Multi-Block Alternating Direction Method of Multipliers

Are there Alternative Algorithms for Linear Programming?

Yinyu Ye

[1]Department of Management Science and Engineering and
Institute for Computational and Mathematical Engineering
Stanford University, Stanford

September 17, 2015

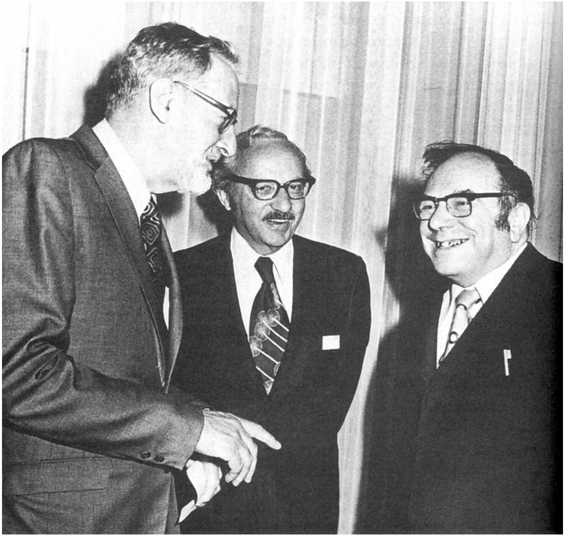
Table of Contents

1. Linear Programming and Algorithms
2. Advances in Simplex and Interior-Point Algorithms
3. First-Order and Alternating Direction Method of Multipliers
4. Convergence of RP-ADMM in Expectation
5. Why Multi-Block ADMM?

Table of Contents

1. Linear Programming and Algorithms
2. Advances in Simplex and Interior-Point Algorithms
3. First-Order and Alternating Direction Method of Multipliers
4. Convergence of RP-ADMM in Expectation
5. Why Multi-Block ADMM?

Linear Programming (LP)



Algebra of Linear Programming

$$\begin{array}{lll} \text{minimize}_{\mathbf{x}} & \mathbf{c}^T\mathbf{x} & \\ \\ \text{subject to} & A\mathbf{x} & (<=>)\mathbf{b}, \\ & \mathbf{x} & \geq \mathbf{0}, \end{array}$$

where given constraint matrix A is an $m \times n$ matrix, the right-hand $\mathbf{b}$ is an m-dimensional vector, the objective coefficients $\mathbf{c}$ is an n-dimensional vector.

Algebra of Linear Programming

$$
\begin{array}{lll}
\text{minimize}_{\mathbf{x}} & \mathbf{c}^T\mathbf{x} & \\
\text{subject to} & A\mathbf{x} & (<=>)\mathbf{b}, \\
& \mathbf{x} & \geq \mathbf{0},
\end{array}
$$

where given constraint matrix A is an $m \times n$ matrix, the right-hand $\mathbf{b}$ is an m-dimensional vector, the objective coefficients $\mathbf{c}$ is an n-dimensional vector.

Variables $\mathbf{x}$ is an n-dimensional vector and need to be optimally decided.

Algebra of Linear Programming

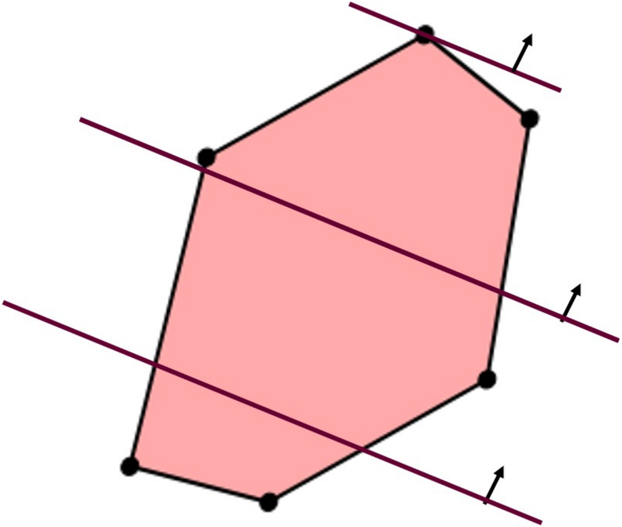
$$\begin{array}{lll} \text{minimize}_{\mathbf{x}} & \mathbf{c}^T\mathbf{x} & \\ \text{subject to} & A\mathbf{x} & (<=>)\mathbf{b}, \\ & \mathbf{x} & \geq \mathbf{0}, \end{array}$$

where given constraint matrix A is an $m \times n$ matrix, the right-hand $\mathbf{b}$ is an m-dimensional vector, the objective coefficients $\mathbf{c}$ is an n-dimensional vector.

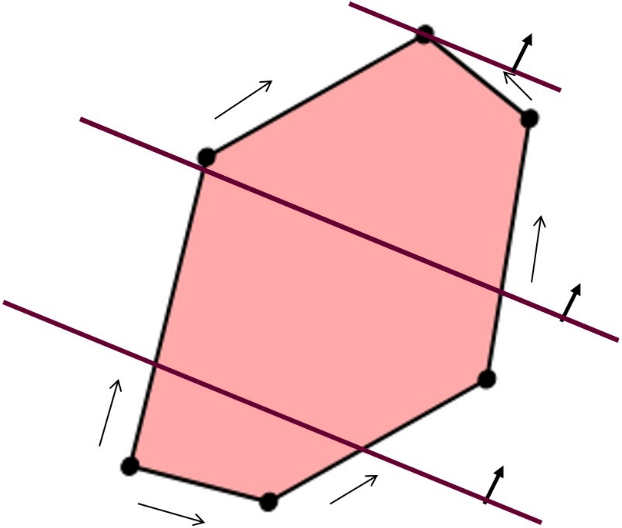
Variables $\mathbf{x}$ is an n-dimensional vector and need to be optimally decided.

LP is a data-driven computation/decision model that has wide applications so that we like to decide the optimal $\mathbf{x}$ fast in theory and practice.

Geometry of Linear Programming



LP Algorithms: the Simplex Method



LP Algorithms: the Interior-Point Method

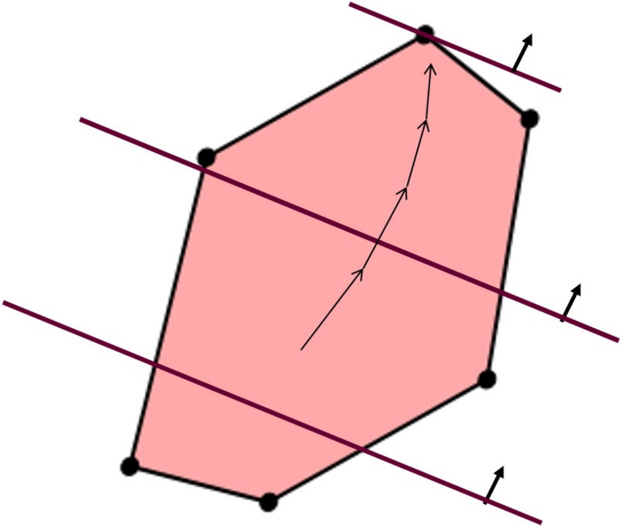


Table of Contents

1. Linear Programming and Algorithms
2. Advances in Simplex and Interior-Point Algorithms
3. First-Order and Alternating Direction Method of Multipliers
4. Convergence of RP-ADMM in Expectation
5. Why Multi-Block ADMM?

Advances in the Simplex Method

- Markov decision processes (MDPs) provide a mathematical framework for sequential decision-making where outcomes are partly random and partly under the control of a decision maker.

