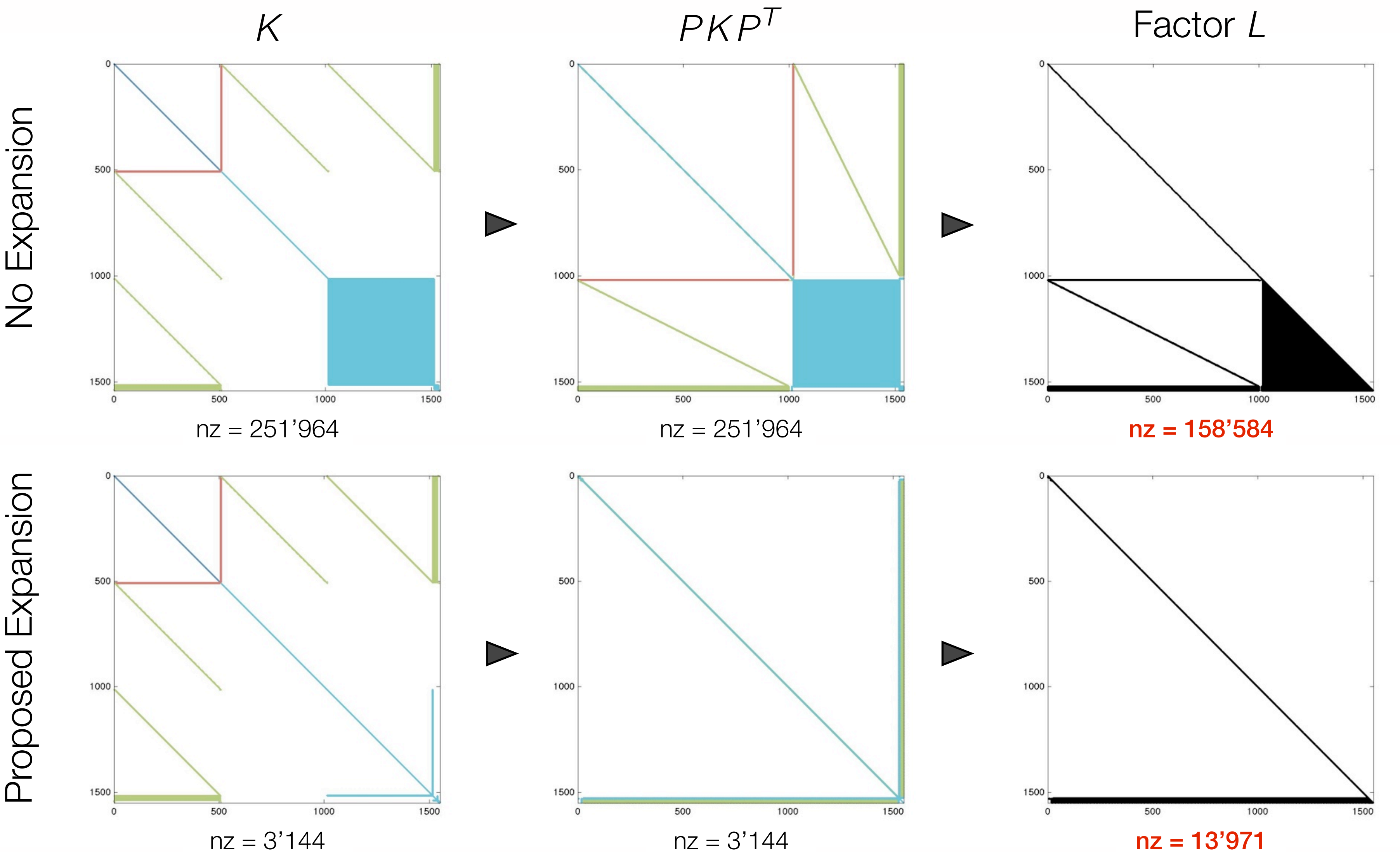
is quasi-definite.

