Inexact ADMM for Distributed Optimal Control Problems with Parabolic Equation Constraints

Xiaoming Yuan

The University of Hong Kong

Joint work with

Yongcun Song and Hangrui Yue

November 21, 2019

Outline

Outline

Augmented Lagrangian Method (ALM)

Augmented Lagrangian Method (ALM)

- A canonical convex minimization model with linear constraints:

$$\min\{\theta(x) \mid Ax = b,\ x \in \mathcal{X}\},$$

  - $A : H_1 \to H_2$, $b \in H_2$, with H_1 and H_2 proper Hilbert spaces; $\mathcal{X} \subseteq H_1$: a closed and convex simple set;
  - $\theta(x) : H_1 \to \mathbb{R}$: a convex but not necessarily smooth function.

Augmented Lagrangian Method (ALM)

- A canonical convex minimization model with linear constraints:

$$\min\{\theta(x) \mid Ax = b,\ x \in \mathcal{X}\},$$

  - $A : H_1 \to H_2$, $b \in H_2$, with H_1 and H_2 proper Hilbert spaces; $\mathcal{X} \subseteq H_1$: a closed and convex simple set;
  - $\theta(x) : H_1 \to \mathbb{R}$: a convex but not necessarily smooth function.

- The augmented Lagrangian method (proposed by H. Hestenes and M. Powell in 1969, individually) reads as:

$$\begin{cases} x^{k+1} & = \arg\min_{x\in\mathcal{X}} \left\{\theta(x) - (\lambda^k, Ax - b) + \frac{\beta}{2}\|Ax - b\|^2\right\}, \\ \lambda^{k+1} & = \lambda^k - \beta\left(Ax^{k+1} - b\right), \end{cases}$$

  where $\lambda \in H_2$ is the Lagrange multiplier and $\beta > 0$ is a penalty parameter.

Alternating Direction Method of Multipliers – 1/2

- Many applications have the separable structure:

$$\begin{aligned} \min_{x_1 \in H_1, x_2 \in H_2} \quad & \theta_1(x_1) + \theta_2(x_2), \\ s.t. \quad & A_1 x_1 + A_2 x_2 = b. \end{aligned}$$

  - $A_1 : H_1 \to H$, $A_2 : H_2 \to H$, $b \in H$, with H_1, H_2 and H proper Hilbert spaces;
  - $\theta_1(x_1)$ and $\theta_2(x_2)$ are convex but not necessarily smooth functions.

Alternating Direction Method of Multipliers – 1/2

- Many applications have the separable structure:

$$\begin{aligned} \min_{x_1 \in H_1, x_2 \in H_2} \quad & \theta_1(x_1) + \theta_2(x_2), \\ s.t. \quad & A_1 x_1 + A_2 x_2 = b. \end{aligned}$$

  - $A_1 : H_1 \to H$, $A_2 : H_2 \to H$, $b \in H$, with H_1, H_2 and H proper Hilbert spaces;
  - $\theta_1(x_1)$ and $\theta_2(x_2)$ are convex but not necessarily smooth functions.
- Applying ALM directly:

$$\begin{cases} (x_1^{k+1}, x_2^{k+1}) = \arg\min\limits_{\substack{x_1 \in H_1 \\ x_2 \in H_2}} \left\{ \sum\limits_{i=1}^{2} \theta_i(x_i) - (\lambda^k, \sum\limits_{i=1}^{2} A_i x_i - b) + \frac{\beta}{2} \left\| \sum\limits_{i=1}^{2} A_i x_i - b \right\|^2 \right\}, \\ \lambda^{k+1} = \lambda^k - \beta \left(A_1 x_1^{k+1} + A_2 x_2^{k+1} - b \right). \end{cases}$$

Alternating Direction Method of Multipliers – 1/2

- Many applications have the separable structure:

$$\begin{aligned} \min_{x_1 \in H_1, x_2 \in H_2} \quad & \theta_1(x_1) + \theta_2(x_2), \\ s.t. \quad & A_1 x_1 + A_2 x_2 = b. \end{aligned}$$

  - $A_1 : H_1 \to H$, $A_2 : H_2 \to H$, $b \in H$, with H_1, H_2 and H proper Hilbert spaces;
  - $\theta_1(x_1)$ and $\theta_2(x_2)$ are convex but not necessarily smooth functions.
- Applying ALM directly:

$$\begin{cases} (x_1^{k+1}, x_2^{k+1}) = \arg\min\limits_{\substack{x_1 \in H_1 \\ x_2 \in H_2}} \left\{ \sum\limits_{i=1}^{2} \theta_i(x_i) - (\lambda^k, \sum\limits_{i=1}^{2} A_i x_i - b) + \frac{\beta}{2} \left\| \sum\limits_{i=1}^{2} A_i x_i - b \right\|^2 \right\}, \\ \lambda^{k+1} = \lambda^k - \beta \left(A_1 x_1^{k+1} + A_2 x_2^{k+1} - b \right). \end{cases}$$

- It is not easy to solve the ALM subproblem exactly even when both θ_1 and θ_2 are very simple.

Alternating Direction Method of Multipliers – 2/2

- Alternating Direction Method of Multipliers (ADMM) was originally proposed by R. Glowinski and A. Marrocco in 1975 for solving nonlinear elliptic equations:

Alternating Direction Method of Multipliers – 2/2

- Alternating Direction Method of Multipliers (ADMM) was originally proposed by R. Glowinski and A. Marrocco in 1975 for solving nonlinear elliptic equations:

$$\begin{cases} x_1^{k+1} = \arg\min\limits_{x_1 \in H_1} \left\{ \theta_1(x_1) - (\lambda^k, A_1 x_1 + A_2 x_2^k - b) + \frac{\beta}{2} \left\| A_1 x_1 + A_2 x_2^k - b \right\|^2 \right\} \\ x_2^{k+1} = \arg\min\limits_{x_2 \in H_2} \left\{ \theta_2(x_2) - (\lambda^k, A_1 x_1^{k+1} + A_2 x_2 - b) + \frac{\beta}{2} \left\| A_1 x_1^{k+1} + A_2 x_2 - b \right\|^2 \right\} \\ \lambda^{k+1} = \lambda^k - \beta \left(A_1 x_1^{k+1} + A_2 x_2^{k+1} - b \right) \end{cases}$$

Alternating Direction Method of Multipliers – 2/2

- Alternating Direction Method of Multipliers (ADMM) was originally proposed by R. Glowinski and A. Marrocco in 1975 for solving nonlinear elliptic equations:

$$\begin{cases} x_1^{k+1} = \arg\min\limits_{x_1 \in H_1} \left\{ \theta_1(x_1) - (\lambda^k, A_1 x_1 + A_2 x_2^k - b) + \frac{\beta}{2} \left\| A_1 x_1 + A_2 x_2^k - b \right\|^2 \right\} \\ x_2^{k+1} = \arg\min\limits_{x_2 \in H_2} \left\{ \theta_2(x_2) - (\lambda^k, A_1 x_1^{k+1} + A_2 x_2 - b) + \frac{\beta}{2} \left\| A_1 x_1^{k+1} + A_2 x_2 - b \right\|^2 \right\} \\ \lambda^{k+1} = \lambda^k - \beta \left(A_1 x_1^{k+1} + A_2 x_2^{k+1} - b \right) \end{cases}$$

- It can be regarded as a Gauss-Seidel splitting version of the ALM.