Advances in the Simplex Method

- Markov decision processes (MDPs) provide a mathematical framework for sequential decision-making where outcomes are partly random and partly under the control of a decision maker.
- MDPs are useful for studying a wide range of optimization problems solved via dynamic programming, where it was known at least as early as the 1950s (cf. Shapley 53, Bellman 57).

Advances in the Simplex Method

- Markov decision processes (MDPs) provide a mathematical framework for sequential decision-making where outcomes are partly random and partly under the control of a decision maker.
- MDPs are useful for studying a wide range of optimization problems solved via dynamic programming, where it was known at least as early as the 1950s (cf. Shapley 53, Bellman 57).
- Modern applications include dynamic planning, reinforcement learning, social networking, and almost all other dynamic/sequential decision making problems in Mathematical, Physical, Management and Social Sciences.

The Markov Decision Process/Game continued

- At each time step, the process is in some state $i \in \{1, ..., m\}$, and the decision maker chooses an action j_i among a set of actions, $\mathcal{A}_i$, for state i.

The Markov Decision Process/Game continued

- At each time step, the process is in some state $i \in \{1, ..., m\}$, and the decision maker chooses an action j_i among a set of actions, $\mathcal{A}_i$, for state i.
- The process responds at the next time step by randomly moving into a state and producing a corresponding cost c_{j_i}. The corresponding probabilities entering next state, $\mathbf{p}_{j_i}$, is conditionally independent of all previous states and actions.

The Markov Decision Process/Game continued

- At each time step, the process is in some state $i \in \{1, ..., m\}$, and the decision maker chooses an action j_i among a set of actions, $\mathcal{A}_i$, for state i.
- The process responds at the next time step by randomly moving into a state and producing a corresponding cost c_{j_i}. The corresponding probabilities entering next state, $\mathbf{p}_{j_i}$, is conditionally independent of all previous states and actions.
- A stationary policy for the decision maker is a set of m actions taking by the decision maker all times.

The Markov Decision Process/Game continued

- At each time step, the process is in some state $i \in \{1, ..., m\}$, and the decision maker chooses an action j_i among a set of actions, $\mathcal{A}_i$, for state i.
- The process responds at the next time step by randomly moving into a state and producing a corresponding cost c_{j_i}. The corresponding probabilities entering next state, $\mathbf{p}_{j_i}$, is conditionally independent of all previous states and actions.
- A stationary policy for the decision maker is a set of m actions taking by the decision maker all times.
- The MDP is to find a policy to optimiza the expected discounted sum over infinite horizon with a discount factor $0 \leq \gamma < 1$.

The Fixed-Point of Cost-to-Go Values

An optimal policy is associated with m cost-to-go values for every state, $\mathbf{y} \in R^m$, such that it is a fixed point:

$$y_i^* = \min\{c_{j_i} + \gamma \mathbf{p}_{j_i}^T \mathbf{y},\ j_i \in \mathcal{A}_i\},\ \forall i,$$

where the optimal action

$$j_i^* = \arg\min\{c_{j_i} + \gamma \mathbf{p}_{j_i}^T \mathbf{y},\ j_i \in \mathcal{A}_i\},\ \forall i.$$

The Fixed-Point of Cost-to-Go Values

An optimal policy is associated with m cost-to-go values for every state, $\mathbf{y} \in R^m$, such that it is a fixed point:

$$y_i^* = \min\{c_{j_i} + \gamma \mathbf{p}_{j_i}^T \mathbf{y},\ j_i \in \mathcal{A}_i\},\ \forall i,$$

where the optimal action

$$j_i^* = \arg\min\{c_{j_i} + \gamma \mathbf{p}_{j_i}^T \mathbf{y},\ j_i \in \mathcal{A}_i\},\ \forall i.$$

$$\begin{array}{lccl}
\text{maximize}_{\mathbf{y}} & \sum_{i=1}^m y_i & & \\
\text{subject to} & y_1 & \leq & \mathbf{c}_{j_1} + \gamma \mathbf{p}_{j_1}^T \mathbf{y},\ j_1 \in \mathcal{A}_1 \\
 & \ldots & \ldots & \ldots \\
 & y_i & \leq & \mathbf{c}_{j_i} + \gamma \mathbf{p}_{j_i}^T \mathbf{y},\ j_i \in \mathcal{A}_i \\
 & \ldots & \ldots & \ldots \\
 & y_m & \leq & \mathbf{c}_{j_m} + \gamma \mathbf{p}_{j_m}^T \mathbf{y},\ j_m \in \mathcal{A}_m.
\end{array}$$

Algorithmic Events of the MDP Methods

- Shapley 53 and Bellman 57 developed a method called the value-iteration method to approximate the optimal state values.

Algorithmic Events of the MDP Methods

- Shapley 53 and Bellman 57 developed a method called the value-iteration method to approximate the optimal state values.
- Another best known method is due to Howard 60 and is known as the policy-iteration method, or block pivoting simplex method, which generate an optimal policy in finite number of iterations in a distributed and decentralized way.

Algorithmic Events of the MDP Methods

- Shapley 53 and Bellman 57 developed a method called the value-iteration method to approximate the optimal state values.
- Another best known method is due to Howard 60 and is known as the policy-iteration method, or block pivoting simplex method, which generate an optimal policy in finite number of iterations in a distributed and decentralized way.
- de Ghellinck 60, D'Epenoux 60 and Manne 60 showed that the MDP has an LP representation, so that it can be solved by the simplex method of Dantzig 47, which is a special policy-iteration method that each step only actions in one state is switched.

Algorithmic Events of the MDP Methods

- Shapley 53 and Bellman 57 developed a method called the value-iteration method to approximate the optimal state values.
- Another best known method is due to Howard 60 and is known as the policy-iteration method, or block pivoting simplex method, which generate an optimal policy in finite number of iterations in a distributed and decentralized way.
- de Ghellinck 60, D'Epenoux 60 and Manne 60 showed that the MDP has an LP representation, so that it can be solved by the simplex method of Dantzig 47, which is a special policy-iteration method that each step only actions in one state is switched.
- But most analyses of these methods are negative: the simplex method with smallest index rule (Melekopoglou/Condon 90), random pivoting rule (Friedman et al 12), and policy-iteration for undiscounted finite-horizon MDP (Fearnley 10) are all exponential.

Positive Results for MDP of m states and n actions

- The classic simplex method, with the Dantzig pivoting rule, terminates in
$$\frac{m(n-m)}{1-\gamma} \cdot \log\left(\frac{m^2}{1-\gamma}\right)$$
iterations, and each iteration uses at most $O(mn)$ arithmetic operations (Y MOR10).

Positive Results for MDP of m states and n actions

- The classic simplex method, with the Dantzig pivoting rule, terminates in
$$\frac{m(n-m)}{1-\gamma} \cdot \log\left(\frac{m^2}{1-\gamma}\right)$$
iterations, and each iteration uses at most $O(mn)$ arithmetic operations (Y MOR10).
- The policy-iteration or block-pivoting method
$$\frac{n}{1-\gamma} \cdot \log\left(\frac{m}{1-\gamma}\right),$$
iterations and each iteration uses at most m^2n arithmetic operations (Hansen/Miltersen/Zwick ACM12).

Positive Results for MDP continued

- The simplex method for deterministic MDP, regardless discount factor, terminates in

$$O(m^3 n^2 \log^2 m)$$

iterations (Post/Y MOR2014).

Positive Results for MDP continued

- The simplex method for deterministic MDP, regardless discount factor, terminates in

$$O(m^3 n^2 \log^2 m)$$

iterations (Post/Y MOR2014).

- The result was extended to general LP by Kitahara and Mizuno 2012; also see renewed exciting research work on the simplex method, e.g., Feinberg/Huang 2013, Lee/Epelman/Romeijn/Smith 2013, Scherrer 2014, Fearnley/Savani 2014, Adler/Papadimitriou/Rubinstein 2014, etc.