- Consequence: SOC blocks in K can be safely expanded into sparse form:

$$\begin{bmatrix} +\delta I & A^T & G^T \\ A & -\delta I & 0 \\ G & 0 & -W^2 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} = \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} \Leftrightarrow \left[\begin{array}{cc|ccc} +\delta I & 0 & A^T & G^T & 0 \\ 0 & 1 & 0 & -u^T & 0 \\ \hline A & 0 & -\delta I & 0 & 0 \\ G & -u & 0 & -D & -v \\ 0 & 0 & 0 & -v^T & -1 \end{array}\right] \begin{bmatrix} \Delta x \\ t_2 \\ \Delta y \\ \Delta z \\ t_1 \end{bmatrix} = \begin{bmatrix} b_x \\ 0 \\ b_y \\ b_z \\ 0 \end{bmatrix}$$

- New KKT matrix is quasi-definite

Effect of Expansion for Portfolio Problem



embotech*

Spinoff ETH zürich

Embedded Conic Solver

github SOCIAL CODING .com/embotech/ecos

- Primal-dual Mehrotra IPM with Nesterov-Todd scalings
- **Detects infeasibility**
- **ANSI C implementation**
  - Solve has **~800 lines of code**, including all linear algebra code
  - size of binary: ~110 KB
  - library free
  - safe divisions
- **Interfaces:**
  - Native: C, Matlab, Python, Julia, MLlib
  - Modeling: CVX, Yalmip, QCML, CVXPY
  - With simple branch-and-bound

- Divided into 3 functions:

Setup
- Allocate memory
- Determine elimination ordering

▼ can be generated

Solve
- Return certificate of optimality or infeasibility

new problem data

▼

Cleanup
- Free memory

Portfolio Benchmark

- Maximize risk-adjusted return for portfolio investments x:

$$\begin{array}{rl} \max & \mu^T x - \gamma(x^T \Sigma x) \\ \text{s.t.} & \mathbf{1}^T x = 1 \\ & x \geq 0 \end{array}$$

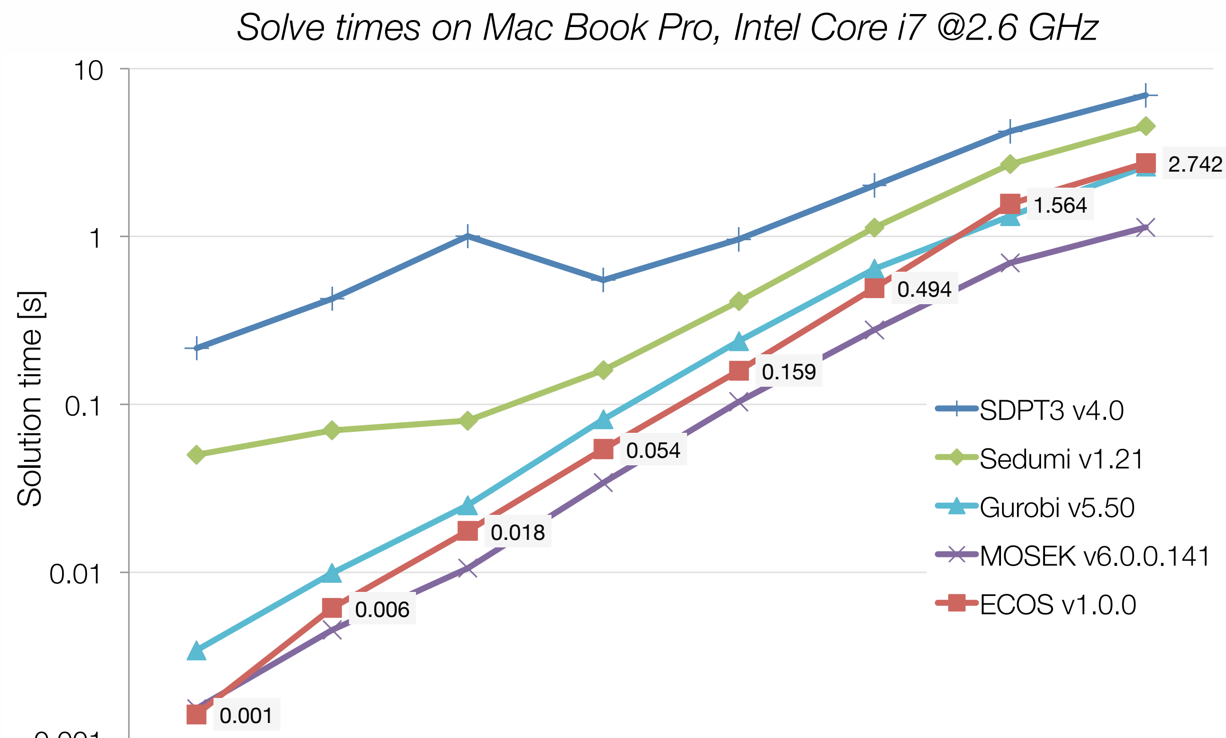
[Boyd & Vandenberghe, 2004]