Douglas-Rachford Splitting Method – 1/2

Douglas-Rachford Splitting Method – 1/2

- The Douglas-Rachford Splitting (DRSM) method was first studied in 1956 by Douglas and Rachford for heat conduction equations and then generalized by Lions and Mercier in 1979 to nonlinear cases.

Douglas-Rachford Splitting Method – 1/2

- The Douglas-Rachford Splitting (DRSM) method was first studied in 1956 by Douglas and Rachford for heat conduction equations and then generalized by Lions and Mercier in 1979 to nonlinear cases.
- It turns out that there is a close relationship between the ADMM and DRSM.

Douglas-Rachford Splitting Method – 1/2

- The Douglas-Rachford Splitting (DRSM) method was first studied in 1956 by Douglas and Rachford for heat conduction equations and then generalized by Lions and Mercier in 1979 to nonlinear cases.
- It turns out that there is a close relationship between the ADMM and DRSM.
- Recall the primal problem:

$$\begin{aligned} \min_{x_1 \in H_1, x_2 \in H_2} \quad & \theta_1(x_1) + \theta_2(x_2), \\ s.t. \quad & A_1 x_1 + A_2 x_2 = b. \end{aligned}$$

Douglas-Rachford Splitting Method – 1/2

- The Douglas-Rachford Splitting (DRSM) method was first studied in 1956 by Douglas and Rachford for heat conduction equations and then generalized by Lions and Mercier in 1979 to nonlinear cases.
- It turns out that there is a close relationship between the ADMM and DRSM.
- Recall the primal problem:

$$\begin{aligned} \min_{x_1 \in H_1, x_2 \in H_2} \quad & \theta_1(x_1) + \theta_2(x_2), \\ s.t. \quad & A_1 x_1 + A_2 x_2 = b. \end{aligned}$$

- The dual problem is:

$$\max_{\lambda \in H} -\theta_1^*(A_1^*\lambda) - \theta_2^*(A_2^*\lambda) - (b, \lambda),$$

where A_i^* is the adjoint operator of A_i and θ_i^* denotes the conjugate function of θ_i , i.e. $\theta_i^*(A_i^*\lambda) = \sup_{x_i \in H_i}\{(A_i^*\lambda, x_i) - \theta_i(x_i)\}$.

Douglas-Rachford Splitting Method – 2/2

- The dual problem can be written as $0 \in A_1 \partial\theta_1^*(A_1^*\lambda) + A_2 \partial\theta_2^*(A_2^*\lambda) + b$, where $\partial\theta_i^*$ denotes the subdifferential of θ_i^*.

Douglas-Rachford Splitting Method – 2/2

- The dual problem can be written as $0 \in A_1 \partial\theta_1^*(A_1^*\lambda) + A_2 \partial\theta_2^*(A_2^*\lambda) + b$, where $\partial\theta_i^*$ denotes the subdifferential of θ_i^*.
- We associate the dual problem with the multivalued initial value problem:

$$\begin{cases} 0 \in \dfrac{d\lambda}{dt} + A_1 \partial\theta_1^*(A_1^*\lambda) + A_2 \partial\theta_2^*(A_2^*\lambda) + b, \\ \lambda(0) = \lambda_0. \end{cases}$$

Douglas-Rachford Splitting Method – 2/2

- The dual problem can be written as $0 \in A_1 \partial\theta_1^*(A_1^*\lambda) + A_2 \partial\theta_2^*(A_2^*\lambda) + b$, where $\partial\theta_i^*$ denotes the subdifferential of θ_i^*.
- We associate the dual problem with the multivalued initial value problem:

$$\begin{cases} 0 \in \dfrac{d\lambda}{dt} + A_1 \partial\theta_1^*(A_1^*\lambda) + A_2 \partial\theta_2^*(A_2^*\lambda) + b, \\ \lambda(0) = \lambda_0. \end{cases}$$

- The DRSM with time stepsize τ reads:

$$\begin{cases} 0 \in \dfrac{\hat{\lambda}^{k+1} - \lambda^k}{\tau} + A_1 \partial\theta_1^*(A_1^*\hat{\lambda}^{k+1}) + A_2 \partial\theta_2^*(A_2^*\lambda^k) + b, \\ 0 \in \dfrac{\lambda^{k+1} - \lambda^k}{\tau} + A_1 \partial\theta_1^*(A_1^*\hat{\lambda}^{k+1}) + A_2 \partial\theta_2^*(A_2^*\lambda^{k+1}) + b, \end{cases}$$

where $\lambda^0 = \lambda_0$, and $k \geq 0$.

Douglas-Rachford Splitting Method – 2/2

- The dual problem can be written as $0 \in A_1 \partial\theta_1^*(A_1^*\lambda) + A_2 \partial\theta_2^*(A_2^*\lambda) + b$, where $\partial\theta_i^*$ denotes the subdifferential of θ_i^*.
- We associate the dual problem with the multivalued initial value problem:

$$\begin{cases} 0 \in \dfrac{d\lambda}{dt} + A_1 \partial\theta_1^*(A_1^*\lambda) + A_2 \partial\theta_2^*(A_2^*\lambda) + b, \\ \lambda(0) = \lambda_0. \end{cases}$$

- The DRSM with time stepsize τ reads:

$$\begin{cases} 0 \in \dfrac{\hat{\lambda}^{k+1} - \lambda^k}{\tau} + A_1 \partial\theta_1^*(A_1^*\hat{\lambda}^{k+1}) + A_2 \partial\theta_2^*(A_2^*\lambda^k) + b, \\ 0 \in \dfrac{\lambda^{k+1} - \lambda^k}{\tau} + A_1 \partial\theta_1^*(A_1^*\hat{\lambda}^{k+1}) + A_2 \partial\theta_2^*(A_2^*\lambda^{k+1}) + b, \end{cases}$$

  where $\lambda^0 = \lambda_0$, and $k \geq 0$.
- The ADMM applied to the primal problem is equivalent to the DRSM applied to the dual problem with a proper time step size and an initial value (see e.g., Chan and Glowinski 1978, Gabay 1983).

Convergence Results

- Convergence analysis of DRSM can be found in Douglas and Rachford 1956, Lions and Mercier 1979, Eckstein and Bertsekas 1992, Corman and Y. 2013, He and Y. 2015.