The Turn-Based Two-Person Zero-Sum Game

- The states is partitioned to two sets where one is to max and the other is to min, i.e., the fixed point:

$$
\begin{aligned}
y_i^* &= \min\{\mathbf{c}_{j_i} + \gamma \mathbf{p}_{j_i}^T \mathbf{y},\ j_i \in \mathcal{A}_i\},\ \forall i \in I^+,\\
y_i^* &= \max\{\mathbf{c}_{j_i} + \gamma \mathbf{p}_{j_i}^T \mathbf{y},\ j_i \in \mathcal{A}_i\},\ \forall i \in I^-.
\end{aligned}
$$

The Turn-Based Two-Person Zero-Sum Game

- The states is partitioned to two sets where one is to max and the other is to min, i.e., the fixed point:

$$
\begin{aligned}
y_i^* &= \min\{\mathbf{c}_{j_i} + \gamma \mathbf{p}_{j_i}^T \mathbf{y},\ j_i \in \mathcal{A}_i\},\ \forall i \in I^+,\\
y_i^* &= \max\{\mathbf{c}_{j_i} + \gamma \mathbf{p}_{j_i}^T \mathbf{y},\ j_i \in \mathcal{A}_i\},\ \forall i \in I^-.
\end{aligned}
$$

- It does not admit a convex programming formulation, and it is unknown if it can be solved in polynomial time in general.

The Turn-Based Two-Person Zero-Sum Game

- The states is partitioned to two sets where one is to max and the other is to min, i.e., the fixed point:

$$
\begin{aligned}
y_i^* &= \min\{\mathbf{c}_{j_i} + \gamma \mathbf{p}_{j_i}^T \mathbf{y},\ j_i \in \mathcal{A}_i\},\ \forall i \in I^+, \\
y_i^* &= \max\{\mathbf{c}_{j_i} + \gamma \mathbf{p}_{j_i}^T \mathbf{y},\ j_i \in \mathcal{A}_i\},\ \forall i \in I^-.
\end{aligned}
$$

- It does not admit a convex programming formulation, and it is unknown if it can be solved in polynomial time in general.
- Hansen/Miltersen/Zwick ACM12 proved that the strategy iteration method solves it in strongly polynomial time when discount factor is fixed – the first strongly polynomial time algorithm.

Advances in Interior-Point Methods

In theory, the best interior-point algorithms converge in $\tilde{O}(\sqrt{\max\{m,n\}})$ iterations, where notation $\tilde{O}$ includes some constant and logarithmic factors of dimensions and accuracy ϵ.

Advances in Interior-Point Methods

In theory, the best interior-point algorithms converge in $\tilde{O}(\sqrt{\max\{m, n\}})$ iterations, where notation $\tilde{O}$ includes some constant and logarithmic factors of dimensions and accuracy ϵ.

- Navigating Central Path with Electrical Flows: from Flows to Matchings, and Back (Madry FOCS13). The paper made a breakthrough on solving a class of max-flow and min-cut problems.

Advances in Interior-Point Methods

In theory, the best interior-point algorithms converge in $\tilde{O}(\sqrt{\max\{m, n\}})$ iterations, where notation $\tilde{O}$ includes some constant and logarithmic factors of dimensions and accuracy ϵ.

- Navigating Central Path with Electrical Flows: from Flows to Matchings, and Back (Madry FOCS13). The paper made a breakthrough on solving a class of max-flow and min-cut problems.
- Path-Finding Methods for Linear Programming : Solving Linear Programs in $\tilde{O}(\sqrt{\min\{m, n\}})$ Iterations and Faster Algorithms for Maximum Flow (Lee and Sidford, FOCS14).

Advances in Interior-Point Methods

In theory, the best interior-point algorithms converge in $\tilde{O}(\sqrt{\max\{m,n\}})$ iterations, where notation $\tilde{O}$ includes some constant and logarithmic factors of dimensions and accuracy ϵ.

- Navigating Central Path with Electrical Flows: from Flows to Matchings, and Back (Madry FOCS13). The paper made a breakthrough on solving a class of max-flow and min-cut problems.
- Path-Finding Methods for Linear Programming : Solving Linear Programs in $\tilde{O}(\sqrt{\min\{m,n\}})$ Iterations and Faster Algorithms for Maximum Flow (Lee and Sidford, FOCS14).
- Efficient Inverse Maintenance and Faster Algorithms for Linear Programming (Lee and Sidford, FOCS15), where the author also have also reduced the operation complexity for linear programming.

Table of Contents

1 Linear Programming and Algorithms

2 Advances in Simplex and Interior-Point Algorithms

3 First-Order and Alternating Direction Method of Multipliers

4 Convergence of RP-ADMM in Expectation

5 Why Multi-Block ADMM?

First-Order or "Inverse-Free" Methods

- Early work include the von Neumann projection method (also see Fruend and Jarre/Rendl 07).

First-Order or "Inverse-Free" Methods

- Early work include the von Neumann projection method (also see Fruend and Jarre/Rendl 07).
- Use subgradient method for a transformed problem, assuming knowing a strict feasible point (Reneger 14, Freund 15) with iteration complexity $O(L^2 D^2 \frac{1}{\epsilon^2})$, where L and D are condition numbers on the optimal solution structure and feasible region.

First-Order or "Inverse-Free" Methods

- Early work include the von Neumann projection method (also see Fruend and Jarre/Rendl 07).
- Use subgradient method for a transformed problem, assuming knowing a strict feasible point (Reneger 14, Freund 15) with iteration complexity $O(L^2D^2\frac{1}{\epsilon^2})$, where L and D are condition numbers on the optimal solution structure and feasible region.
- First-order Karmarkar potential reduction reduction algorithm...

First-Order or "Inverse-Free" Methods

- First order method for packing and covering LP (Allen-Zhu/Orecchia 14): penalize the constraint, then use stochastic coordinate descent and several techniques. The iteration complexity: $\tilde{O}(\frac{1}{\epsilon})$ for packing LP and $\tilde{O}(\frac{1}{\epsilon^{1.5}})$ for covering LP.

First-Order or "Inverse-Free" Methods

- First order method for packing and covering LP (Allen-Zhu/Orecchia 14): penalize the constraint, then use stochastic coordinate descent and several techniques. The iteration complexity: $\tilde{O}(\frac{1}{\epsilon})$ for packing LP and $\tilde{O}(\frac{1}{\epsilon^{1.5}})$ for covering LP.
- Two-Block ADMM for solve primal-LP (He/Yuan 11, Monteiro/Svaiter 13, Monteiro/Ortiz/Svaiter 14) with iteration complexity $\tilde{O}(\frac{1}{\epsilon})$ (a matrix needs to be inversed once).

First-Order Potential Reduction I

$$\begin{array}{ll} \text{Minimize} & f(\mathbf{x}) \\ \text{Subject to} & \mathbf{e}^T\mathbf{x} = 1;\ \mathbf{x} \geq \mathbf{0}, \end{array}$$

where $\mathbf{e}$ is the vector of all ones.

First-Order Potential Reduction I

$$\begin{array}{ll} \text{Minimize} & f(\mathbf{x}) \\ \text{Subject to} & \mathbf{e}^T\mathbf{x} = 1;\ \mathbf{x} \geq \mathbf{0}, \end{array}$$

where $\mathbf{e}$ is the vector of all ones.

