A conic operator splitting method for large convex problems

Michael Garstka · Paul Goulart · Mark Cannon

International Conference on Continuous Optimization, Berlin
6th August 2019
Why do we care about solving large convex conic problems?

- Problem dimensions grow drastically
- State-of-the-art (interior point) solver do not scale well
• ADMM solver for large convex conic problems
• Support of major convex cones:
  Zero cone       Second order cone       Power cone
  Nonnegatives    Positive semidefinite cone
  Hyperbox        Exponential cone
• Quadratic cost function and conic constraints
• Implemented in Julia
Overview

COSMO.jl

Example: Nearest Correlation Matrix problem

Conic Problem Format

ADMM Algorithm

Chordal decomposition of PSD constraints

Customisable and extensible code

Conclusion
Example: Nearest correlation matrix problem

- Given data matrix $C' \in \mathbb{R}^{n \times n}$ find the nearest correlation matrix $X$:
  
  \[
  \text{minimize} \quad \frac{1}{2} \|X - C\|_F^2 \\
  \text{subject to} \quad X_{ii} = 1, \quad i = 1, \ldots, n \\
  \quad X \in S_+^n,
  \]

- The objective function can be rewritten as
  
  \[
  \frac{1}{2} \|X - C\|_F^2 = \frac{1}{2} x^\top x - c^\top x + \frac{1}{2} c^\top c
  \]

  with $x = \text{vec}(X)$ and $c = \text{vec}(C')$
Example: Nearest correlation matrix problem

- We can solve this with a few lines of code with JuMP and COSMO:

```julia
C = Symmetric(rand(n, n));
c = vec(C);
m = JuMP.Model(with_optimizer(COSMO.Optimizer));
@variable(m, X[1:n, 1:n], PSD);
x = vec(X);
@objective(m, Min, 0.5 * x' * x - c' * x + 0.5 * c' * c) \frac{1}{2} \| X - C \|_F^2
@constraint(m, [i = 1:n], X[i, i] == 1.) \text{X}_{ii} = 1, i = 1, \ldots, n
JuMP.optimize!(m)
```
Example: Nearest correlation matrix problem

![Graph showing the solve time in seconds for different methods as a function of the dimension of the data matrix. The x-axis represents the dimension $n$ of the data matrix $C \in \mathbb{R}^{n \times n}$, and the y-axis represents the solve time in seconds. The methods compared are SeDuMi (MATLAB), Mosek (C), SCS (C), and COSMO (Julia).]
Problem Format

minimize \[ \frac{1}{2} x^T P x + q^T x \]
subject to \[ Ax + s = b \]
\[ s \in \mathcal{K} \]

- Decision variables: \( x \in \mathbb{R}^n, s \in \mathbb{R}^m \)
- Problem data: real matrices \( P \succeq 0, A \), and real vectors \( q, b \)
- Convex cone \( \mathcal{K} \) which can be a Cartesian product of cones:

\[ \mathcal{K} = \mathcal{K}_1 \times \mathcal{K}_2 \times \cdots \times \mathcal{K}_N \]
Problem Format

\[
\begin{align*}
\text{minimize} & \quad \frac{1}{2} x^\top P x + q^\top x \\
\text{subject to} & \quad A x + s = b \\
& \quad s \in \{0\}^{m_1} \times \mathbb{R}^{m_2} 
\end{align*}
\]

Linear Program

- Decision variables: \(x \in \mathbb{R}^n, s \in \mathbb{R}^m\)
- Problem data: real matrices \(P \succeq 0, A\), and real vectors \(q, b\)
- Convex cone \(\mathcal{K}\) which can be a Cartesian product of cones:

\[
\mathcal{K} = \mathcal{K}_1 \times \mathcal{K}_2 \times \cdots \times \mathcal{K}_N
\]
Problem Format

minimize \( \frac{1}{2} x^\top P x + q^\top x \)
subject to \( Ax + s = b \)
\( \text{mat}(s) \succeq 0 \)

Semidefinite Program

- Decision variables: \( x \in \mathbb{R}^n, s \in \mathbb{R}^m \)
- Problem data: real matrices \( P \succeq 0, A, \) and real vectors \( q, b \)
- Convex cone \( \mathcal{K} \) which can be a Cartesian product of cones:

\[
\mathcal{K} = \mathcal{K}_1 \times \mathcal{K}_2 \times \cdots \times \mathcal{K}_N
\]
Generic ADMM

minimize \quad f(x) + g(z)
subject to \quad Ax + Bz = c

- Augmented Lagrangian:

\[ L_\rho(x, z, y) = f(x) + g(z) + y^\top (Ax + Bz - c) + \frac{\rho}{2} \|Ax + Bz - c\|_2^2, \]