Convergence Results

- Convergence analysis of DRSM can be found in Douglas and Rachford 1956, Lions and Mercier 1979, Eckstein and Bertsekas 1992, Corman and Y. 2013, He and Y. 2015.
- Convergence analysis of ADMM and its variants dates back to Gabay 1976, Glowinski 1984, Eckstein 1994, He and Yang 1998, He *et al.* 2002, He and Y. 2011, He and Y. 2015, Liu, Y., Zeng and Zhang 2018.

Popularity of ADMM

- A “renaissance" of ADMM in many application domains.
- Google “ADMM" returns 1,340,000 outputs on November 17, 2019.
- Review papers: Boyd *et al.* 2010, Glowinski 2012, Eckstein and Yao 2012, He and Y. 201X.

Outline

LASSO

- Consider the large-scale least absolute shrinkage and selection operator (LASSO) model:

$$\min\{\frac{1}{2}\|Ax-b\|_2^2+\tau\|x\|_1\},$$

where $x\in\mathbb{R}^n$, $A\in\mathbb{R}^{m\times n}$ with $m\ll n$, $b\in\mathbb{R}^m$ and $\tau>0$.

LASSO

- Consider the large-scale least absolute shrinkage and selection operator (LASSO) model:

$$\min\{\frac{1}{2}\|Ax-b\|_2^2+\tau\|x\|_1\},$$

where $x\in\mathbb{R}^n$, $A\in\mathbb{R}^{m\times n}$ with $m\ll n$, $b\in\mathbb{R}^m$ and $\tau>0$.

- Reformulate it as

$$\begin{array}{ll}\min & \frac{1}{2}\|Ax-b\|_2^2+\tau\|y\|_1\\ \text{s.t.} & x=y.\end{array}$$

LASSO

- Consider the large-scale least absolute shrinkage and selection operator (LASSO) model:

$$\min\{\frac{1}{2}\|Ax-b\|_2^2+\tau\|x\|_1\},$$

where $x\in\mathbb{R}^n$, $A\in\mathbb{R}^{m\times n}$ with $m\ll n$, $b\in\mathbb{R}^m$ and $\tau>0$.

- Reformulate it as

$$\begin{array}{ll}\min & \frac{1}{2}\|Ax-b\|_2^2+\tau\|y\|_1\\ \text{s.t.} & x=y.\end{array}$$

- ADMM iterations:

1. x^{k+1}: $(\beta I+A^TA)x=A^Tb+\beta y^k-\lambda^k$;
2. $y^{k+1}=\arg\min\limits_{y\in\mathcal{R}^n}\ \{\tau\|y\|_1+\frac{\beta}{2}\left\|y-x^{k+1}+\frac{\lambda^k}{\beta}\right\|_2^2\}$;
3. $\lambda^{k+1}=\lambda^k-\beta\left(x^{k+1}-y^{k+1}\right)$.

High-dimension Concerns

High-dimension Concerns

- The main computation is solving the system of linear equations

$$(\beta I + A^T A)x = A^T b + \beta y^k - \lambda^k,$$

High-dimension Concerns

- The main computation is solving the system of linear equations

$$(\beta I + A^T A)x = A^T b + \beta y^k - \lambda^k,$$

whose solution can be obtained theoretically by a direct method ($O(n^3)$ flops) or iteratively by standard numerical linear algebra solvers such as Jacobian, Gauss-Seidel, CG, PCG (sometimes by faster solvers like DFT, FFT).

High-dimension Concerns

- The main computation is solving the system of linear equations

$$(\beta I + A^T A)x = A^T b + \beta y^k - \lambda^k,$$

  whose solution can be obtained theoretically by a direct method ($O(n^3)$ flops) or iteratively by standard numerical linear algebra solvers such as Jacobian, Gauss-Seidel, CG, PCG (sometimes by faster solvers like DFT, FFT).
- What if the dimension is extremely high?

High-dimension Concerns

- The main computation is solving the system of linear equations

$$(\beta I + A^T A)x = A^T b + \beta y^k - \lambda^k,$$

  whose solution can be obtained theoretically by a direct method ($O(n^3)$ flops) or iteratively by standard numerical linear algebra solvers such as Jacobian, Gauss-Seidel, CG, PCG (sometimes by faster solvers like DFT, FFT).
- What if the dimension is extremely high? — too computationally expensive even for CG (with n-iteration termination).

High-dimension Concerns

- The main computation is solving the system of linear equations

$$(\beta I + A^T A)x = A^T b + \beta y^k - \lambda^k,$$

  whose solution can be obtained theoretically by a direct method ($O(n^3)$ flops) or iteratively by standard numerical linear algebra solvers such as Jacobian, Gauss-Seidel, CG, PCG (sometimes by faster solvers like DFT, FFT).
- What if the dimension is extremely high? — too computationally expensive even for CG (with n-iteration termination).
- More importantly, there is no obvious justification to solve the subproblems too accurately (especially at the first stage of iterations).

High-dimension Concerns

- The main computation is solving the system of linear equations

$$(\beta I + A^T A)x = A^T b + \beta y^k - \lambda^k,$$

  whose solution can be obtained theoretically by a direct method ($O(n^3)$ flops) or iteratively by standard numerical linear algebra solvers such as Jacobian, Gauss-Seidel, CG, PCG (sometimes by faster solvers like DFT, FFT).
- What if the dimension is extremely high? — too computationally expensive even for CG (with n-iteration termination).
- More importantly, there is no obvious justification to solve the subproblems too accurately (especially at the first stage of iterations).
- How to solve these subproblems approximately (engineers know this much better practically) and what is the rigorous math theory inside (mathematicians' duty)?

Inexact ADMM

[1] "Implementing the ADMM to big datasets: A case study of LASSO", SIAM Journal on Scientific Computing, 40(5), A3121-A3156, 2018.

Inexact ADMM

- Answer – solving them inexactly:

$$\begin{cases} x_1^{k+1} \approx \arg\min\{\theta_1(x_1) - (\lambda^k)^T(A_1x_1) + \frac{\beta}{2}\|A_1x_1 + A_2x_2^k - b\|^2 \mid x_1 \in \mathcal{X}_1\}, \\ x_2^{k+1} \approx \arg\min\{\theta_2(x_2) - (\lambda^k)^T(A_2x_2) + \frac{\beta}{2}\|A_1x_1^{k+1} + A_2x_2 - b\|^2 \mid x_2 \in \mathcal{X}_2\}, \\ \lambda^{k+1} = \lambda^k - \beta(A_1x_1^{k+1} + A_2x_2^{k+1} - b). \end{cases}$$

[1] "Implementing the ADMM to big datasets: A case study of LASSO", SIAM Journal on Scientific Computing, 40(5), A3121-A3156, 2018.