Assume that $f(\mathbf{x})$ is a convex function in $\mathbf{x} \in R^n$ and $f(\mathbf{x}^*) = 0$ where $\mathbf{x}^*$ is a minimizer of the problem. Furthermore,

$$f(\mathbf{x}+\mathbf{d}) - f(\mathbf{x}) \leq \nabla f(\mathbf{x})^T\mathbf{d} + \frac{\gamma}{2}\|\mathbf{d}\|^2,$$

where positive γ is the Lipschitz parameter.

First-Order Potential Reduction I

$$\begin{array}{ll} \text{Minimize} & f(\mathbf{x}) \\ \text{Subject to} & \mathbf{e}^T\mathbf{x} = 1;\ \mathbf{x} \ge \mathbf{0}, \end{array}$$

where $\mathbf{e}$ is the vector of all ones.

Assume that $f(\mathbf{x})$ is a convex function in $\mathbf{x} \in R^n$ and $f(\mathbf{x}^*) = 0$ where $\mathbf{x}^*$ is a minimizer of the problem. Furthermore,

$$f(\mathbf{x}+\mathbf{d}) - f(\mathbf{x}) \le \nabla f(\mathbf{x})^T\mathbf{d} + \frac{\gamma}{2}\|\mathbf{d}\|^2,$$

where positive γ is the Lipschitz parameter.

We consider the potential function (e.g., Karmarkar 84,Todd/Ye 87)

$$\phi(\mathbf{x}) = \rho \ln(f(\mathbf{x})) - \sum_j \ln(x_j),$$

where $\rho \ge n$ over the simplex. If we start from $\mathbf{x}^0 = \frac{1}{n}\mathbf{e}$, and generate a sequence of points $\mathbf{x}^k$, $k = 1, ...,$, whose potential value is strictly decrease by a fixed amount.

First-Order Potential Reduction II

Update $\mathbf{x}$ by solving

$$\begin{array}{ll} \text{Minimize} & \nabla\phi(\mathbf{x})^T\mathbf{d} \\ \text{Subject to} & \mathbf{e}^T\mathbf{d} = 0,\ \|X^{-1}\mathbf{d}\| \leq \beta; \end{array}$$

where $\beta < 1$ is yet to be determined; and $\mathbf{x}^+ = \mathbf{x} + \mathbf{d}$.

First-Order Potential Reduction II

Update $\mathbf{x}$ by solving

$$\begin{array}{ll} \text{Minimize} & \nabla\phi(\mathbf{x})^T\mathbf{d} \\ \text{Subject to} & \mathbf{e}^T\mathbf{d} = 0, \ \|X^{-1}\mathbf{d}\| \le \beta; \end{array}$$

where $\beta < 1$ is yet to be determined; and $\mathbf{x}^+ = \mathbf{x} + \mathbf{d}$.

$$\phi(\mathbf{x}^+) - \phi(\mathbf{x}) \le -\beta + \frac{\rho\gamma}{2f(\mathbf{x})}\beta^2 + \frac{\beta^2}{2(1-\beta)}$$

so that one can choose a β to make

$$\phi(\mathbf{x}^+) - \phi(\mathbf{x}) \le \frac{-f(\mathbf{x})}{2(f(\mathbf{x}) + 2\rho\gamma)}.$$

First-Order Potential Reduction II

Update $\mathbf{x}$ by solving

$$\begin{array}{ll} \text{Minimize} & \nabla\phi(\mathbf{x})^T\mathbf{d} \\ \text{Subject to} & \mathbf{e}^T\mathbf{d}=0,\ \|X^{-1}\mathbf{d}\|\le\beta; \end{array}$$

where $\beta<1$ is yet to be determined; and $\mathbf{x}^+=\mathbf{x}+\mathbf{d}$.

$$\phi(\mathbf{x}^+)-\phi(\mathbf{x})\le-\beta+\frac{\rho\gamma}{2f(\mathbf{x})}\beta^2+\frac{\beta^2}{2(1-\beta)}$$

so that one can choose a β to make

$$\phi(\mathbf{x}^+)-\phi(\mathbf{x})\le\frac{-f(\mathbf{x})}{2(f(\mathbf{x})+2\rho\gamma)}.$$

Theorem

The steepest descent potential reduction algorithm generates a $\mathbf{x}^k$ with $f(\mathbf{x}^k)/f(\mathbf{x}^0)\le\epsilon$ in no more than $4(n+\sqrt{n})\frac{\max\{1,2(n+\sqrt{n})\gamma/f(x^0)\}}{\epsilon}\ln(\frac{1}{\epsilon})$ steps.

Alternating Direction Method of Multipliers

- Consider the convex optimization problem

$$\begin{aligned}
\min_{\mathbf{x}\in R^n} \quad & f_1(\mathbf{x}_1) + \cdots + f_p(\mathbf{x}_p), \\
s.t. \quad & A\mathbf{x} \triangleq A_1\mathbf{x}_1 + \cdots + A_p\mathbf{x}_p = \mathbf{b}, \\
& \mathbf{x}_i \in \mathcal{X}_i \subset R^{d_i},\ \ i = 1, \ldots, p.
\end{aligned}$$

Alternating Direction Method of Multipliers

- Consider the convex optimization problem

$$\begin{aligned} \min_{\mathbf{x}\in R^n} \quad & f_1(\mathbf{x}_1) + \cdots + f_p(\mathbf{x}_p), \\ s.t. \quad & A\mathbf{x} \triangleq A_1\mathbf{x}_1 + \cdots + A_p\mathbf{x}_p = \mathbf{b}, \\ & \mathbf{x}_i \in \mathcal{X}_i \subset R^{d_i}, \ \ i = 1, \ldots, p. \end{aligned}$$

- Augmented Lagrangian function:

$$\begin{aligned} L_\gamma(\mathbf{x}_1, .., \mathbf{x}_p; \mathbf{y}) = \ & \textstyle\sum_i f_i(\mathbf{x}_i) - \mathbf{y}^T(\sum_i A_i\mathbf{x}_i - \mathbf{b}) \\ & + \tfrac{\gamma}{2}\|\textstyle\sum_i A_i\mathbf{x}_i - \mathbf{b}\|^2. \end{aligned}$$

Alternating Direction Method of Multipliers continued

- Multi-block ADMM:

$$\begin{cases} \mathbf{x}_1^{k+1} \longleftarrow \arg\min_{\mathbf{x}_1 \in \mathcal{X}_1} L_\gamma(\mathbf{x}_1, \mathbf{x}_2^k \ldots, \mathbf{x}_p^k; \mathbf{y}^k), \\ \qquad\qquad \vdots \\ \mathbf{x}_p^{k+1} \longleftarrow \arg\min_{\mathbf{x}_p \in \mathcal{X}_p} L_\gamma(\mathbf{x}_1^{k+1}, \ldots, \mathbf{x}_{p-1}^{k+1}, \mathbf{x}_p; \mathbf{y}^k), \\ \mathbf{y}^{k+1} \longleftarrow \mathbf{y}^k - \gamma(\sum_i A_i \mathbf{x}_i^{k+1} - \mathbf{b}). \end{cases}$$

Alternating Direction Method of Multipliers continued

- Multi-block ADMM:

$$\begin{cases} \mathbf{x}_1^{k+1} \longleftarrow \arg\min_{\mathbf{x}_1 \in \mathcal{X}_1} L_\gamma(\mathbf{x}_1, \mathbf{x}_2^k \ldots, \mathbf{x}_p^k; \mathbf{y}^k), \\ \qquad\qquad \vdots \\ \mathbf{x}_p^{k+1} \longleftarrow \arg\min_{\mathbf{x}_p \in \mathcal{X}_p} L_\gamma(\mathbf{x}_1^{k+1}, \ldots, \mathbf{x}_{p-1}^{k+1}, \mathbf{x}_p; \mathbf{y}^k), \\ \mathbf{y}^{k+1} \longleftarrow \mathbf{y}^k - \gamma(\sum_i A_i \mathbf{x}_i^{k+1} - \mathbf{b}). \end{cases}$$

- Convergence was well established when $p = 1$ or $p = 2$ (..., Glowinski/Marrocco 75, Gabay/Mercier '76, Eckstein/Bertsekas 92,...)

