Model-based X-ray CT image reconstruction using variable splitting methods with ordered subsets

Hung Nien (粘紘)
Advisor: Prof. Jeffrey A. Fessler

University of Michigan, Ann Arbor

May 19, 2014
Acknowledgements

- Supported in part by NIH grant R01-HL-098686
- Supported in part by an equipment donation from Intel Corp.
- Sinogram data are provided by GE Healthcare
Outline of the thesis

1. Introduction
2. Background of X-ray CT and its reconstruction
3. Fast X-ray CT image reconstruction using VS methods with OS
   [HN & J A Fessler, Fully 3D, 2013]
   [HN & J A Fessler, SPIE MI, 2014]
   [HN & J A Fessler, CT Meeting, 2014]
4. Blind gain correction for X-ray CT image reconstruction
   [HN & J A Fessler, SPIE MI, 2013]
5. Model-based light field reconstruction
6. Conclusion and future work
Outline of the thesis

1. Introduction
2. Background of X-ray CT and its reconstruction
3. Fast X-ray CT image reconstruction using VS methods with OS
   [HN & J A Fessler, Fully 3D, 2013]
   [HN & J A Fessler, SPIE MI, 2014]
   [HN & J A Fessler, CT Meeting, 2014]
4. Blind gain correction for X-ray CT image reconstruction
   [HN & J A Fessler, SPIE MI, 2013]
5. Model-based light field reconstruction
6. Conclusion and future work
Outline of the thesis

1. Introduction
2. Background of X-ray CT and its reconstruction
   - Fast X-ray CT image reconstruction using VS methods with OS
     [HN & J A Fessler, Fully 3D, 2013]
     [HN & J A Fessler, SPIE MI, 2014]
     [HN & J A Fessler, CT Meeting, 2014]
   - Blind gain correction for X-ray CT image reconstruction
     [HN & J A Fessler, SPIE MI, 2013]
3. Model-based light field reconstruction
4. Conclusion and future work
Outline of the thesis

1. Introduction
2. Background of X-ray CT and its reconstruction
   - Fast X-ray CT image reconstruction using VS methods with OS
     - [HN & J A Fessler, Fully 3D, 2013]
     - [HN & J A Fessler, SPIE MI, 2014]
     - [HN & J A Fessler, CT Meeting, 2014]
   - Blind gain correction for X-ray CT image reconstruction
     - [HN & J A Fessler, SPIE MI, 2013]
3. Model-based light field reconstruction
4. Conclusion and future work
Outline of the thesis

1. Introduction
2. Background of X-ray CT and its reconstruction
   - Fast X-ray CT image reconstruction using VS methods with OS
     [HN & J A Fessler, Fully 3D, 2013]
     [HN & J A Fessler, SPIE MI, 2014]
     [HN & J A Fessler, CT Meeting, 2014]
3. Blind gain correction for X-ray CT image reconstruction
   [HN & J A Fessler, SPIE MI, 2013]
4. Model-based light field reconstruction
5. Conclusion and future work
Outline of the thesis

1. Introduction
2. Background of X-ray CT and its reconstruction
   - Fast X-ray CT image reconstruction using VS methods with OS
     - [HN & J A Fessler, Fully 3D, 2013]
     - [HN & J A Fessler, SPIE MI, 2014]
     - [HN & J A Fessler, CT Meeting, 2014]
3. Blind gain correction for X-ray CT image reconstruction
   - [HN & J A Fessler, SPIE MI, 2013]
4. Model-based light field reconstruction
5. Conclusion and future work
Outline of the thesis

1. Introduction
2. Background of X-ray CT and its reconstruction
3. Fast X-ray CT image reconstruction using VS methods with OS
   [HN & J A Fessler, Fully 3D, 2013]
   [HN & J A Fessler, SPIE MI, 2014]
   [HN & J A Fessler, CT Meeting, 2014]
4. Blind gain correction for X-ray CT image reconstruction
   [HN & J A Fessler, SPIE MI, 2013]
5. Model-based light field reconstruction
6. Conclusion and future work
Outline of the thesis

