Conic Optimization via Operator Splitting and Homogeneous Self-Dual Embedding

B. O’Donoghue E. Chu N. Parikh S. Boyd

Convex Optimization and Beyond, Edinburgh, 11/6/2104
Cone programming

Homogeneous embedding

Operator splitting

Numerical results

Conclusions
Cone programming

\[
\begin{align*}
\text{minimize} & \quad c^T x \\
\text{subject to} & \quad Ax + s = b, \quad s \in \mathcal{K}
\end{align*}
\]

- variables \( x \in \mathbb{R}^n \) and (slack) \( s \in \mathbb{R}^m \)
- \( \mathcal{K} \) is a proper convex cone
  - \( \mathcal{K} \) nonnegative orthant \( \rightarrow \) LP
  - \( \mathcal{K} \) Lorentz cone \( \rightarrow \) SOCP
  - \( \mathcal{K} \) positive semidefinite matrices \( \rightarrow \) SDP
- the ‘modern’ canonical form for convex optimization
- popularized by Nesterov, Nemirovsky, others, in 1990s
Cone programming

- parser/solvers like CVX, CVXPY, YALMIP translate or canonicalize to cone problems
- focus has been on symmetric self-dual cones
- for medium scale problems with enough sparsity, interior-point methods reliably attain high accuracy
- but they scale superlinearly in problem size
- open source software (SDPT3, SeDuMi, ...) widely used
This talk

a new first order method that

- solves general cone programs
- finds primal and dual solutions, or certificate of primal/dual infeasibility
- obtains modest accuracy quickly
- scales to large problems and is easy parallelized
- is matrix-free: only requires $z \rightarrow Az$, $w \rightarrow A^T w$
Some previous work

- projected subgradient type methods (Polyak 1980s)
- primal-dual subgradient methods (Chambelle-Pock 2011)
- matrix-free interior-point methods (Gondzio 2012)
- can use iterative linear solver (CG) in any interior-point method
Outline

Cone programming

Homogeneous embedding

Operator splitting

Numerical results

Conclusions

Homogeneous embedding
Primal-dual cone problem pair

primal and dual cone problems:

minimize \( c^T x \)  
subject to \( Ax + s = b \)  
\((x, s) \in \mathbb{R}^n \times \mathcal{K} \)

maximize \( -b^T y \)  
subject to \( -A^T y + r = c \)  
\((r, y) \in \{0\}^n \times \mathcal{K}^* \)

- primal variables \( x \in \mathbb{R}^n, s \in \mathbb{R}^m \); dual variables \( r \in \mathbb{R}^n, y \in \mathbb{R}^m \)
- \( \mathcal{K}^* \) is dual of closed convex proper cone \( \mathcal{K} \)
- note that \( \mathbb{R}^n \times \mathcal{K} \) and \( \{0\}^n \times \mathcal{K}^* \) are dual cones
Example cones

\( \mathcal{K} \) is typically a Cartesian product of smaller cones, e.g.,

- \( \mathbb{R}, \{0\}, \mathbb{R}_+ \)
- second-order cone \( Q = \{(x, t) \in \mathbb{R}^{k+1} | \|x\|_2 \leq t\} \)
- positive semidefinite cone \( \{X \in \mathbb{S}^k | X \succeq 0\} \)
- exponential cone \( \text{cl}\{(x, y, z) \in \mathbb{R}^3 | y > 0, \ e^{x/y} \leq z/y\} \)

these cones would handle almost all convex problems that arise in applications
Optimality conditions

KKT conditions (necessary and sufficient, assuming strong duality):

- primal feasibility: $Ax + s = b$, $s \in \mathcal{K}$
- dual feasibility: $A^T y + c = r$, $r = 0$, $y \in \mathcal{K}^*$
- complementary slackness: $y^T s = 0$
  equivalent to zero duality gap: $c^T x + b^T y = 0$
Primal-dual embedding

- KKT conditions as feasibility problem: find

\[(x, s, r, y) \in \mathbb{R}^n \times \mathcal{K} \times \{0\}^n \times \mathcal{K}^*\]

that satisfy

\[
\begin{bmatrix}
  r \\
  s \\
  0
\end{bmatrix} = \begin{bmatrix} 0 & A^T \\ -A & 0 \\ c^T & b^T \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} c \\ b \\ 0 \end{bmatrix}
\]

- reduces solving cone program to finding point in intersection of cone and affine set

- no solution if primal or dual problem infeasible/unbounded
Homogeneous self-dual (HSD) embedding

(Ye, Todd, Mizuno, 1994)