Inexact ADMM

- Answer – solving them inexactly:

$$\begin{cases} x_1^{k+1} \approx \arg\min\{\theta_1(x_1) - (\lambda^k)^T(A_1x_1) + \frac{\beta}{2}\|A_1x_1 + A_2x_2^k - b\|^2 \mid x_1 \in \mathcal{X}_1\}, \\ x_2^{k+1} \approx \arg\min\{\theta_2(x_2) - (\lambda^k)^T(A_2x_2) + \frac{\beta}{2}\|A_1x_1^{k+1} + A_2x_2 - b\|^2 \mid x_2 \in \mathcal{X}_2\}, \\ \lambda^{k+1} = \lambda^k - \beta(A_1x_1^{k+1} + A_2x_2^{k+1} - b). \end{cases}$$

- How to define "$\approx$" mathematically and rigorously?

[1] "Implementing the ADMM to big datasets: A case study of LASSO", SIAM Journal on Scientific Computing, 40(5), A3121-A3156, 2018.

Inexact ADMM

- Answer – solving them inexactly:

$$\begin{cases} x_1^{k+1} \approx \arg\min\{\theta_1(x_1) - (\lambda^k)^T(A_1x_1) + \frac{\beta}{2}\|A_1x_1 + A_2x_2^k - b\|^2 \mid x_1 \in \mathcal{X}_1\}, \\ x_2^{k+1} \approx \arg\min\{\theta_2(x_2) - (\lambda^k)^T(A_2x_2) + \frac{\beta}{2}\|A_1x_1^{k+1} + A_2x_2 - b\|^2 \mid x_2 \in \mathcal{X}_2\}, \\ \lambda^{k+1} = \lambda^k - \beta(A_1x_1^{k+1} + A_2x_2^{k+1} - b). \end{cases}$$

- How to define "$\approx$" mathematically and rigorously? — see, e.g., Eckstein and Bertsekas 1992; He *et al.* 2002; Y., 2005; Ng, Wang and Y., 2011.

[1] "Implementing the ADMM to big datasets: A case study of LASSO", SIAM Journal on Scientific Computing, 40(5), A3121-A3156, 2018.

Inexact ADMM

- Answer – solving them inexactly:

$$\begin{cases} x_1^{k+1} \approx \arg\min\{\theta_1(x_1) - (\lambda^k)^T(A_1x_1) + \frac{\beta}{2}\|A_1x_1 + A_2x_2^k - b\|^2 \mid x_1 \in \mathcal{X}_1\}, \\ x_2^{k+1} \approx \arg\min\{\theta_2(x_2) - (\lambda^k)^T(A_2x_2) + \frac{\beta}{2}\|A_1x_1^{k+1} + A_2x_2 - b\|^2 \mid x_2 \in \mathcal{X}_2\}, \\ \lambda^{k+1} = \lambda^k - \beta(A_1x_1^{k+1} + A_2x_2^{k+1} - b). \end{cases}$$

- How to define "$\approx$" mathematically and rigorously? — see, e.g., Eckstein and Bertsekas 1992; He *et al.* 2002; Y., 2005; Ng, Wang and Y., 2011.
- Yue/Yang/Wang/Y.(2018) [1] proposed a specific inexactness criterion for LASSO with high dimensionality, which can save computation while ensure convergence.

[1] "Implementing the ADMM to big datasets: A case study of LASSO", SIAM Journal on Scientific Computing, 40(5), A3121-A3156, 2018.

Observation

Observation

- The subproblem for LASSO is

$$x^{k+1} = \arg\min_{x\in\mathbb{R}^n} \left\{ \frac{1}{2}\|Ax-b\|_2^2 + \frac{\beta}{2}\left\| x - y^k - \frac{\lambda^k}{\beta}\right\|_2^2 \right\},$$

Observation

- The subproblem for LASSO is

$$x^{k+1} = \arg\min_{x\in\mathbb{R}^n} \left\{ \frac{1}{2}\|Ax-b\|_2^2 + \frac{\beta}{2}\left\| x - y^k - \frac{\lambda^k}{\beta}\right\|_2^2 \right\},$$

- Thus, x^{k+1} is given by the system of linear equations:

$$Hx^{k+1} = h^k$$

with

$$H := \left(A^\top A + \beta I\right) \quad \text{and} \quad h^k := A^\top b + \beta\left(y^k + \frac{\lambda^k}{\beta}\right).$$

Observation

- The subproblem for LASSO is

$$x^{k+1} = \arg\min_{x\in\mathbb{R}^n} \left\{ \frac{1}{2}\|Ax-b\|_2^2 + \frac{\beta}{2}\left\| x - y^k - \frac{\lambda^k}{\beta}\right\|_2^2 \right\},$$

- Thus, x^{k+1} is given by the system of linear equations:

$$Hx^{k+1} = h^k$$

with

$$H := \left(A^\top A + \beta I\right) \quad \text{and} \quad h^k := A^\top b + \beta\left(y^k + \frac{\lambda^k}{\beta}\right).$$

- More explicitly, we have

$$x^{k+1} = H^{-1}h^k = \left(A^\top A + \beta I\right)^{-1}\left(A^\top b + \beta\left(y^k + \frac{\lambda^k}{\beta}\right)\right).$$

Cont'd

- Recall that our interest is the big-data scenario where n is huge. It is thus not preferable to directly solve the linear system with the matrix H in dimension of $n \times n$.

Cont'd

- Recall that our interest is the big-data scenario where n is huge. It is thus not preferable to directly solve the linear system with the matrix H in dimension of $n \times n$.
- Alternatively, we can calculate x^{k+1} via the following process:

$$\hat{H}\eta^{k+1} = \frac{1}{\beta}Ah^k,$$

$$x^{k+1} = \frac{1}{\beta}\left(h^k - A^\top \eta^{k+1}\right),$$

with

$$\hat{H} := \frac{1}{\beta}AA^\top + I \in \mathbb{R}^{m\times m}.$$

Cont'd

- Recall that our interest is the big-data scenario where n is huge. It is thus not preferable to directly solve the linear system with the matrix H in dimension of $n \times n$.
- Alternatively, we can calculate x^{k+1} via the following process:

$$\hat{H}\eta^{k+1} = \frac{1}{\beta}Ah^k,$$

$$x^{k+1} = \frac{1}{\beta}\left(h^k - A^\top \eta^{k+1}\right),$$

with

$$\hat{H} := \frac{1}{\beta}AA^\top + I \in \mathbb{R}^{m\times m}.$$

- Recall $m \ll n$.

Big-Data LASSO

- For big-data scenarios of LASSO, even m could be still huge and hence it is not preferable to solve the simplified linear system exactly by a direct method or inexactly up to a very high accuracy by an iterative method. Thus, we need to further consider how to solve it inexactly.

Big-Data LASSO

- For big-data scenarios of LASSO, even m could be still huge and hence it is not preferable to solve the simplified linear system exactly by a direct method or inexactly up to a very high accuracy by an iterative method. Thus, we need to further consider how to solve it inexactly.
- Obviously, its residual is $e_k(\eta) := \frac{1}{\beta} A h^k - \hat{H}\eta$.