1. Introduction
2. Background of X-ray CT and its reconstruction
3. Fast X-ray CT image reconstruction using VS methods with OS
   [HN & J A Fessler, Fully 3D, 2013]
   [HN & J A Fessler, SPIE MI, 2014]
   [HN & J A Fessler, CT Meeting, 2014]
4. Blind gain correction for X-ray CT image reconstruction
   [HN & J A Fessler, SPIE MI, 2013]
5. Model-based light field reconstruction
6. Conclusion and future work
Outline of the thesis

1. Introduction
2. Background of X-ray CT and its reconstruction
3. Fast X-ray CT image reconstruction using VS methods with OS
   [HN & J A Fessler, Fully 3D, 2013]
   [HN & J A Fessler, SPIE MI, 2014]
   [HN & J A Fessler, CT Meeting, 2014]
4. Blind gain correction for X-ray CT image reconstruction
   [HN & J A Fessler, SPIE MI, 2013]
5. Model-based light field reconstruction
6. Conclusion and future work
1 Introduction
2 Background of X-ray CT and its reconstruction
3 Fast X-ray CT image reconstruction using VS methods with OS
   [HN & J A Fessler, Fully 3D, 2013]
   [HN & J A Fessler, SPIE MI, 2014]
   [HN & J A Fessler, CT Meeting, 2014]
4 Blind gain correction for X-ray CT image reconstruction
   [HN & J A Fessler, SPIE MI, 2013]
5 Model-based light field reconstruction
6 Conclusion and future work
1 Background
   - Model-based CT reconstruction
   - Fast (2D) CT reconstruction using ADMM

2 OS-LALM: a splitting-based OS algorithm for PWLS problems
   - Linearized AL method with OS acceleration
   - Deterministic downward continuation approach
   - Low-memory OS-LALM with additional variable splits

3 Experimental results
   - Low-dose CT with edge-preserving regularizers
   - Sparse-view CT with TV-like regularizers

4 Conclusion and future work
Outline

1 Background
   - Model-based CT reconstruction
   - Fast (2D) CT reconstruction using ADMM

2 OS-LALM: a splitting-based OS algorithm for PWLS problems
   - Linearized AL method with OS acceleration
   - Deterministic downward continuation approach
   - Low-memory OS-LALM with additional variable splits

3 Experimental results
   - Low-dose CT with edge-preserving regularizers
   - Sparse-view CT with TV-like regularizers

4 Conclusion and future work
Outline

1 Background
   - Model-based CT reconstruction
   - Fast (2D) CT reconstruction using ADMM

2 OS-LALM: a splitting-based OS algorithm for PWLS problems
   - Linearized AL method with OS acceleration
   - Deterministic downward continuation approach
   - Low-memory OS-LALM with additional variable splits

3 Experimental results
   - Low-dose CT with edge-preserving regularizers
   - Sparse-view CT with TV-like regularizers

4 Conclusion and future work
What is computed tomography?

Projection radiography

An imaging technique that uses X-rays to view the internal structure of a non-uniformly composed and opaque object such as the human body.

Computed tomography

An imaging technique that combines a series of X-ray projections taken from many different angles and computer processing (i.e., reconstruction methods) to create cross-sectional images of the bones and soft tissues inside the human body.
What is computed tomography?

Projection radiography

An imaging technique that uses X-rays to view the internal structure of a non-uniformly composed and opaque object such as the human body.
What is computed tomography?

Projection radiography
An imaging technique that uses X-rays to view the internal structure of a non-uniformly composed and opaque object such as the human body.

Computed tomography
An imaging technique that combines a series of X-ray projections taken from many different angles and computer processing (i.e., reconstruction methods) to create cross-sectional images of the bones and soft tissues inside to the human body.
What is computed tomography?

Figure: Chest X-ray image (left) and cross-sectional image of abdomen (right).
Background

Model-based CT reconstruction

Basics of X-ray computed tomography
Basics of X-ray computed tomography
Basics of X-ray computed tomography
Basics of X-ray computed tomography
Basics of X-ray computed tomography

\[ Y_i \propto N_0 \exp(-y_i) \]
Basics of X-ray computed tomography

\[
y = Ax + \epsilon
\]
Basics of X-ray computed tomography

\[ y = A x + \varepsilon \]
Basics of X-ray computed tomography

\[ y = A x + \varepsilon \]
Basics of X-ray computed tomography

\[ \mathbf{y} = \mathbf{A} \mathbf{x} + \mathbf{\epsilon} \]
Image reconstruction methods