- find nonzero

\[(x, s, r, y) \in \mathbb{R}^n \times \mathcal{K} \times \{0\}^n \times \mathcal{K}^*, \quad \tau \geq 0, \quad \kappa \geq 0\]

that satisfy

\[
\begin{bmatrix}
  r \\
  s \\
  \kappa
\end{bmatrix} =
\begin{bmatrix}
  0 & A^T & c \\
  -A & 0 & b \\
  -c^T & -b^T & 0
\end{bmatrix}
\begin{bmatrix}
  x \\
  y \\
  \tau
\end{bmatrix}
\]

- this feasibility problem is homogeneous and self-dual

- \(\tau = 1, \kappa = 0\) reduces to primal-dual embedding

- due to skew symmetry, any solution satisfies

\[(x, y, \tau) \perp (r, s, \kappa), \quad \tau \kappa = 0\]
any HSD solution \((x, s, r, y, \tau, \kappa)\) falls into one of three cases:

1. \(\tau > 0, \kappa = 0\): \((\hat{x}, \hat{y}, \hat{s}) = (x/\tau, y/\tau, s/\tau)\) is a solution

2. \(\tau = 0, \kappa > 0\): in this case \(c^T x + b^T y < 0\)
   - if \(b^T y < 0\), then \(\hat{y} = y/(-b^T y)\) certifies primal infeasibility
   - if \(c^T x < 0\), then \(\hat{x} = x/(-c^T x)\) certifies dual infeasibility

3. \(\tau = \kappa = 0\): nothing can be said about original problem (a pathology)
Homogeneous primal-dual embedding

HSD embedding

- obviates need for phase I / phase II solves to handle infeasibility/unboundedness
- is used in all interior-point cone solvers
- is a particularly nice form to solve
  (for reasons not completely understood)
Notation

- define

\[
    u = \begin{bmatrix} x \\ y \\ \tau \end{bmatrix}, \quad v = \begin{bmatrix} r \\ s \\ \kappa \end{bmatrix}, \quad Q = \begin{bmatrix} 0 & A^T & c \\ -A & 0 & b \\ -c^T & -b^T & 0 \end{bmatrix}
\]

- HSD embedding is: find \((u, v)\) that satisfy

\[
    v = Qu, \quad (u, v) \in C \times C^*
\]

with \(C = \mathbb{R}^n \times \mathcal{K}^* \times \mathbb{R}_+\)
Outline

Cone programming

Homogeneous embedding

Operator splitting

Numerical results

Conclusions

Operator splitting
Consensus problem

- consensus problem:
  
  minimize \( f(x) + g(z) \)

  subject to \( x = z \)

- \( f, g \) convex, not necessarily smooth, can take infinite values

- \( p^* \) is optimal objective value
Alternating direction method of multipliers

- ADMM is: for $k = 0, \ldots,$

$$x^{k+1} = \arg\min_x \left( f(x) + \left(\frac{\rho}{2}\right)\|x - z^k - \lambda^k\|_2^2\right)$$

$$z^{k+1} = \arg\min_z \left( g(z) + \left(\frac{\rho}{2}\right)\|x^{k+1} - z - \lambda^k\|_2^2\right)$$

$$\lambda^{k+1} = \lambda^k - x^{k+1} + z^{k+1}$$

- $\rho > 0$ step-size
- $\lambda$ (scaled) dual variable for $x = z$ constraint
- same as many other operator splitting methods for consensus problem, e.g., Douglas-Rachford method
Convergence of ADMM

under benign conditions ADMM guarantees:

- $f(x^k) + g(z^k) \to p^*$
- $\lambda^k \to \lambda^*$, an optimal dual variable
- $x^k - z^k \to 0$
ADMM applied to HSD embedding

- HSD in consensus form

\[
\begin{align*}
\text{minimize} & \quad l_{C \times C^*}(u, v) + l_{Q_{\tilde{u} = \tilde{v}}}(\tilde{u}, \tilde{v}) \\
\text{subject to} & \quad (u, v) = (\tilde{u}, \tilde{v})
\end{align*}
\]

\(l_S\) is indicator function of set \(S\)

- ADMM is:

\[
\begin{align*}
(\tilde{u}^{k+1}, \tilde{v}^{k+1}) &= \Pi_{Q_{u = v}}(u^k + \lambda^k, v^k + \mu^k) \\
u^{k+1} &= \Pi_C(\tilde{u}^{k+1} - \lambda^k) \\
v^{k+1} &= \Pi_{C^*}(\tilde{v}^{k+1} - \mu^k) \\
\lambda^{k+1} &= \lambda^k - \tilde{u}^{k+1} + u^{k+1} \\
\mu^{k+1} &= \mu^k - \tilde{v}^{k+1} + v^{k+1}
\end{align*}
\]

\(\Pi_S(x)\) is Euclidean projection of \(x\) onto \(S\)
Simplifications

(straightforward, but not immediate)