Big-Data LASSO

- For big-data scenarios of LASSO, even m could be still huge and hence it is not preferable to solve the simplified linear system exactly by a direct method or inexactly up to a very high accuracy by an iterative method. Thus, we need to further consider how to solve it inexactly.
- Obviously, its residual is $e_k(\eta) := \frac{1}{\beta} A h^k - \hat{H}\eta$.
- We suggest finding an approximate solution of the linear system, η^{k+1}, such that

$$\left\| e_k\left(\eta^{k+1}\right)\right\|_2 \leq \sigma \left\| e_k\left(\eta^{k}\right)\right\|_2,$$

where σ is an arbitrary constant satisfying

$$0 < \sigma < \frac{\sqrt{2\beta}}{\sqrt{2\beta} + \|A\|_2} \in (0,1).$$

Cont'd

- The parameter σ measures the relative error of $\|e_k(\eta^k)\|_2$ and it plays the role of controlling the accuracy of solving the linear system inexactly.

Cont'd

- The parameter σ measures the relative error of $\|e_k(\eta^k)\|_2$ and it plays the role of controlling the accuracy of solving the linear system inexactly.
- It is worth noting that σ is fixed as a constant; hence the new inexactness criterion significantly differs from those in existing literatures which need a sequence of accuracy constants and require summable conditions on the sequence.

Cont'd

- The parameter σ measures the relative error of $\|e_k(\eta^k)\|_2$ and it plays the role of controlling the accuracy of solving the linear system inexactly.
- It is worth noting that σ is fixed as a constant; hence the new inexactness criterion significantly differs from those in existing literatures which need a sequence of accuracy constants and require summable conditions on the sequence.
- Moreover, specifying the sequence of accuracy constants a prior (which must be manually) is very challenging and inappropriate values may easily deteriorate the numerical performance of the mentioned inexact versions of the ADMM in the literature.

Cont'd

- The parameter σ measures the relative error of $\|e_k(\eta^k)\|_2$ and it plays the role of controlling the accuracy of solving the linear system inexactly.
- It is worth noting that σ is fixed as a constant; hence the new inexactness criterion significantly differs from those in existing literatures which need a sequence of accuracy constants and require summable conditions on the sequence.
- Moreover, specifying the sequence of accuracy constants a prior (which must be manually) is very challenging and inappropriate values may easily deteriorate the numerical performance of the mentioned inexact versions of the ADMM in the literature.
- The new inexactness criterion, however, is fully automatic for numerical implementation by just choosing a constant.

More Details

More Details

- When a standard solver is determined for this system of linear equations, we can precisely specify a bound of the maximal number for internal iterations (denoted by N_0) to ensure the proposed criterion

More Details

- When a standard solver is determined for this system of linear equations, we can precisely specify a bound of the maximal number for internal iterations (denoted by N_0) to ensure the proposed criterion — this is a conservative but automatic (user-friendly) strategy.

More Details

- When a standard solver is determined for this system of linear equations, we can precisely specify a bound of the maximal number for internal iterations (denoted by N_0) to ensure the proposed criterion — this is a conservative but automatic (user-friendly) strategy.
- If PCG is used, $N_0 = \left\lceil \log_c \left(\frac{\sqrt{2\beta}}{\sqrt{2\beta}+\|A\|_2} / \left(2\sqrt{\kappa(\hat{H})} \right) \right) \right\rceil$,

More Details

- When a standard solver is determined for this system of linear equations, we can precisely specify a bound of the maximal number for internal iterations (denoted by N_0) to ensure the proposed criterion — this is a conservative but automatic (user-friendly) strategy.
- If PCG is used, $N_0 = \left\lceil \log_c \left(\frac{\sqrt{2\beta}}{\sqrt{2\beta}+\|A\|_2} / \left(2\sqrt{\kappa(\hat{H})} \right) \right) \right\rceil$, where $\hat{H} = \left(\beta^{-1} AA^T + I\right)$, $c = \frac{\sqrt{\kappa(P^{-1}\hat{H})}-1}{\sqrt{\kappa(P^{-1}\hat{H})}+1}$, $\kappa(P^{-1}\hat{H})$ is the condition number of $P^{-1}\hat{H}$, P is a nonsingular preconditioner.

Cont'd

Cont'd

- For CG, i.e., $P = I$ in PCG, we have
$$N_0 = \left\lceil \log_c \left(\frac{\sqrt{2\beta}}{\sqrt{2\beta}+\|A\|_2} / \left(2\sqrt{\kappa(\hat{H})} \right) \right) \right\rceil \text{ and } c = \frac{\sqrt{\kappa(\hat{H})}-1}{\sqrt{\kappa(\hat{H})}+1}.$$

Cont'd

- For CG, i.e., $P = I$ in PCG, we have
  $N_0 = \left\lceil \log_c \left(\frac{\sqrt{2\beta}}{\sqrt{2\beta}+\|A\|_2} / \left(2\sqrt{\kappa(\hat{H})} \right) \right) \right\rceil$ and $c = \frac{\sqrt{\kappa(\hat{H})}-1}{\sqrt{\kappa(\hat{H})}+1}$.
- For Jacobi, Gauss-Seidel and *SOR*: N_0 is about $\left\lceil \log_{\rho(T)} \left(\frac{\sqrt{2\beta}}{\sqrt{2\beta}+\|A\|_2} \right) \right\rceil$, where T is the corresponding iterative matrix of one of these three linear algebra solvers. If $\|\hat{H}T\hat{H}^{-1}\|_2$ is smaller than 1, then we can obtain
  $N_0 \leq \left\lceil \log_{\|\hat{H}T\hat{H}^{-1}\|_2} \left(\frac{\sqrt{2\beta}}{\sqrt{2\beta}+\|A\|_2} \right) \right\rceil$.

Outline

Model

We consider the following distributed optimal control problem with parabolic equation constraints:

$$\min_{u\in\mathcal{C},y\in L^2(Q)} \quad \frac{1}{2}\iint_Q |y-y_d|^2 dxdt + \frac{\alpha}{2}\iint_{\mathcal{O}} |u|^2 dxdt$$

$$\text{s.t.} \quad \begin{cases} \dfrac{\partial y}{\partial t} - \nu\Delta y + a_0 y = u\chi_{\mathcal{O}}, & \text{in} \quad \Omega\times(0,T), \\ y = 0, & \text{on} \quad \Gamma\times(0,T), \\ y(0) = \varphi. \end{cases}$$

- $Q=\Omega\times(0,T)$, $\mathcal{O}=\omega\times(0,T)$, $0<T<+\infty$, ω is an open subset of domain $\Omega(\subset\mathbb{R}^d, d\geq 1)$, $\Gamma=\partial\Omega$.
- $\chi_{\mathcal{O}}$ is the characteristic function of set $\mathcal{O}$. The target function y_d and the initial value φ are given in $Y:=L^2(Q)$.
- The admissible set $\mathcal{C}$:
$$\mathcal{C}=\{v|v\in L^\infty(\mathcal{O}), a\leq v(x;t)\leq b \text{ a.e. in } \mathcal{O}\}\subset U:=L^2(\mathcal{O}),$$
where a and b are given constants.
- The constant $\alpha>0$ is a regularization parameter.
- The coefficients $a_0(\geq 0)\in L^\infty(Q)$ and ν is a positive constant.