Alternating Direction Method of Multipliers continued

- Multi-block ADMM:

$$\begin{cases} \mathbf{x}_1^{k+1} \longleftarrow \arg\min_{\mathbf{x}_1 \in \mathcal{X}_1} L_\gamma(\mathbf{x}_1, \mathbf{x}_2^k \ldots, \mathbf{x}_p^k; \mathbf{y}^k), \\ \qquad\qquad \vdots \\ \mathbf{x}_p^{k+1} \longleftarrow \arg\min_{\mathbf{x}_p \in \mathcal{X}_p} L_\gamma(\mathbf{x}_1^{k+1}, \ldots, \mathbf{x}_{p-1}^{k+1}, \mathbf{x}_p; \mathbf{y}^k), \\ \mathbf{y}^{k+1} \longleftarrow \mathbf{y}^k - \gamma(\sum_i A_i \mathbf{x}_i^{k+1} - \mathbf{b}). \end{cases}$$

- Convergence was well established when $p = 1$ or $p = 2$ (..., Glowinski/Marrocco 75, Gabay/Mercier '76, Eckstein/Bertsekas 92,...)
- But what about $p > 2$?

Convergence of Multi-Block ADMM

- Recent Discovery: When $p \geq 3$, ADMM can diverge [Chen/He/Y/Yuan MP14]:

Convergence of Multi-Block ADMM

- Recent Discovery: When $p \geq 3$, ADMM can diverge [Chen/He/Y/Yuan MP14]: $p = 3$.

$$A\mathbf{x} = \mathbf{a}_1 x_1 + \mathbf{a}_2 x_2 + \mathbf{a}_3 x_3 = \mathbf{0}, \text{ where } A = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 2 \\ 1 & 2 & 2 \end{bmatrix}.$$

Convergence of Multi-Block ADMM

- Recent Discovery: When $p \geq 3$, ADMM can diverge [Chen/He/Y/Yuan MP14]: $p = 3$.

$$A\mathbf{x} = \mathbf{a}_1 x_1 + \mathbf{a}_2 x_2 + \mathbf{a}_3 x_3 = \mathbf{0}, \text{ where } A = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 2 \\ 1 & 2 & 2 \end{bmatrix}.$$

- The ADMM would be a linear mapping of matrix M

$$\mathbf{z}^{k+1} = M\mathbf{z}^k, \text{ where } \mathbf{z}^k = (x_1^k; x_2^k; x_3^k; \mathbf{y}^k)$$

Convergence of Multi-Block ADMM

- Recent Discovery: When $p \geq 3$, ADMM can diverge [Chen/He/Y/Yuan MP14]: $p = 3$.

$$A\mathbf{x} = \mathbf{a}_1 x_1 + \mathbf{a}_2 x_2 + \mathbf{a}_3 x_3 = \mathbf{0}, \text{ where } A = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 2 \\ 1 & 2 & 2 \end{bmatrix}.$$

- The ADMM would be a linear mapping of matrix M

$$\mathbf{z}^{k+1} = M\mathbf{z}^k, \text{ where } \mathbf{z}^k = (x_1^k; x_2^k; x_3^k; \mathbf{y}^k)$$

- Why diverges? $\rho(M) > 1$ for above A and any choice of γ.

Convergence of Multi-Block ADMM

- Recent Discovery: When $p \geq 3$, ADMM can diverge [Chen/He/Y/Yuan MP14]: $p = 3$.

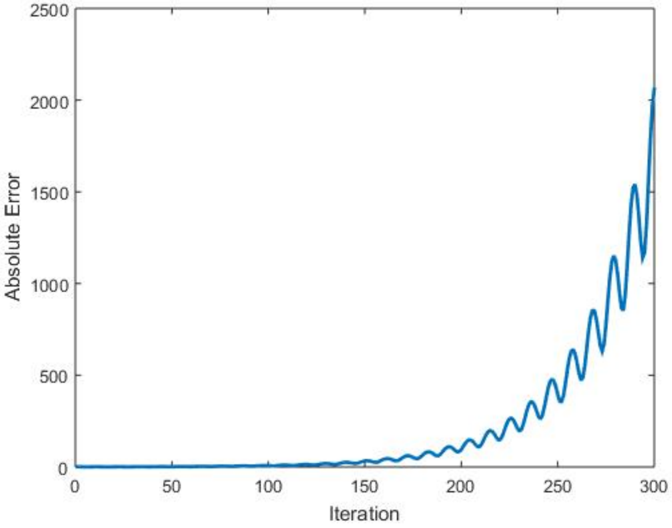
$$A\mathbf{x} = \mathbf{a}_1 x_1 + \mathbf{a}_2 x_2 + \mathbf{a}_3 x_3 = \mathbf{0}, \text{ where } A = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 2 \\ 1 & 2 & 2 \end{bmatrix}.$$

- The ADMM would be a linear mapping of matrix M

$$\mathbf{z}^{k+1} = M\mathbf{z}^k, \text{ where } \mathbf{z}^k = (x_1^k; x_2^k; x_3^k; \mathbf{y}^k)$$

- Why diverges? $\rho(M) > 1$ for above A and any choice of γ.
- When you randomly choose an initial solution, it diverges with probability one!

Almost Always Diverges



Strong Convexity Does not Help

$$\begin{array}{ll} \min & 0.05x_1^2 + 0.05x_2^2 + 0.05x_3^2 \\ \text{s.t.} & \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 2 \\ 1 & 2 & 2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = 0. \end{array}$$

Strong Convexity Does not Help

$$\min \quad 0.05x_1^2 + 0.05x_2^2 + 0.05x_3^2$$
$$\text{s.t.} \quad \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 2 \\ 1 & 2 & 2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = 0.$$

- The mapping matrix M in the ADMM ($\gamma = 1$) has

$$\rho(M) = 1.0087 > 1$$

Strong Convexity Does not Help

$$\begin{array}{ll} \min & 0.05x_1^2 + 0.05x_2^2 + 0.05x_3^2 \\ \text{s.t.} & \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 2 \\ 1 & 2 & 2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = 0. \end{array}$$

- The mapping matrix M in the ADMM ($\gamma = 1$) has

$$\rho(M) = 1.0087 > 1$$

- That is, even for strongly convex programming, the ADMM is not necessarily convergent for a range of $\gamma > 0$.

The Stepsize of ADMM

For some step-size $0 < \beta < 1$, update the Lagrangian multiplier:

$$\mathbf{y}^{k+1} := \mathbf{y}^k - \beta\gamma(\sum_i A_i \mathbf{x}_i^{k+1} - \mathbf{b}).$$

The Stepsize of ADMM

For some step-size $0 < \beta < 1$, update the Lagrangian multiplier:

$$\mathbf{y}^{k+1} := \mathbf{y}^k - \beta\gamma(\sum_i A_i \mathbf{x}_i^{k+1} - \mathbf{b}).$$

- $p = 1$; (Augmented Lagrangian Method) for $\beta \in (0, 2)$, (Hestenes 69, Powell 69).
- $p = 2$; (Alternating Direction Method of Multipliers) for $\beta \in (0, \frac{1+\sqrt{5}}{2})$, (Glowinski 84).
- $p \geq 3$; for β sufficiently small provided additional conditions on the problem, (Hong/Luo 12)

Is there a Problem-Data-Independent β?