- Risk covariance matrix is in factor model form:

$$\Sigma = D + FF^T$$

D diagonal, $F \in \mathbf{R}^{n \times m}$, $m \ll n$

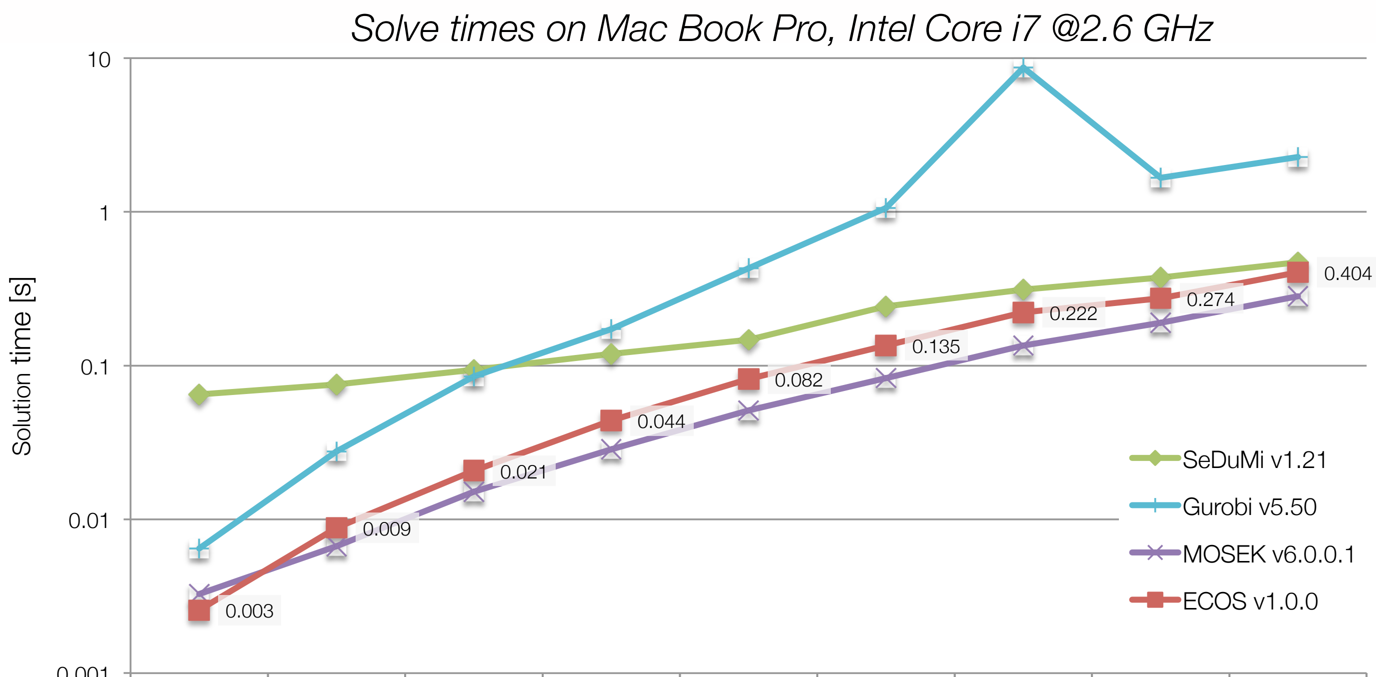
- Converting to SOCP yields large cone sizes



#variables	217	622	1032	2042	4052	8062	15072	20082
#factors	5	10	20	30	40	50	60	70
#assets	100	300	500	1000	2000	4000	7500	10000

Soft-constrained MPC Benchmark

- Relax state constraints but guarantee stability *[Zeilinger et al., 2013]*
- Many second-order cone constraints of small dimension