Cont'd

Cont'd

- This old problem was introduced by Lions in 1971 and it finds applications in many areas such as physics, chemistry, engineering, medicine, finance, etc..

Cont'd

- This old problem was introduced by Lions in 1971 and it finds applications in many areas such as physics, chemistry, engineering, medicine, finance, etc..
- More results in theoretical analysis, discretization, numerical algorithms, and applications perspectives can be found in the literature,

Cont'd

- This old problem was introduced by Lions in 1971 and it finds applications in many areas such as physics, chemistry, engineering, medicine, finance, etc..
- More results in theoretical analysis, discretization, numerical algorithms, and applications perspectives can be found in the literature, e.g., Lions and Glowinski 1994, Tröltzsch 2010; and Rösch 2004, Lions, Glowinski and He 2008, Borzì and Schulz 2009; Hinze, Pinnau, Ulbrich and Ulbrich 2009.

Unconstrained Case

Unconstrained Case

- If $a = -\infty$ and $b = +\infty$, i.e. $\mathcal{C} = L^2(\mathcal{O})$, then the model reduces to the unconstrained optimal control problem whose optimality condition is a large-scale linear saddle point system.

Unconstrained Case

- If $a = -\infty$ and $b = +\infty$, i.e. $\mathcal{C} = L^2(\mathcal{O})$, then the model reduces to the unconstrained optimal control problem whose optimality condition is a large-scale linear saddle point system.
- Numerical methods include

Unconstrained Case

- If $a = -\infty$ and $b = +\infty$, i.e. $\mathcal{C} = L^2(\mathcal{O})$, then the model reduces to the unconstrained optimal control problem whose optimality condition is a large-scale linear saddle point system.
- Numerical methods include
  - Black-box approach(the solution of the state equation is embedded into an optimization loop):

Unconstrained Case

- If $a = -\infty$ and $b = +\infty$, i.e. $\mathcal{C} = L^2(\mathcal{O})$, then the model reduces to the unconstrained optimal control problem whose optimality condition is a large-scale linear saddle point system.
- Numerical methods include
  - Black-box approach(the solution of the state equation is embedded into an optimization loop):
    - Conjugate gradient(CG) method(J. L. Lions and R. Glowinski 1994);
    - Newton method (R. Becker, D. Meidner and B. Vexler 2007).

Unconstrained Case

- If $a = -\infty$ and $b = +\infty$, i.e. $\mathcal{C} = L^2(\mathcal{O})$, then the model reduces to the unconstrained optimal control problem whose optimality condition is a large-scale linear saddle point system.
- Numerical methods include
  - Black-box approach(the solution of the state equation is embedded into an optimization loop):
    - Conjugate gradient(CG) method(J. L. Lions and R. Glowinski 1994);
    - Newton method (R. Becker, D. Meidner and B. Vexler 2007).
  - “One-shot" approach which solves the whole linear optimality systems (T. P. Mathew , M. Sarkis and C. E. Schaerer 2010, E. Mcdonald 2016, J. W. Pearson, M. Stoll and A. J. Wathen 2012).

Constrained Case

Constrained Case

- The control constraint results in nonsmoothness and its optimality condition is a nonlinear saddle point system for which numerical methods for unconstrained cases are not applicable.

Constrained Case

- The control constraint results in nonsmoothness and its optimality condition is a nonlinear saddle point system for which numerical methods for unconstrained cases are not applicable.
- Existing methods: interior point method (e.g., Pearson and Gondzio 2017), active-set strategies which are in general based on the semismooth Newton method (e.g., Ulbrich 2003; Hinze, Pinnau, Ulbrich and Ulbrich 2009).

Constrained Case

- The control constraint results in nonsmoothness and its optimality condition is a nonlinear saddle point system for which numerical methods for unconstrained cases are not applicable.
- Existing methods: interior point method (e.g., Pearson and Gondzio 2017), active-set strategies which are in general based on the semismooth Newton method (e.g., Ulbrich 2003; Hinze, Pinnau, Ulbrich and Ulbrich 2009).
- Semismooth Newton type methods converge superlinearly and obtain high-precision solutions.

Numerical Experiments

Set-up:

- Stopping criterion:

$$\max\{p_s, d_s\} \leq tol,$$

where the primal residual p_s and dual residual d_s are defined as:

$$p_s = \|z^k - z^{k-1}\|/\|z^{k-1}\|, \text{ and } d_s = \|u^k - z^k\|/\max\{\|u^{k-1}\|, \|z^{k-1}\|\}.$$

- To verify the accuracy of the numerical solution, we define the relative distance "RelDis" and the objective function value "Obj" as

$$RelDis := \frac{\|y - y_d\|^2_{L^2(Q)}}{\|y_d\|^2_{L^2(Q)}}, \quad Obj := \frac{1}{2}\|y - y_d\|^2_{L^2(Q)} + \frac{\alpha}{2}\|u\|^2_{L^2(\mathcal{O})}.$$

- Initial values: $u = 0, z = 0$ and $\lambda = 0$.
- We set $\beta = 3$ in all experiments, which implies

$$\sigma = \frac{\sqrt{2}}{\sqrt{2} + \sqrt{3}} \approx 0.414.$$

- We solve the linear subproblems with the CG method.

Example 1 – With Known Exact Solution

- Setting $\Omega = (0,1) \times (0,1)$, $\omega = \Omega$ and $T = 1$, we consider the problem:

$$\min_{u \in \mathcal{C}, y \in L^2(Q)} \quad \frac{1}{2} \iint_Q |y - y_d|^2 dxdt + \frac{\alpha}{2} \iint_Q |u|^2 dxdt$$

$$\text{s.t.} \quad \begin{cases} \dfrac{\partial y}{\partial t} - \Delta y = f + u, & \text{in} \quad \Omega \times (0,T), \\ y = 0, & \text{on} \quad \Gamma \times (0,T), \\ y(0) = \varphi. \end{cases}$$

- The function $f \in L^2(Q)$ is a source term which is somehow artificial to help us construct the exact solution.
- We set $a = -0.5$ and $b = 0.5$ in the control constraint, and the regularization parameter $\alpha = 10^{-5}$.