For any positive β the following linear system makes ADMM diverge

$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1+\beta \\ 1 & 1+\beta & 1+\beta \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = 0.$$

Is there a Problem-Data-Independent β?

For any positive β the following linear system makes ADMM diverge

$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1+\beta \\ 1 & 1+\beta & 1+\beta \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = 0.$$

There is no practical problem-data-independent β such that the small-step size variant would work.

Is there a Problem-Data-Independent β?

For any positive β the following linear system makes ADMM diverge

$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1+\beta \\ 1 & 1+\beta & 1+\beta \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = 0.$$

There is no practical problem-data-independent β such that the small-step size variant would work.

Many other alternatives were proposed, but somehow they all make ADMM converge slower in practice...

Randomly Permuted ADMM

Random-Permuted ADMM (RP-ADMM): each round, draw a random permutation $\sigma = (\sigma(1), \ldots, \sigma(p))$ of $\{1, \ldots, p\}$, and

$$\text{Update } \mathbf{x}_{\sigma(1)} \to \mathbf{x}_{\sigma(2)} \to \cdots \to \mathbf{x}_{\sigma(p)} \to \mathbf{y}.$$

(This is block sample without replacement)

Randomly Permuted ADMM

Random-Permuted ADMM (RP-ADMM): each round, draw a random permutation $\sigma = (\sigma(1), \ldots, \sigma(p))$ of $\{1, \ldots, p\}$, and

$$\text{Update } \mathbf{x}_{\sigma(1)} \to \mathbf{x}_{\sigma(2)} \to \cdots \to \mathbf{x}_{\sigma(p)} \to \mathbf{y}.$$

(This is block sample without replacement)

- Force "absolute fairness" among blocks or a mixed strategy on updating order.

Randomly Permuted ADMM

Random-Permuted ADMM (RP-ADMM): each round, draw a random permutation $\sigma = (\sigma(1), \ldots, \sigma(p))$ of $\{1, \ldots, p\}$, and

$$\text{Update } \mathbf{x}_{\sigma(1)} \rightarrow \mathbf{x}_{\sigma(2)} \rightarrow \cdots \rightarrow \mathbf{x}_{\sigma(p)} \rightarrow \mathbf{y}.$$

(This is block sample without replacement)

- Force "absolute fairness" among blocks or a mixed strategy on updating order.
- Random permutation has been effectively used in many areas but little theoretical analysis is known.

Randomly Permuted ADMM

Random-Permuted ADMM (RP-ADMM): each round, draw a random permutation $\sigma = (\sigma(1), \ldots, \sigma(p))$ of $\{1, \ldots, p\}$, and

$$\text{Update } \mathbf{x}_{\sigma(1)} \to \mathbf{x}_{\sigma(2)} \to \cdots \to \mathbf{x}_{\sigma(p)} \to \mathbf{y}.$$

(This is block sample without replacement)

- Force "absolute fairness" among blocks or a mixed strategy on updating order.
- Random permutation has been effectively used in many areas but little theoretical analysis is known.
- Simulation Test Result: almost always converges and fast!

Almost Always Converges when Randomize the Update Order

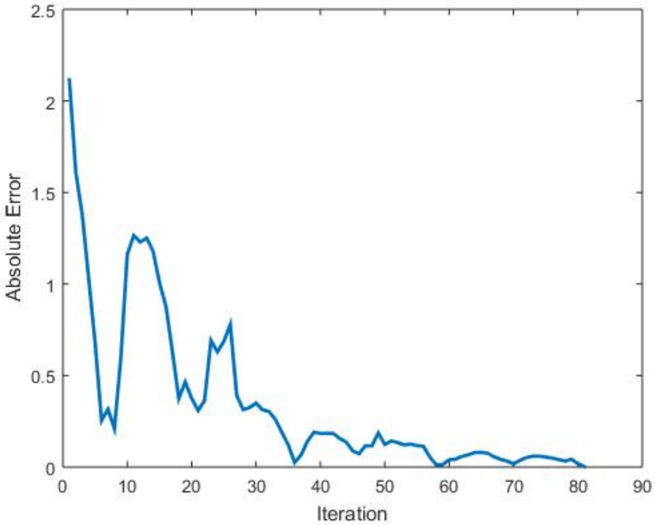


Table of Contents

1 Linear Programming and Algorithms

2 Advances in Simplex and Interior-Point Algorithms

3 First-Order and Alternating Direction Method of Multipliers

4 Convergence of RP-ADMM in Expectation

5 Why Multi-Block ADMM?

Any Theory Behind the Success?

Any Theory Behind the Success?

Consider a square system of linear equation:

$$
\begin{array}{lll}
(S) & \min\limits_{\mathbf{x}\in R^n} & 0, \\
 & s.t. & A_1\mathbf{x}_1 + \cdots + A_p\mathbf{x}_p = \mathbf{b},
\end{array}
$$

where $A = [A_1, \ldots, A_p] \in R^{n\times n}$, $\mathbf{x}_i \in R^{d_i}$ and $\sum_i d_i = n$.

Any Theory Behind the Success?

Consider a square system of linear equation:

$$(\mathrm{S}) \quad \min_{\mathbf{x}\in R^n} \quad 0, \qquad s.t. \quad A_1\mathbf{x}_1 + \cdots + A_p\mathbf{x}_p = \mathbf{b},$$

where $A = [A_1, \ldots, A_p] \in R^{n\times n}$, $\mathbf{x}_i \in R^{d_i}$ and $\sum_i d_i = n$.

After k rounds, RP-ADMM generates $\mathbf{z}^k$, an r.v. depending on

$$\boldsymbol{\xi}_k = (\sigma_1, \ldots, \sigma_k),$$

where σ_j is the randomly picked permutation at j-th round. Let

$$\phi^k \triangleq E_{\xi_k}(\mathbf{z}^k).$$

Convergence in Expectation (Sun/Luo/Y 15)

Theorem 1

The expected iterate $\phi^k \triangleq E_{\xi_k}(\mathbf{z}^k)$ converges to the solution linearly for any $1 \leq p \leq n$ if A is invertible, i.e.

$$\{\phi^k\}_{k\to\infty} \longrightarrow \begin{bmatrix} A^{-1}\mathbf{b} \\ 0 \end{bmatrix}.$$

Convergence in Expectation (Sun/Luo/Y 15)

Theorem 1

The expected iterate $\phi^k \triangleq E_{\xi_k}(\mathbf{z}^k)$ converges to the solution linearly for any $1 \le p \le n$ if A is invertible, i.e.

$$\{\phi^k\}_{k\to\infty} \longrightarrow \begin{bmatrix} A^{-1}\mathbf{b} \\ 0 \end{bmatrix}.$$

Remark: Expected convergence $\neq$ convergence, but is a strong evidence for convergence for solving most problems, e.g., when iterates are bounded.

The Average Mapping is a Contraction

- The update equation of RP-ADMM for (S) (with $\mathbf{b} = \mathbf{0}$) is

$$\mathbf{z}^{k+1} = M_\sigma \mathbf{z}^k,$$

where $M_\sigma \in R^{2n \times 2n}$ depend on σ.