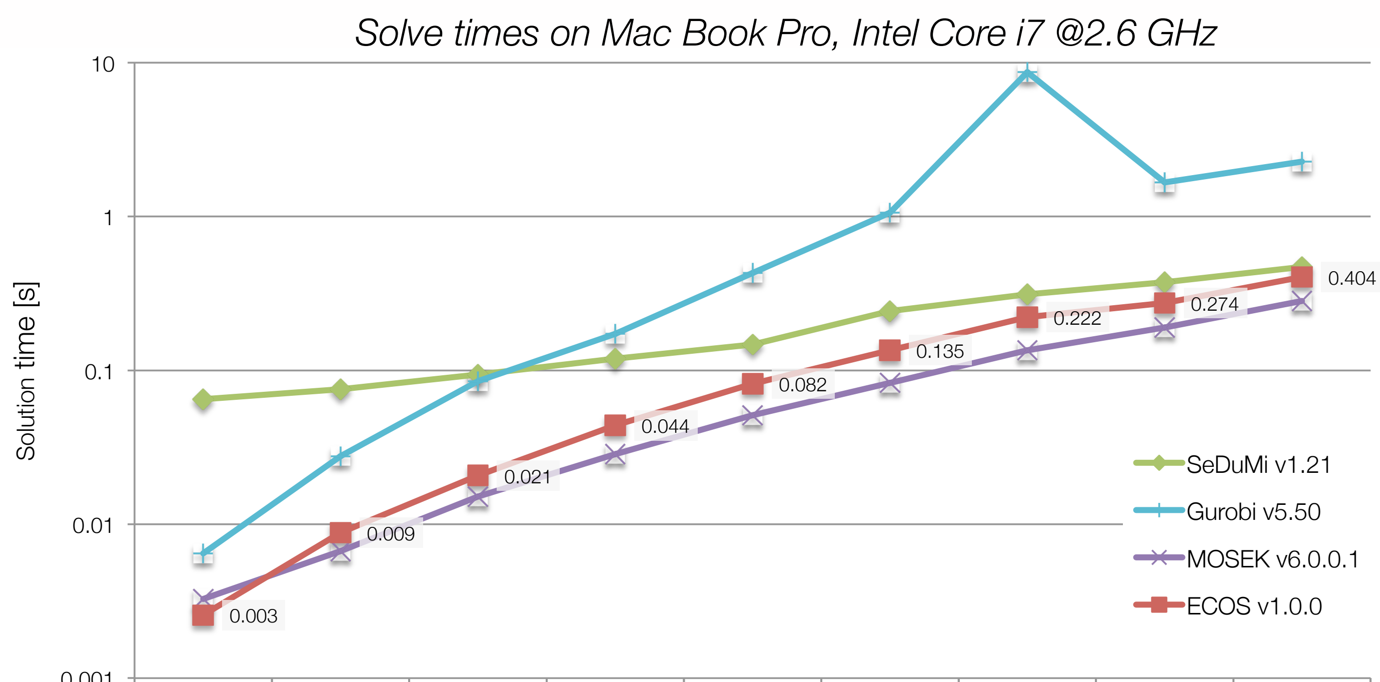


non-zeros in [A;G]	861	2844	6693	15208	26411	42113	63085	90090	123901
SOC constraints	10x3 | 10x6 | 10x3 | 1x10 | 8x5	10x5 | 10x10 | 10x3 | 1x20 | 16x9	10x7 | 10x14 | 10x3 | 1x30 | 24x13	10x9 | 10x18 | 10x3 | 1x40 | 32x17	10x11 | 10x22 | 10x3 | 1x50 | 40x21	10x13 | 10x26 | 10x3 | 1x60 | 48x25	10x15 | 10x30 | 10x3 | 1x70 | 56x29	10x17 | 10x34 | 10x3 | 1x80 | 64x33	10x19 | 10x38 | 10x3 | 1x90 | 72x37
Linear constraints	120	252	384	516	648	780	912	1044	1176
# of variables	97	163	229	295	361	427	493	559	625
# of inputs	1	3	5	7	9	11	13	15	17
# of states	4	8	12	16	20	24	28	32	36