- ADMM steps:
Generic ADMM

minimize \[ f(x) + g(z) \]
subject to \[ Ax + Bz = c \]

• Augmented Lagrangian:

\[ L_\rho(x, z, y) = f(x) + g(z) + y^\top(Ax + Bz - c) + \frac{\rho}{2} \|Ax + Bz - c\|_2^2, \]

• ADMM steps:

\[ x^{k+1} := \arg\min_x L_\rho(x, z^k, y^k) \]
Generic ADMM

minimize \quad f(x) + g(z)

subject to \quad Ax + Bz = c

\begin{itemize}
  \item Augmented Lagrangian:
    \[ L_\rho(x, z, y) = f(x) + g(z) + y^\top (Ax + Bz - c) + \frac{\rho}{2} \| Ax + Bz - c \|^2_2, \]
  \item ADMM steps:
    \[ x^{k+1} := \text{argmin}_x L_\rho(x, z^k, y^k) \]
    \[ z^{k+1} := \text{argmin}_z L_\rho(x^{k+1}, z, y^k) \]
\end{itemize}
Generic ADMM

minimize \[ f(x) + g(z) \]
subject to \[ Ax + Bz = c \]

- Augmented Lagrangian:

\[ L_\rho(x, z, y) = f(x) + g(z) + y^\top(Ax + Bz - c) + \frac{\rho}{2}\|Ax + Bz - c\|_2^2, \]

- ADMM steps:

\[ x^{k+1} := \arg\min_x L_\rho(x, z^k, y^k) \]
\[ z^{k+1} := \arg\min_z L_\rho(x^{k+1}, z, y^k) \]
\[ y^{k+1} := y^k + \rho(Ax^{k+1} + Bz^{k+1} - c) \]
Splitting method

\[
\begin{align*}
\text{minimize} & \quad \frac{1}{2} x^\top P x + q^\top x \\
\text{subject to} & \quad A x + s = b \\
& \quad s \in \mathcal{K}
\end{align*}
\]

\[
\begin{align*}
\text{minimize} & \quad \frac{1}{2} \tilde{x}^\top P \tilde{x} + q^\top \tilde{x} + I_{Ax+s=b}(\tilde{x}, \tilde{s}) \\
& \quad + I_{\mathcal{K}}(s)
\end{align*}
\]

subject to \( (\tilde{x}, \tilde{s}) = (x, s) \)
ADMM algorithm

1: **Input:** Initial values $x^0, s^0, y^0$, step sizes $\sigma, \rho$

2: **Do**

3: 

   $$(\tilde{x}^{k+1}, \tilde{s}^{k+1}) = \arg\min_{\tilde{x}, \tilde{s}} L_\rho (\tilde{x}, \tilde{s}, x^k, s^k, y^k)$$

   equality constrained QP

   $\rightarrow$ linear KKT system

4: 

   $$x^{k+1} = \Pi_{\mathbb{R}^n} (\tilde{x}^{k+1})$$

   projection onto $\mathbb{R}^n$

5: 

   $$s^{k+1} = \Pi_K (\tilde{s}^{k+1} + \frac{1}{\rho} y^k)$$

   projection onto $K$

6: 

   $$y^{k+1} = y^k + \rho \left( \tilde{s}^{k+1} - s^{k+1} \right)$$

7: **while** termination criteria not satisfied
Chordal Decomposition

minimize \( \frac{1}{2} x^\top P x + q^\top x \)

subject to \( \sum_{i=1}^{m} A_i x_i + S = B \)
\( S \in \mathbb{S}_+^r \)

\[
\begin{bmatrix}
S_{11} & S_{12} & 0 & 0 & 0 & S_{16} \\
S_{21} & S_{22} & S_{23} & S_{24} & 0 & S_{26} \\
0 & S_{32} & S_{33} & S_{34} & 0 & 0 \\
0 & S_{42} & S_{43} & S_{44} & S_{45} & S_{46} \\
0 & 0 & 0 & S_{54} & S_{55} & S_{56} \\
S_{61} & S_{62} & 0 & S_{64} & S_{65} & S_{66}
\end{bmatrix}
\]
Chordal Decomposition

minimize \[ \frac{1}{2} x^\top P x + q^\top x \]

subject to \[ \sum_{i=1}^{m} A_i x_i + S = B \]

\[ S \in \mathbb{S}_+^r \]
Chordal Decomposition

minimize \( \frac{1}{2} x^\top P x + q^\top x \)

subject to \( \sum_{i=1}^{m} A_i x_i + S = B \)