The Average Mapping is a Contraction

- The update equation of RP-ADMM for (S) (with $\mathbf{b} = \mathbf{0}$) is
$$\mathbf{z}^{k+1} = M_\sigma \mathbf{z}^k,$$
where $M_\sigma \in R^{2n \times 2n}$ depend on σ.
- Define the expected update matrix as
$$M = E_\sigma(M_\sigma) = \frac{1}{m!} \sum_\sigma M_\sigma.$$

The Average Mapping is a Contraction

- The update equation of RP-ADMM for (S) (with $\mathbf{b} = \mathbf{0}$) is
$$\mathbf{z}^{k+1} = M_\sigma \mathbf{z}^k,$$
where $M_\sigma \in R^{2n \times 2n}$ depend on σ.
- Define the expected update matrix as
$$M = E_\sigma(M_\sigma) = \frac{1}{m!} \sum_\sigma M_\sigma.$$

Theorem 2

The spectral radius of M is strictly less than 1, i.e. $\rho(M) < 1$, for any $1 \le p \le n$ if A is invertible.

Proof Difficulty of Theorem 2

- Theorem 2 implies Theorem 1 is relatively easy to show, but Theorem 2...

Proof Difficulty of Theorem 2

- Theorem 2 implies Theorem 1 is relatively easy to show, but Theorem 2...
- Few tools deal with spectral radius of non-symmetric matrices.
  - E.g. $\rho(X+Y) \leq \rho(X) + \rho(Y)$ and $\rho(XY) \leq \rho(X)\rho(Y)$ don't hold.
  - Though $\rho(M) < \|M\|$, it turns out $\|M\| > 2.3$ for the counterexample.

Proof Difficulty of Theorem 2

- Theorem 2 implies Theorem 1 is relatively easy to show, but Theorem 2...
- Few tools deal with spectral radius of non-symmetric matrices.
  - E.g. $\rho(X+Y) \leq \rho(X)+\rho(Y)$ and $\rho(XY) \leq \rho(X)\rho(Y)$ don't hold.
  - Though $\rho(M) < \|M\|$, it turns out $\|M\| > 2.3$ for the counterexample.
- RP is not independent so that M is a complicated function of A.

Proof Difficulty of Theorem 2

- Theorem 2 implies Theorem 1 is relatively easy to show, but Theorem 2...
- Few tools deal with spectral radius of non-symmetric matrices.
  - E.g. $\rho(X+Y) \le \rho(X) + \rho(Y)$ and $\rho(XY) \le \rho(X)\rho(Y)$ don't hold.
  - Though $\rho(M) < \|M\|$, it turns out $\|M\| > 2.3$ for the counterexample.
- RP is not independent so that M is a complicated function of A.
- Techniques: Symmetrization and Mathematical Induction.

Two Main Lemmas to Prove Theorem 2

- **Step 1**: Relate M to a symmetric matrix AQA^T.

Lemma 1

$$\lambda \in \text{eig}(M) \Longleftrightarrow \frac{(1-\lambda)^2}{1-2\lambda} \in \text{eig}(AQA^T).$$

Sicne Q is symmetric, we have

$$\rho(M) < 1 \Longleftrightarrow \text{eig}(AQA^T) \subseteq (0, \frac{4}{3}).$$

Two Main Lemmas to Prove Theorem 2

- **Step 1**: Relate M to a symmetric matrix AQA^T.

Lemma 1

$$\lambda \in \text{eig}(M) \Longleftrightarrow \frac{(1-\lambda)^2}{1-2\lambda} \in \text{eig}(AQA^T).$$

Sicne Q is symmetric, we have

$$\rho(M) < 1 \Longleftrightarrow \text{eig}(AQA^T) \subseteq (0, \frac{4}{3}).$$

- **Step 2**: Bound eigenvalues of AQA^T - prove by induction.

Lemma 2

$$\text{eig}(AQA^T) \subseteq (0, \frac{4}{3}).$$

Two Main Lemmas to Prove Theorem 2

- **Step 1**: Relate M to a symmetric matrix AQA^T.

Lemma 1

$$\lambda \in \text{eig}(M) \Longleftrightarrow \frac{(1-\lambda)^2}{1-2\lambda} \in \text{eig}(AQA^T).$$

Sicne Q is symmetric, we have

$$\rho(M) < 1 \Longleftrightarrow \text{eig}(AQA^T) \subseteq (0, \frac{4}{3}).$$

- **Step 2**: Bound eigenvalues of AQA^T - prove by induction.

Lemma 2

$$\text{eig}(AQA^T) \subseteq (0, \frac{4}{3}).$$

- Remark: $4/3$ is "almost" tight; for $m = 3$, maximum ≈ 1.18. Increase to $4/3$ as m increases.

Fixed Cyclic ADMM vs RP-ADMM

Solving weakly Laplacian linear systems:

$$A(i,i) = 1, \quad A(i,j) = A(j,i) = \tau \cdot \text{rand}(1,1).$$

(n,τ) - (iter,time)	CycADMM	RP-ADMM
$(500, 0.01)$	diverge	(10.1, 1.3)
$(500, 0.05)$	diverge	(67, 0.61)
$(5000, 0.002)$	diverge	(7.5, 21.1)
$(5000, 0.01)$	diverge	(22, 38)
$(5000, 0.02)$	diverge	(205, 251)

Further Result on Randomized Permutation for Convex Optimization

Consider the non-separable convex quadratic problem

$$\begin{array}{ll} \min_{\mathbf{x} \in R^n} & \mathbf{x}^T H \mathbf{x} + \mathbf{c}^T \mathbf{x}, \\ s.t. & A\mathbf{x} \triangleq A_1\mathbf{x}_1 + \cdots + A_p\mathbf{x}_p = \mathbf{b}, \\ & \mathbf{x}_i \in \mathcal{X}_i \subset R^{d_i}, \; i = 1, \ldots, p. \end{array}$$

Further Result on Randomized Permutation for Convex Optimization

Consider the non-separable convex quadratic problem

$$\begin{aligned}
\min_{\mathbf{x}\in R^n} \quad & \mathbf{x}^T H\mathbf{x} + \mathbf{c}^T\mathbf{x}, \\
s.t. \quad & A\mathbf{x} \triangleq A_1\mathbf{x}_1 + \cdots + A_p\mathbf{x}_p = \mathbf{b}, \\
& \mathbf{x}_i \in \mathcal{X}_i \subset R^{d_i}, \;\; i = 1, \ldots, p.
\end{aligned}$$

Theorem 3

If each block subproblem possesses a unique solution, the expected iterate $\phi^k \triangleq E_{\xi_k}(\mathbf{z}^k)$ converges to an optimal solution of the original problem (Chen, Li, Liu and Y 15).

Table of Contents

1 Linear Programming and Algorithms

2 Advances in Simplex and Interior-Point Algorithms

3 First-Order and Alternating Direction Method of Multipliers

4 Convergence of RP-ADMM in Expectation

5 Why Multi-Block ADMM?

Universal Decomposition and Block Coordinate Descent

Consider the homogeneous and self-dual linear program to find feasible solution $(\mathbf{x}, \mathbf{y}, \mathbf{s})$ such that

$$\begin{array}{rcl} A\mathbf{x} - \mathbf{b}\tau & = & \mathbf{0}, \\ -A^T\mathbf{y} - \mathbf{s} + \mathbf{c}\tau & = & \mathbf{0}, \\ \mathbf{b}^T\mathbf{y} - \mathbf{c}^T\mathbf{x} - \kappa & = & 0, \\ \mathbf{e}^T\mathbf{x} + \tau + \mathbf{e}^T\mathbf{s} + \kappa & = & 1, \\ (\mathbf{x}, \tau, \mathbf{s}, \kappa) & \geq & \mathbf{0}, \end{array}$$

where three blocks $(\mathbf{x}, \tau)$, $\mathbf{y}$ and $(\mathbf{s}, \kappa)$ are alternatively updated (Donoghue/Chu/Parikh/Boyd 15), also see Sun and Toh 14 for SDP.