Non-iterative methods

Iterative methods
Image reconstruction methods

Non-iterative methods

- Direct Fourier reconstruction

Iterative methods
Image reconstruction methods

Non-iterative methods
- Direct Fourier reconstruction
- Filter-backproject (FBP) method

Iterative methods
Image reconstruction methods

Non-iterative methods
- Direct Fourier reconstruction
- Filter-backproject (FBP) method
- **Very fast (seconds) but prone to noise (medium/high dose)**

Iterative methods
Image reconstruction methods

Non-iterative methods
- Direct Fourier reconstruction
- Filter-backproject (FBP) method
- Very fast (seconds) but prone to noise (medium/high dose)

Iterative methods
- Maximum a posteriori (MAP) formulation
Image reconstruction methods

Non-iterative methods
- Direct Fourier reconstruction
- Filter-backproject (FBP) method
- Very fast (seconds) but prone to noise (medium/high dose)

Iterative methods
- Maximum a posteriori (MAP) formulation
- Penalized weighted least-squares (PWLS) formulation [TS\(^+\)07]

\[ \hat{x} \in \arg \min_{x \in \Omega} \left\{ \psi(x) \triangleq \frac{1}{2} \| y - Ax \|_W^2 + R(x) \right\} \]
Image reconstruction methods

Figure: Dose reduction: FBP (left), ASiR (middle), and MBIR (right).
Image reconstruction methods

Non-iterative methods

- Direct Fourier reconstruction
- Filter-backproject (FBP) method
- Very fast (seconds) but prone to noise (medium/high dose)

Iterative methods

- Maximum a posteriori (MAP) formulation
- Penalized weighted least-squares (PWLS) formulation [TS$^+$07]

$$\hat{x} \in \arg \min_{x \in \Omega} \left\{ \Psi(x) \triangleq \frac{1}{2} \| y - Ax \|_W^2 + R(x) \right\}$$
Image reconstruction methods

Large problem size

- $\mathbf{x}$: $512 \times 512 \times 100 \approx 3 \cdot 10^7$ unknown image volume
- $\mathbf{y}$: $888 \times 32 \times 7000 \approx 2 \cdot 10^8$ measured noisy sinogram
- $\mathbf{A}$: $(3 \cdot 10^7) \times (2 \cdot 10^8)$ system matrix
- $\mathbf{A}$ is sparse but still too large to store
- Projection $\mathbf{A}\mathbf{x}$ and back-projection $\mathbf{A}'\mathbf{r}$ operations computed on the fly
- Computing gradient $\nabla \psi(\mathbf{x}) = \mathbf{A}'\mathbf{W}(\mathbf{A}\mathbf{x} - \mathbf{y}) + \nabla R(\mathbf{x})$ requires projection and back-projection operations that dominate computation
Large problem size

- \( \mathbf{x} : 512 \times 512 \times 100 \approx 3 \cdot 10^7 \) unknown image volume
- \( \mathbf{y} : 888 \times 32 \times 7000 \approx 2 \cdot 10^8 \) measured noisy sinogram
- \( \mathbf{A} : (3 \cdot 10^7) \times (2 \cdot 10^8) \) system matrix
- \( \mathbf{A} \) is sparse but still too large to store
- Projection \( \mathbf{Ax} \) and back-projection \( \mathbf{A'}r \) operations computed on the fly
- Computing gradient \( \nabla \psi(\mathbf{x}) = \mathbf{A'}W (\mathbf{Ax} - \mathbf{y}) + \nabla R(\mathbf{x}) \) requires projection and back-projection operations that dominate computation

Enormous dynamic range of transmission data

- The dynamic range of weighting \( \mathbf{W} \) is huge
- \( \mathbf{A'}WA \) is highly shift-variant, and the problem is very ill-conditioned
Image reconstruction methods

Non-iterative methods
- Direct Fourier reconstruction
- Filter-backproject (FBP) method
- Very fast (seconds) but prone to noise (medium/high dose)

Iterative methods
- Maximum a posteriori (MAP) formulation
- Penalized weighted least-squares (PWLS) formulation \([\text{TS}^+07]\)
  \[
  \hat{x} \in \arg \min_{x \in \Omega} \left\{ \Psi(x) \triangleq \frac{1}{2} \|y - Ax\|_W^2 + R(x) \right\}
  \]
- Very slow (hours) but noise robust (low dose)
Image reconstruction methods