Example 1 – Cont'd

We let

$$\begin{aligned} y &= (1-t)\sin \pi x \sin \pi y, \\ p &= \alpha(1-t)\sin 2\pi x \sin 2\pi y, \\ u &= \min(a, \max(b, -p/\alpha)), \end{aligned}$$

and set

$$f = -u + \frac{dy}{dt} - \Delta y, \quad y_d = y + \frac{dp}{dt} + \Delta p,$$

then it is easy to verify that $(u^*, y^*) = (u, y)$ is the optimal solution.

Example 1 – Cont'd

Table: Numerical comparison of InADMM and ADMM_{1e-k} in Example 1.

Discretization	Algorithm	ADMM_{Iter}	Mean/Max CG	Time(s)	RelDis	Obj
$(h = \Delta t = \frac{1}{16})$	ADMM_{1e-10}	24	87.67/116	3.0058	1.5348×10^{-6}	4.5101×10^{-7}
	ADMM_{1e-6}	24	38.79/48	1.6118	1.5347×10^{-6}	4.5101×10^{-7}
	ADMM_{1e-2}	$\sim$	$\sim$	$\sim$	$\sim$	$\sim$
	InADMM	26	5.61/7	0.3614	1.5353×10^{-6}	4.5101×10^{-7}
$(h = \Delta t = \frac{1}{32})$	ADMM_{1e-10}	21	61.71/83	17.4940	7.5987×10^{-7}	3.6825×10^{-7}
	ADMM_{1e-6}	21	28.47/49	8.5941	7.5986×10^{-7}	3.6825×10^{-7}
	ADMM_{1e-2}	$\sim$	$\sim$	$\sim$	$\sim$	$\sim$
	InADMM	24	5.88/8	1.9306	7.5954×10^{-7}	3.6823×10^{-7}
$(h = \Delta t = \frac{1}{64})$	ADMM_{1e-10}	19	60.20/94	196.6824	6.7055×10^{-7}	3.5036×10^{-7}
	ADMM_{1e-6}	19	27.47/48	93.6593	6.7055×10^{-7}	3.5036×10^{-7}
	ADMM_{1e-2}	$\sim$	$\sim$	$\sim$	$\sim$	$\sim$
	InADMM	22	6.00/7	20.8675	6.7075×10^{-7}	3.5035×10^{-7}
$(h = \Delta t = \frac{1}{128})$	ADMM_{1e-10}	19	59.10/93	3372.3055	6.5295×10^{-7}	3.4473×10^{-7}
	ADMM_{1e-6}	19	27.15/48	1653.1018	6.5295×10^{-7}	3.4473×10^{-7}
	ADMM_{1e-2}	$\sim$	$\sim$	$\sim$	$\sim$	$\sim$
	InADMM	20	6.20/8	307.0674	6.5294×10^{-7}	3.4473×10^{-7}
$(h = \Delta t = \frac{1}{256})$	ADMM_{1e-10}	19	58.30/93	47106.7696	6.4876×10^{-7}	3.4260×10^{-7}
	ADMM_{1e-6}	19	26.95/48	20801.5575	6.4876×10^{-7}	3.4260×10^{-7}
	ADMM_{1e-2}	$\sim$	$\sim$	$\sim$	$\sim$	$\sim$
	InADMM	20	6.05/8	3839.6788	6.4877×10^{-7}	3.4260×10^{-7}

Example 1 – Cont'd

Table: Accuracy results with $\beta = 3$ and $tol = 10^{-4}$ in Example 1.

error	$h = \Delta t = 1/16$	$h = \Delta t = 1/32$	$h = \Delta t = 1/64$	$h = \Delta t = 1/128$	$h = \Delta t = 1/256$
$\|u - u^*\|$	8.0151×10^{-3}	2.0965×10^{-3}	5.3381×10^{-4}	1.3893×10^{-4}	4.3774×10^{-5}
$\|y - y^*\|$	2.6319×10^{-3}	6.8713×10^{-4}	1.7508×10^{-4}	4.4163×10^{-5}	1.1077×10^{-5}

Example 1 – Cont'd

Table: Accuracy results with $\beta = 3$ and $tol = 10^{-4}$ in Example 1.

error	$h = \Delta t = 1/16$	$h = \Delta t = 1/32$	$h = \Delta t = 1/64$	$h = \Delta t = 1/128$	$h = \Delta t = 1/256$
$\|u - u^*\|$	8.0151×10^{-3}	2.0965×10^{-3}	5.3381×10^{-4}	1.3893×10^{-4}	4.3774×10^{-5}
$\|y - y^*\|$	2.6319×10^{-3}	6.8713×10^{-4}	1.7508×10^{-4}	4.4163×10^{-5}	1.1077×10^{-5}

Some immediate observations:

Example 1 – Cont'd

Table: Accuracy results with $\beta = 3$ and $tol = 10^{-4}$ in Example 1.

error	$h = \Delta t = 1/16$	$h = \Delta t = 1/32$	$h = \Delta t = 1/64$	$h = \Delta t = 1/128$	$h = \Delta t = 1/256$
$\|u - u^*\|$	8.0151×10^{-3}	2.0965×10^{-3}	5.3381×10^{-4}	1.3893×10^{-4}	4.3774×10^{-5}
$\|y - y^*\|$	2.6319×10^{-3}	6.8713×10^{-4}	1.7508×10^{-4}	4.4163×10^{-5}	1.1077×10^{-5}

Some immediate observations:

- Solving the subproblems exactly or too accurately is not necessary at all.

Example 1 – Cont'd

Table: Accuracy results with $\beta = 3$ and $tol = 10^{-4}$ in Example 1.

error	$h = \Delta t = 1/16$	$h = \Delta t = 1/32$	$h = \Delta t = 1/64$	$h = \Delta t = 1/128$	$h = \Delta t = 1/256$
$\|u - u^*\|$	8.0151×10^{-3}	2.0965×10^{-3}	5.3381×10^{-4}	1.3893×10^{-4}	4.3774×10^{-5}
$\|y - y^*\|$	2.6319×10^{-3}	6.8713×10^{-4}	1.7508×10^{-4}	4.4163×10^{-5}	1.1077×10^{-5}

Some immediate observations:

- Solving the subproblems exactly or too accurately is not necessary at all.
- Our strategy significantly saves computation, especially for the fine discretization case.

Example 1 – Cont'd

Table: Accuracy results with $\beta = 3$ and $tol = 10^{-4}$ in Example 1.

error	$h = \Delta t = 1/16$	$h = \Delta t = 1/32$	$h = \Delta t = 1/64$	$h = \Delta t = 1/128$	$h = \Delta t = 1/256$
$\|u - u^*\|$	8.0151×10^{-3}	2.0965×10^{-3}	5.3381×10^{-4}	1.3893×10^{-4}	4.3774×10^{-5}
$\|y - y^*\|$	2.6319×10^{-3}	6.8713×10^{-4}	1.7508×10^{-4}	4.4163×10^{-5}	1.1077×10^{-5}

Some immediate observations:

- Solving the subproblems exactly or too accurately is not necessary at all.
- Our strategy significantly saves computation, especially for the fine discretization case.
- Convergence is fast, and the accuracy is indeed not low.