Combine ADMM and Interior-Point

Consider the logarithmic barrier function as objective:

$$
\begin{array}{llcl}
\min & -\mu \ln(\tau\kappa) - \mu \sum_j \ln(x_j s_j) & & \\
s.t. & A\mathbf{x} - \mathbf{b}\tau & = & \mathbf{0}, \\
 & -A^T\mathbf{y} - \mathbf{s} + \mathbf{c}\tau & = & \mathbf{0}, \\
 & \mathbf{b}^T\mathbf{y} - \mathbf{c}^T\mathbf{x} - \kappa & = & 0, \\
 & \mathbf{e}^T\mathbf{x} + \tau + \mathbf{e}^T\mathbf{s} + \kappa & = & 1,
\end{array}
$$

which is solved by Multi-Block ADMM.

Combine ADMM and Interior-Point

Consider the logarithmic barrier function as objective:

$$\begin{array}{lrcl} \min & -\mu \ln(\tau\kappa) - \mu \sum_j \ln(x_j s_j) & & \\ s.t. & A\mathbf{x} - \mathbf{b}\tau & = & \mathbf{0}, \\ & -A^T\mathbf{y} - \mathbf{s} + \mathbf{c}\tau & = & \mathbf{0}, \\ & \mathbf{b}^T\mathbf{y} - \mathbf{c}^T\mathbf{x} - \kappa & = & 0, \\ & \mathbf{e}^T\mathbf{x} + \tau + \mathbf{e}^T\mathbf{s} + \kappa & = & 1, \end{array}$$

which is solved by Multi-Block ADMM.

Then μ is gradually reduced to 0 as in interior-point methods – initial computational results are encouraging (Lin/Ma 15)!

Implementation on Distributed Platforms I

Reformulate

$$\begin{array}{lll} \text{minimize}_{\mathbf{x}} & \mathbf{c}^T\mathbf{x} & \\ \text{subject to} & A\mathbf{x} & = \mathbf{b}, \\ & \mathbf{x} & \geq \mathbf{0}, \end{array} \quad \text{as}$$

$$\begin{array}{llll} \text{minimize}_{\mathbf{x}_0,\ldots,\mathbf{x}_m} & \mathbf{c}^T\mathbf{x}_0 & & \\ \text{subject to} & \mathbf{a}_i\mathbf{x}_i & = b_i, & i = 1, \ldots, m \\ & \mathbf{x}_i - \mathbf{x}_0 & = \mathbf{0}, & i = 1, \ldots, m, \\ & \mathbf{x}_0 & \geq \mathbf{0}. & \end{array}$$

Implementation on Distributed Platforms I

Reformulate

$$\begin{array}{ll} \text{minimize}_{\mathbf{x}} & \mathbf{c}^T\mathbf{x} \\ \text{subject to} & A\mathbf{x} = \mathbf{b}, \\ & \mathbf{x} \geq \mathbf{0}, \end{array} \quad \text{as}$$

$$\begin{array}{lll} \text{minimize}_{\mathbf{x}_0,\dots,\mathbf{x}_m} & \mathbf{c}^T\mathbf{x}_0 & \\ \text{subject to} & \mathbf{a}_i\mathbf{x}_i & = b_i, \;\; i = 1, \dots, m \\ & \mathbf{x}_i - \mathbf{x}_0 & = \mathbf{0}, \;\; i = 1, \dots, m, \\ & \mathbf{x}_0 & \geq \mathbf{0}. \end{array}$$

Then apply Multi-Block ADMM to the new formulation (He and Yuan 15, Sun/Y 15).

Implementation on Distributed Platforms II

In each round of ADMM, for $i = 1, ..., m$, for a simple quadratic objective one independently solves

$$\begin{array}{lll} \text{minimize}_{\mathbf{x}_i} & q_i(\mathbf{x}_i) & \\ \text{subject to} & \mathbf{a}_i\mathbf{x} & = b_i; \end{array}$$

then, update $\mathbf{x}_0$ as

$$\mathbf{x}_0 = \max\{\frac{1}{m}\sum_i \mathbf{x}_i,\ 0\}.$$

Implementation on Distributed Platforms II

In each round of ADMM, for $i = 1, ..., m$, for a simple quadratic objective one independently solves

$$\begin{array}{lll} \text{minimize}_{\mathbf{x}_i} & q_i(\mathbf{x}_i) & \\ \text{subject to} & \mathbf{a}_i\mathbf{x} & = b_i; \end{array}$$

then, update $\mathbf{x}_0$ as

$$\mathbf{x}_0 = \max\{\frac{1}{m}\sum_i \mathbf{x}_i,\ 0\}. \quad \text{Or}$$

$$\begin{array}{lll} \text{minimize}_{\mathbf{x}_i} & q_i(\mathbf{x}_i) & \\ \text{subject to} & \mathbf{a}_i\mathbf{x} & = b_i,\ \mathbf{x}_i \geq \mathbf{0}; \end{array}$$

$$\mathbf{x}_0 = \frac{1}{m}\sum_i \mathbf{x}_i.$$

Summary and Future Directions

- Could MDP be solved in strongly polynomial time regardless discount factors?

Summary and Future Directions

- Could MDP be solved in strongly polynomial time regardless discount factors?
- Could the recent theoretically better interior-point algorithms be realized in practice?

Summary and Future Directions

- Could MDP be solved in strongly polynomial time regardless discount factors?
- Could the recent theoretically better interior-point algorithms be realized in practice?
- Could the RP-ADMM convergence be with high probability, and/or the RP-ADMM result extended to more general convex optimization problems?

Summary and Future Directions

- Could MDP be solved in strongly polynomial time regardless discount factors?
- Could the recent theoretically better interior-point algorithms be realized in practice?
- Could the RP-ADMM convergence be with high probability, and/or the RP-ADMM result extended to more general convex optimization problems?
- Big Picture: Are there other competitive linear programming algorithms in both theory and practice?

Summary and Future Directions

- Could MDP be solved in strongly polynomial time regardless discount factors?
- Could the recent theoretically better interior-point algorithms be realized in practice?
- Could the RP-ADMM convergence be with high probability, and/or the RP-ADMM result extended to more general convex optimization problems?
- Big Picture: Are there other competitive linear programming algorithms in both theory and practice?
- Multi-Block ADMM might have a chance (at least in practice):

Summary and Future Directions

- Could MDP be solved in strongly polynomial time regardless discount factors?
- Could the recent theoretically better interior-point algorithms be realized in practice?
- Could the RP-ADMM convergence be with high probability, and/or the RP-ADMM result extended to more general convex optimization problems?
- Big Picture: Are there other competitive linear programming algorithms in both theory and practice?
- Multi-Block ADMM might have a chance (at least in practice):
  - Good theories have been established.

Summary and Future Directions

- Could MDP be solved in strongly polynomial time regardless discount factors?
- Could the recent theoretically better interior-point algorithms be realized in practice?
- Could the RP-ADMM convergence be with high probability, and/or the RP-ADMM result extended to more general convex optimization problems?
- Big Picture: Are there other competitive linear programming algorithms in both theory and practice?
- Multi-Block ADMM might have a chance (at least in practice):
  - Good theories have been established.
  - Could be easily implemented in a distributed way on a cloud and/or GPU platform.

Summary and Future Directions

- Could MDP be solved in strongly polynomial time regardless discount factors?
- Could the recent theoretically better interior-point algorithms be realized in practice?
- Could the RP-ADMM convergence be with high probability, and/or the RP-ADMM result extended to more general convex optimization problems?
- Big Picture: Are there other competitive linear programming algorithms in both theory and practice?
- Multi-Block ADMM might have a chance (at least in practice):
  - Good theories have been established.
  - Could be easily implemented in a distributed way on a cloud and/or GPU platform.
- Linear Programming Research Continues ...