Non-iterative methods
- Direct Fourier reconstruction
- Filter-backproject (FBP) method
- Very fast (seconds) but prone to noise (medium/high dose)

Fast iterative methods
- Maximum \textit{a posteriori} (MAP) formulation
- Penalized weighted least-squares (PWLS) formulation [TS$^+$07]
  \[
  \hat{x} \in \arg\min_{x \in \Omega} \left\{ \Psi(x) \triangleq \frac{1}{2} \|y - Ax\|^2_W + R(x) \right\}
  \]
- Fast (minutes) and noise robust (low dose)
Outline

1 Background
   - Model-based CT reconstruction
   - Fast (2D) CT reconstruction using ADMM

2 OS-LALM: a splitting-based OS algorithm for PWLS problems
   - Linearized AL method with OS acceleration
   - Deterministic downward continuation approach
   - Low-memory OS-LALM with additional variable splits

3 Experimental results
   - Low-dose CT with edge-preserving regularizers
   - Sparse-view CT with TV-like regularizers

4 Conclusion and future work
CT reconstruction using split Bregman method

Equivalent formulation and split Bregman method

\[
\hat{x} \in \text{arg min}_x \left\{ \frac{1}{2} \| y - Ax \|_2^2 W + \Phi(Cx) \right\}
\]

Corresponding (scaled) augmented Lagrangian:

\[
L_A(x, v, e; \eta) = \frac{1}{2} \| y - Ax \|_2^2 W + \Phi(v) + \frac{\eta}{2} \| Cx - v - e \|_2^2
\]
CT reconstruction using split Bregman method

Equivalent formulation and split Bregman method

- PWLS CT reconstruction: \( \hat{x} \in \arg\min_x \left\{ \frac{1}{2} \| y - Ax \|_W^2 + \Phi(Cx) \right\} \)
CT reconstruction using split Bregman method

Equivalent formulation and split Bregman method

- PWLS CT reconstruction: \( \hat{x} \in \text{arg min}_{x} \left\{ \frac{1}{2} \| y - Ax \|^2_W + \Phi(Cx) \right\} \)
- Equivalent formulation [GO09]:
  \[
  (\hat{x}, \hat{v}) \in \text{arg min}_{x,v} \left\{ \frac{1}{2} \| y - Ax \|^2_W + \Phi(v) \right\} \quad \text{s.t.} \quad v = Cx
  \]
CT reconstruction using split Bregman method

Equivalent formulation and split Bregman method

- PWLS CT reconstruction: \( \hat{x} \in \arg \min_x \left\{ \frac{1}{2} \| y - Ax \|_W^2 + \Phi(Cx) \right\} \)

- Equivalent formulation [GO09]:
  
  \[
  (\hat{x}, \hat{v}) \in \arg \min_{x, v} \left\{ \frac{1}{2} \| y - Ax \|_W^2 + \Phi(v) \right\} \quad \text{s.t.} \quad v = Cx
  \]

- Corresponding (scaled) augmented Lagrangian:
  
  \[
  \mathcal{L}_A(x, v, e; \eta) \triangleq \frac{1}{2} \| y - Ax \|_W^2 + \Phi(v) + \frac{\eta}{2} \| Cx - v - e \|_2^2
  \]
CT reconstruction using split Bregman method (cont’d)

- Split Bregman iterates [GO09]:
  \[
  \begin{align*}
  &x^{(k+1)} \in \arg \min_x \left\{ \frac{1}{2} \| y - Ax \|_W^2 + \frac{\eta}{2} \| Cx - v^{(k)} - e^{(k)} \|_2^2 \right\} \\
  &v^{(k+1)} \in \arg \min_v \left\{ \Phi(v) + \frac{\eta}{2} \| Cx^{(k+1)} - v - e^{(k)} \|_2^2 \right\} \\
  &e^{(k+1)} = e^{(k)} - Cx^{(k+1)} + v^{(k+1)}
  \end{align*}
  \]
CT reconstruction using split Bregman method

Equivalent formulation and split Bregman method (cont’d)