Example 1 – Cont'd

Table: Accuracy results with $\beta = 3$ and $tol = 10^{-4}$ in Example 1.

error	$h = \Delta t = 1/16$	$h = \Delta t = 1/32$	$h = \Delta t = 1/64$	$h = \Delta t = 1/128$	$h = \Delta t = 1/256$
$\|u - u^*\|$	8.0151×10^{-3}	2.0965×10^{-3}	5.3381×10^{-4}	1.3893×10^{-4}	4.3774×10^{-5}
$\|y - y^*\|$	2.6319×10^{-3}	6.8713×10^{-4}	1.7508×10^{-4}	4.4163×10^{-5}	1.1077×10^{-5}

Some immediate observations:

- Solving the subproblems exactly or too accurately is not necessary at all.
- Our strategy significantly saves computation, especially for the fine discretization case.
- Convergence is fast, and the accuracy is indeed not low.
- Iterations are robust with respect to the mesh size.

Example 2 – Unknown Exact Solutions

- We consider

$$\min_{u\in\mathcal{C},y\in L^2(Q)} \quad \frac{1}{2}\iint_Q |y-y_d|^2 dxdt + \frac{\alpha}{2}\iint_{\mathcal{O}} |u|^2 dxdt$$

$$\text{s.t.} \quad \begin{cases} \dfrac{\partial y}{\partial t} - \Delta y + y = u\chi_{\mathcal{O}}, & \text{in} \quad \Omega\times(0,T), \\ y = 0, & \text{on} \quad \Gamma\times(0,T), \\ y(0) = \sin 4x \sin 4y. \end{cases}$$

- $\Omega = (0,1)\times(0,1), \omega = (0,0.25)\times(0,0.25) \subseteq \Omega$, $T = 1$;
- The constants $a = -300$ and $b = 300$ in the control constraints, and the regularization parameter $\alpha = 10^{-6}$;
- The target function y_d is given by $y_d = e^t \sin 4x \sin 4y$.

Example 2 – Cont'd

Table: Numerical comparison with $\beta = 3$ and $tol = 10^{-3}$ in Example 2.

Mesh	Algorithm	ADMM_{Iter}	Mean/Max CG	Time(s)	RelDis	Obj
$\frac{1}{16}$	ADMM_{1e-10}	27	51.33/83	2.3728	0.9300	0.3310
	ADMM_{1e-6}	27	30.96/52	1.3587	0.9300	0.3310
	ADMM_{1e-2}	27	11.85/23	0.6171	0.9300	0.3310
	InADMM	28	3.07/4	0.2786	0.9300	0.3310
$\frac{1}{32}$	ADMM_{1e-10}	14	51.50/62	8.5682	0.9388	0.3726
	ADMM_{1e-6}	14	32.64/43	5.3468	0.9388	0.3726
	ADMM_{1e-2}	14	13.71/23	2.2433	0.9388	0.3726
	InADMM	17	3.35/4	0.8344	0.9388	0.3726
$\frac{1}{64}$	ADMM_{1e-10}	16	51.63/62	110.0527	0.9428	0.3812
	ADMM_{1e-6}	16	32.31/43	64.4313	0.9428	0.3812
	ADMM_{1e-2}	16	13.25/23	27.1500	0.9428	0.3812
	InADMM	18	3.39/4	8.2943	0.9428	0.3812
$\frac{1}{128}$	ADMM_{1e-10}	16	50.50/61	1834.3239	0.9455	0.3821
	ADMM_{1e-6}	16	31.81/42	1291.1113	0.9455	0.3821
	ADMM_{1e-2}	16	12.81/23	401.5557	0.9455	0.3821
	InADMM	18	3.33/4	129.3381	0.9455	0.3821
$\frac{1}{256}$	ADMM_{1e-10}	16	49.69/60	22540.1768	0.9470	0.3817
	ADMM_{1e-6}	16	31.25/41	14969.8387	0.9470	0.3817
	ADMM_{1e-2}	16	12.63/22	6281.9469	0.9470	0.3817
	InADMM	18	3.33/4	1609.7365	0.9470	0.3817

Outline

Conclusions

Conclusions

- Application of an “simple" algorithm to the large-scale scenario of an “simple" problem may not be simple at all.

Conclusions

- Application of an “simple" algorithm to the large-scale scenario of an “simple" problem may not be simple at all.
- Some serious mathematical issues may arise,

Conclusions

- Application of an “simple" algorithm to the large-scale scenario of an “simple" problem may not be simple at all.
- Some serious mathematical issues may arise, e.g., can convergence be guaranteed if an algorithm with theoretically known convergence is implemented inexactly but with an embedded internal iteration process?

Conclusions

- Application of an “simple" algorithm to the large-scale scenario of an “simple" problem may not be simple at all.
- Some serious mathematical issues may arise, e.g., can convergence be guaranteed if an algorithm with theoretically known convergence is implemented inexactly but with an embedded internal iteration process?
- ADMM (and operator splitting methods) can be used to solve some optimal control problems efficiently

Conclusions

- Application of an “simple" algorithm to the large-scale scenario of an “simple" problem may not be simple at all.
- Some serious mathematical issues may arise, e.g., can convergence be guaranteed if an algorithm with theoretically known convergence is implemented inexactly but with an embedded internal iteration process?
- ADMM (and operator splitting methods) can be used to solve some optimal control problems efficiently but only in a smart way (decoupling main PDE-related constraints and other simple constraints).

Conclusions

- Application of an “simple" algorithm to the large-scale scenario of an “simple" problem may not be simple at all.
- Some serious mathematical issues may arise, e.g., can convergence be guaranteed if an algorithm with theoretically known convergence is implemented inexactly but with an embedded internal iteration process?
- ADMM (and operator splitting methods) can be used to solve some optimal control problems efficiently but only in a smart way (decoupling main PDE-related constraints and other simple constraints).
- Implementability and accuracy should be considered simultaneously in algorithmic design.

Conclusions

- Application of an “simple" algorithm to the large-scale scenario of an “simple" problem may not be simple at all.
- Some serious mathematical issues may arise, e.g., can convergence be guaranteed if an algorithm with theoretically known convergence is implemented inexactly but with an embedded internal iteration process?
- ADMM (and operator splitting methods) can be used to solve some optimal control problems efficiently but only in a smart way (decoupling main PDE-related constraints and other simple constraints).
- Implementability and accuracy should be considered simultaneously in algorithmic design.
- Machine Learning (Deep Learning) has redefined “solvable" problems; it is very influential to implementing operator splitting techniques with a balance between “solvable subproblems" and “rigorous theory".