- if $\lambda^0 = v^0$ and $\mu^0 = u^0$, then $\lambda^k = v^k$ and $\mu^k = u^k$ for all $k$
- simplify projection onto $Qu = v$ using $Q^T = -Q$
- nothing depends on $\tilde{v}^k$, so can be eliminated
Final algorithm

- for $k = 0, \ldots,$

\[
\begin{align*}
\tilde{u}^{k+1} &= (I + Q)^{-1}(u^k + v^k) \\
u^{k+1} &= \Pi_C (\tilde{u}^{k+1} - v^k) \\
v^{k+1} &= v^k - \tilde{u}^{k+1} + u^{k+1}
\end{align*}
\]

- parameter free
- homogeneous
- same complexity as ADMM applied to primal or dual alone
Variation: Approximate projection

- replace exact projection with any $\tilde{u}^{k+1}$ that satisfies

$$\|\tilde{u}^{k+1} - (I + Q)^{-1}(u^k + v^k)\|_2 \leq \mu^k,$$

where $\mu^k > 0$ satisfy $\sum_k \mu_k < \infty$

- useful when an iterative method is used to compute $\tilde{u}^{k+1}$

- implied by the (more easily verified) inequality

$$\|(Q + I)\tilde{u}^{k+1} - (u^k + v^k)\|_2 \leq \mu^k$$

by skew-symmetry of $Q$
Convergence

can show the following (even with approximate projection):

- for all iterations $k > 0$ we have
  \[ u^k \in C, \quad v^k \in C^*, \quad (u^k)^T v^k = 0 \]
- as $k \to \infty$, 
  \[ Qu^k - v^k \to 0 \]
- with $\tau^0 = 1, \kappa^0 = 1$, $(u^k, v^k)$ bounded away from zero
Solving the linear system

\[ \begin{bmatrix} I & A^T & c \\ -A & I & b \\ -c^T & -b^T & 1 \end{bmatrix} \begin{bmatrix} \tilde{u}_x \\ \tilde{u}_y \\ \tilde{u}_\tau \end{bmatrix} = \begin{bmatrix} w_x \\ w_y \\ w_\tau \end{bmatrix} \]

\( \Rightarrow \) let

\[ M = \begin{bmatrix} I & A^T \\ -A & I \end{bmatrix}, \quad h = \begin{bmatrix} c \\ b \end{bmatrix} \]

so

\[ I + Q = \begin{bmatrix} M & h \\ -h^T & 1 \end{bmatrix} \]

\( \Rightarrow \) it follows that

\[ \begin{bmatrix} \tilde{u}_x \\ \tilde{u}_y \end{bmatrix} = (M + hh^T)^{-1} \left( \begin{bmatrix} w_x \\ w_y \end{bmatrix} - w_\tau h \right), \]
Solving the linear system, contd.

- applying matrix inversion lemma to \((M + hh^T)^{-1}\) yields

\[
\begin{align*}
\begin{bmatrix}
\tilde{u}_x \\
\tilde{u}_y
\end{bmatrix} &= \left( M^{-1} - \frac{M^{-1}hh^TM^{-1}}{1 + h^T M^{-1}h} \right) \begin{bmatrix} w_x \\
w_y \end{bmatrix} - w_\tau h
\end{align*}
\]

and

\[
\tilde{u}_\tau = w_\tau + c^T \tilde{u}_x + b^T \tilde{u}_y
\]

- first compute and cache \(M^{-1}h\)

- so each iteration requires that we compute

\[
M^{-1} \begin{bmatrix} w_x \\
w_y \end{bmatrix}
\]

and perform vector operations with cached quantities
Direct method

- to solve
\[
\begin{bmatrix}
I & -A^T \\
-A & -I
\end{bmatrix}
\begin{bmatrix}
z_x \\
z_y
\end{bmatrix} =
\begin{bmatrix}
w_x \\
w_y
\end{bmatrix}
\]

- compute sparse permuted LDL factorization of matrix

- re-use cached factorization for subsequent solves

- factorization guaranteed to exist for all permutations, since matrix is symmetric quasi-definite
Indirect method

- by elimination

\[ z_x = (I + A^T A)^{-1}(w_x - A^T w_y) \]
\[ z_y = w_y + Az_x \]

- can apply *conjugate gradient (CG)* to first equation

- CG requires only multiplies by \( A \) and \( A^T \)

- terminate CG iterations when residual smaller than \( \mu^k \)

- easily parallelized; can exploit warm-starting

Operator splitting
Scaling / preconditioning

convergence greatly improved by scaling / preconditioning:

- replace original data $A$, $b$, $c$ with $\hat{A} = DAE$, $\hat{b} = Db$, $\hat{c} = Ec$
- $D$ and $E$ are diagonal positive; $D$ respects cone boundaries
- $D$ and $E$ chosen by equilibrating $A$ (details in paper)
- stopping condition retains unscaled (original) data
Outline

Cone programming

Homogeneous embedding

Operator splitting

Numerical results

Conclusions
SCS software package

- available from:
  https://github.com/cvxgrp/scs
- written in C with matlab and python hooks
- can be called from CVX and CVXPY
- solves LPs, SOCPs, ECPs, and SDPs
- includes sparse direct and indirect linear system solvers
- can use single or double precision, ints or longs for indices
Portfolio optimization

- $z \in \mathbb{R}^p$ gives weights of (long-only) portfolio with $p$ assets
- maximize risk-adjusted portfolio return:
  \[
  \text{maximize } \mu^T z - \gamma (z^T \Sigma z)
  \]
  subject to
  \[
  1^T z = 1, \quad z \geq 0
  \]
- $\mu, \Sigma$ are return mean, covariance
- $\gamma > 0$ is risk aversion parameter
- $\Sigma$ given as factor model $\Sigma = FF^T + D$
- $F \in \mathbb{R}^{q \times p}$ is factor loading matrix
- can be transformed to SOCP

Numerical results
## Portfolio optimization results

|                          | 5000   | 50000  | 100000 |
|--------------------------|--------|--------|--------|
| assets $p$               |        |        |        |
| factors $q$              | 50     | 500    | 1000   |
| SOCP variables $n$       | 5002   | 50002  | 100002 |
| SOCP constraints $m$     | 10055  | 100505 | 201005 |
| nonzeros in $A$          | $3.8 \times 10^4$ | $2.5 \times 10^6$ | $1.0 \times 10^7$ |

### SDPT3:
- solve time: 1.14 sec, 17836.7 sec, OOM

### SCS direct:
- solve time: 0.17 sec, 4.7 sec, 37.1 sec
- iterations: 420, 340, 760

### SCS indirect:
- solve time: 0.23 sec, 12.2 sec, 101 sec
- average CG iterations: 1.62, 1.39, 1.82
- iterations: 400, 400, 800
\( \ell_1 \)-regularized logistic regression

- fit logistic model, with \( \ell_1 \) regularization
- data \( z_i \in \mathbb{R}^p, i = 1, \ldots, q \) with labels \( y_i \in \{-1, 1\} \)
- solve

\[
\text{minimize} \quad \sum_{i=1}^{q} \log(1 + \exp(y_i w^T z_i)) + \mu \| w \|_1
\]

over variable \( w \in \mathbb{R}^p; \mu > 0 \) regularization parameter
- can be transformed to exponential cone program (ECP)

Numerical results
\( \ell_1 \)-regularized logistic regression results

|                  | small  | medium | large  |
|------------------|--------|--------|--------|
| features \( p \) | 600    | 2000   | 6000   |
| samples \( q \)  | 3000   | 10000  | 30000  |
| ECP variables \( n \) | 10200 | 34000  | 102000 |
| ECP constraints \( m \) | 22200 | 74000  | 222000 |
| nonzeros in \( A \) | \( 1.9 \times 10^5 \) | \( 1.9 \times 10^6 \) | \( 1.7 \times 10^7 \) |

**SCS direct:**

|                  | small  | medium | large  |
|------------------|--------|--------|--------|
| solve time       | 22.1 sec | 165 sec | 1020 sec |
| iterations       | 280    | 660    | 1240   |

**SCS indirect:**

|                  | small  | medium | large  |
|------------------|--------|--------|--------|
| solve time       | 24.0 sec | 199 sec | 1290 sec |
| average CG iterations | 2.00    | 2.49    | 2.82   |
| iterations       | 300    | 760    | 1320   |
Large random SOCP

- randomly generated SOCP with known optimal value
- \( n = 1.6 \times 10^6 \) variables, \( m = 4.8 \times 10^6 \) constraints
- \( 2 \times 10^9 \) nonzeros in \( A \), 22.5Gb memory to store
- indirect solver, tolerance \( 10^{-3} \), parallelized over 32 threads
- results:
  - 740 SCS iterations, about 5000 matrix multiplies
  - 10 hours wall-clock time
  - \(|c^T x - c^T x^*| / |c^T x^*| = 7 \times 10^{-4}\)
  - \(|b^T y - b^T y^*| / |b^T y^*| = 1 \times 10^{-3}\)
Outline

Cone programming

Homogeneous embedding

Operator splitting

Numerical results

Conclusions
Conclusions

- HSD embedding is great for first-order methods
- diagonal preconditioning critical
- matrix-free algorithm: only $z \rightarrow Az$, $w \rightarrow A^T w$
- SCS is now standard large scale solver in CVXPY