\( S \in \mathbb{S}^r_+ \)

\[
\begin{bmatrix}
S_{11} & S_{12} & 0 & 0 & 0 & S_{16} \\
S_{21} & S_{22} & S_{23} & S_{24} & 0 & S_{26} \\
0 & S_{32} & S_{33} & S_{34} & 0 & 0 \\
0 & S_{42} & S_{43} & S_{44} & S_{45} & S_{46} \\
0 & 0 & 0 & S_{54} & S_{55} & S_{56} \\
S_{61} & S_{62} & 0 & S_{64} & S_{65} & S_{66}
\end{bmatrix}
\]
Chordal Decomposition

minimize \( \frac{1}{2} x^\top P x + q^\top x \)

subject to \( \sum_{i=1}^{m} A_i x_i + S = B \)

\( S \in \mathbb{S}_+^r \)
Chordal Decomposition

minimize $\frac{1}{2}x^\top Px + q^\top x$

subject to $\sum_{i=1}^{m} A_i x_i + S = B$

$S \in \mathbb{S}_+^r$
Chordal Decomposition

\[
\begin{align*}
\text{minimize} & \quad \frac{1}{2} x^T P x + q^T x \\
\text{subject to} & \quad \sum_{i=1}^{m} A_i x_i + S = B \\
& \quad S \in \mathbb{S}_+^r
\end{align*}
\]

\[
\begin{align*}
\text{minimize} & \quad \frac{1}{2} x^T P x + q^T x \\
\text{subject to} & \quad \sum_{i=1}^{m} A_i x_i + \sum_{\ell=1}^{p} T_{\ell}^T S_{\ell} T_{\ell} = B \\
& \quad S_{\ell} \in \mathbb{S}_{+}^{|C_{\ell}|}, \quad \ell = 1, \ldots, p
\end{align*}
\]

Agler’s theorem
Example: Nearest correlation matrix problem

- Given data matrix $C \in \mathbb{R}^{n \times n}$ find the nearest correlation matrix $X$:

$$\begin{align*}
\text{minimize} & \quad \frac{1}{2} \| X - C \|_F^2 \\
\text{subject to} & \quad X_{ii} = 1, \quad i = 1, \ldots, n \\
& \quad X \in S_n^+, 
\end{align*}$$

- Let’s assume that $C$ has a chordal sparsity structure with $G(V, E)$:

- We want $X$ to keep the same sparsity structure $X \in S_n^+(E, 0)$
Example: Nearest correlation matrix problem

```julia
m = JuMP.Model(with_optimizer(COSMO.Optimizer, decompose = true));
@variable(m, X[1:n, 1:n]);
x = vec(X);
@objective(m, Min, 0.5 .* x' * x - c' * x + 0.5 * c' * c)
@constraint(m, [i = 1: n], X[i, i] == 1.)

@constraint(m, A * x in MOI.PositiveSemidefiniteConeTriangle(n));
JuMP.optimize!(m)
```
Example: Nearest correlation matrix problem
ADMM algorithm

1: Input: Initial values $x^0$, $s^0$, $y^0$, step sizes $\sigma$, $\rho$
2: Do
3: $(\tilde{x}^{k+1}, \tilde{s}^{k+1}) = \operatorname{argmin}_{x, s} L_{\rho} (\tilde{x}, \tilde{s}, x^k, s^k, y^k)$
   equality constrained QP
   $\rightarrow$ linear KKT system
4: $x^{k+1} = \Pi_{\mathbb{R}^n} (\tilde{x}^{k+1})$
5: $s^{k+1} = \Pi_{K} (\tilde{s}^{k+1} + \frac{1}{\rho} y^k)$
   projection onto $K$
6: $y^{k+1} = y^k + \rho \left( \tilde{s}^{k+1} - s^{k+1} \right)$
7: while termination criteria not satisfied
Customisable and extensible code

- Custom solver for KKT system
- User-defined convex sets:

```julia
# Define new convex set
struct MyConvexSet <: COSMO.AbstractConvexSet
    dim::Int
end

# define a projection function
function COSMO.project!(x, convex_set::MyConvexSet)
    # projection code for x onto convex_set
end
```
Conclusion:

- open source ADMM-based solver written in Julia
- supports quadratic objectives
- supports major convex cones
- infeasibility detection
- chordal decomposition of PSD constraints
- allows user-defined convex sets
**Conclusion:**

- open source ADMM-based solver written in Julia
- supports quadratic objectives
- supports major convex cones
- infeasibility detection
- chordal decomposition of PSD constraints
- allows user-defined convex sets

**Future work:**

- Acceleration methods
- Approximate projections
- Parallel Implementation of projections
Future Work: Smart Clique Merging

\[
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{bmatrix}
\]
Future Work: Smart Clique Merging
Future Work: Smart Clique Merging
Future Work: Smart Clique Merging
Future Work: Smart Clique Merging
COSMO.jl Package

- Installation via the Julia package manager

- Code and documentation available at:
  https://github.com/oxfordcontrol/COSMO.jl