Soft-constrained MPC Benchmark

- Relax state constraints but guarantee stability *[Zeilinger et al., 2013]*

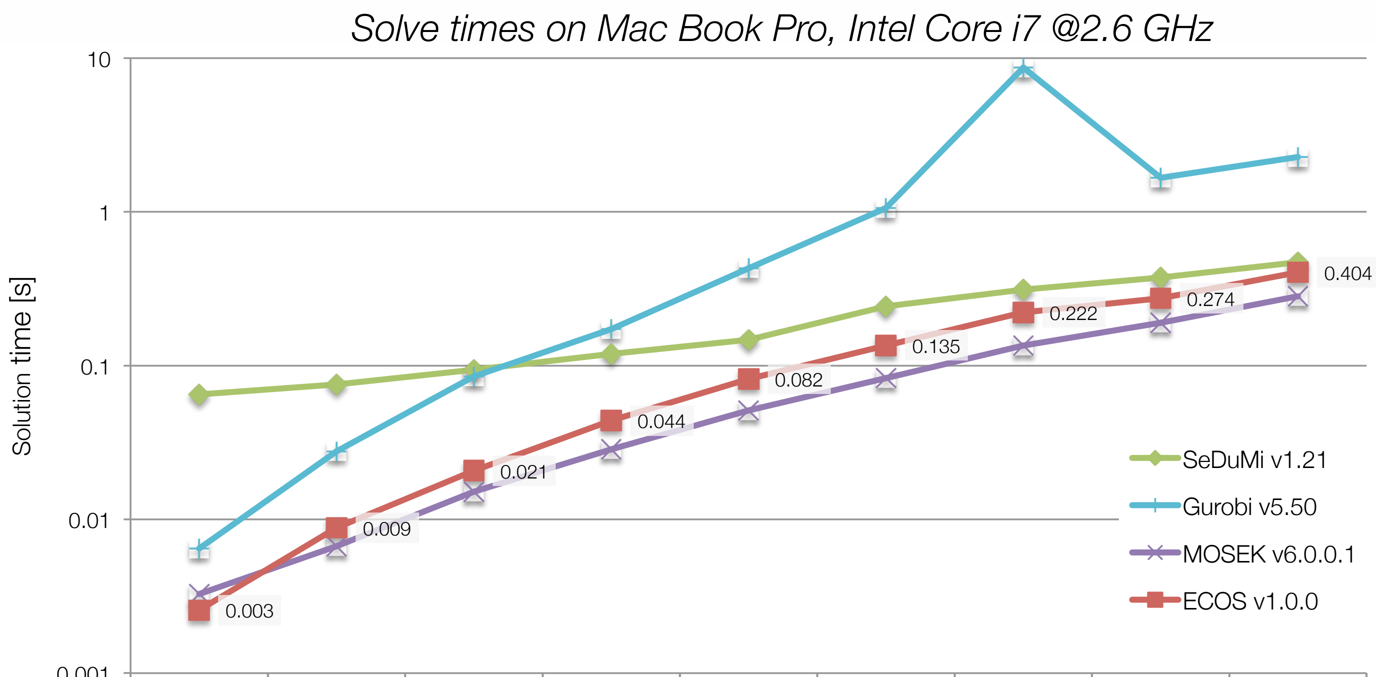
- Many second-order cone constraints of small dimension



non-zeros in [A;G]	861	2844	6693	15208	26411	42113	63085	90090	123901
SOC constraints	10x3 | 10x6 | 10x3 | 1x10 | 8x5	10x5 | 10x10 | 10x3 | 1x20 | 16x9	10x7 | 10x14 | 10x3 | 1x30 | 24x13	10x9 | 10x18 | 10x3 | 1x40 | 32x17	10x11 | 10x22 | 10x3 | 1x50 | 40x21	10x13 | 10x26 | 10x3 | 1x60 | 48x25	10x15 | 10x30 | 10x3 | 1x70 | 56x29	10x17 | 10x34 | 10x3 | 1x80 | 64x33	10x19 | 10x38 | 10x3 | 1x90 | 72x37
Linear constraints	120	252	384	516	648	780	912	1044	1176
# of variables	97	163	229	295	361	427	493	559	625
# of inputs	1	3	5	7	9	11	13	15	17
# of states	4	8	12	16	20	24	28	32	36

Soft-constrained MPC Benchmark

- Relax state constraints but guarantee stability *[Zeilinger et al., 2013]*
- Many second-order cone constraints of small dimension



non-zeros in [A;G]	861	2844	6693	15208	26411	42113	63085	90090	123901
SOC constraints	10x3 | 10x6 | 10x3 | 1x10 | 8x5	10x5 | 10x10 | 10x3 | 1x20 | 16x9	10x7 | 10x14 | 10x3 | 1x30 | 24x13	10x9 | 10x18 | 10x3 | 1x40 | 32x17	10x11 | 10x22 | 10x3 | 1x50 | 40x21	10x13 | 10x26 | 10x3 | 1x60 | 48x25	10x15 | 10x30 | 10x3 | 1x70 | 56x29	10x17 | 10x34 | 10x3 | 1x80 | 64x33	10x19 | 10x38 | 10x3 | 1x90 | 72x37
Linear constraints	120	252	384	516	648	780	912	1044	1176
# of variables	97	163	229	295	361	427	493	559	625
# of inputs	1	3	5	7	9	11	13	15	17
# of states	4	8	12	16	20	24	28	32	36

Competitive computation times, but embeddable

Exercise Session

- Two Tasks:
  - TASK 1: **ECOS as sparse QP solver**
    - Errata: The hint should read

$$\left\{(t,x) \mid \frac{1}{2}x^T W^T W x + q^T x \leq t\right\} = \left\{(t,x) \mid \left\| \begin{matrix} Wx \\ \frac{t - q^T x - 1}{\sqrt{2}} \end{matrix} \right\|_2 \leq \frac{t - q^T x + 1}{\sqrt{2}}\right\}$$

  - TASK 2: **Thrust allocation problem**