- Split Bregman iterates [GO09]:

\[
\begin{align*}
\mathbf{x}^{(k+1)} &\in \arg \min_{\mathbf{x}} \left\{ \frac{1}{2} \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_W^2 + \frac{\eta}{2} \|\mathbf{C}\mathbf{x} - \mathbf{v}^{(k)} - \mathbf{e}^{(k)}\|_2^2 \right\} \\
\mathbf{v}^{(k+1)} &\in \arg \min_{\mathbf{v}} \left\{ \Phi(\mathbf{v}) + \frac{\eta}{2} \|\mathbf{C}\mathbf{x}^{(k+1)} - \mathbf{v} - \mathbf{e}^{(k)}\|_2^2 \right\} \\
\mathbf{e}^{(k+1)} &= \mathbf{e}^{(k)} - \mathbf{C}\mathbf{x}^{(k+1)} + \mathbf{v}^{(k+1)}
\end{align*}
\]
Equivalent formulation and split Bregman method (cont’d)

- Split Bregman iterates [GO09]:

\[
\begin{align*}
  x^{(k+1)} &\in \arg \min_x \left\{ \frac{1}{2} \| y - Ax \|^2_W + \frac{\eta}{2} \| Cx - v^{(k)} - e^{(k)} \|^2_2 \right\} \\
  v^{(k+1)} &\in \arg \min_v \left\{ \Phi(v) + \frac{\eta}{2} \| Cx^{(k+1)} - v - e^{(k)} \|^2_2 \right\} \\
  e^{(k+1)} &= e^{(k)} - Cx^{(k+1)} + v^{(k+1)}
\end{align*}
\]
CT reconstruction using split Bregman method

Equivalent formulation and split Bregman method (cont’d)

- Split Bregman iterates [GO09]:

\[
\begin{align*}
  x^{(k+1)} & \in \arg \min_x \left\{ \frac{1}{2} \| y - Ax \|_W^2 + \frac{\eta}{2} \| Cx - v^{(k)} - e^{(k)} \|_2^2 \right\} \\
  v^{(k+1)} & \in \arg \min_v \left\{ \Phi(v) + \frac{\eta}{2} \| Cx^{(k+1)} - v - e^{(k)} \|_2^2 \right\} \\
  e^{(k+1)} & = e^{(k)} - Cx^{(k+1)} + v^{(k+1)}
\end{align*}
\]

Convergent with inexact updates [NF14a]

Slow x-update due to the highly shift-variant Hessian $A^TWA^T + \eta C^T C$
CT reconstruction using split Bregman method

Equivalent formulation and split Bregman method (cont’d)

- Split Bregman iterates [GO09]:
  \[
  \begin{align*}
  x^{(k+1)} &\in \arg\min_x \left\{ \frac{1}{2} \|y - Ax\|_W^2 + \frac{\eta}{2} \|Cx - v^{(k)} - e^{(k)}\|_2^2 \right\} \\
  v^{(k+1)} &\in \arg\min_v \left\{ \Phi(v) + \frac{\eta}{2} \|Cx^{(k+1)} - v - e^{(k)}\|_2^2 \right\} \\
  e^{(k+1)} &= e^{(k)} - Cx^{(k+1)} + v^{(k+1)}
  \end{align*}
  \]

- Convergent with inexact updates [NF14a]
CT reconstruction using split Bregman method

Equivalent formulation and split Bregman method (cont’d)

- Split Bregman iterates [GO09]:

\[
\begin{align*}
\mathbf{x}^{(k+1)} & \in \arg \min_{\mathbf{x}} \left\{ \frac{1}{2} \| \mathbf{y} - \mathbf{A} \mathbf{x} \|^2_W + \frac{\eta}{2} \| \mathbf{C} \mathbf{x} - \mathbf{v}^{(k)} - \mathbf{e}^{(k)} \|^2 \right\} \\
\mathbf{v}^{(k+1)} & \in \arg \min_{\mathbf{v}} \left\{ \Phi(\mathbf{v}) + \frac{\eta}{2} \| \mathbf{C} \mathbf{x}^{(k+1)} - \mathbf{v} - \mathbf{e}^{(k)} \|^2 \right\} \\
\mathbf{e}^{(k+1)} & = \mathbf{e}^{(k)} - \mathbf{C} \mathbf{x}^{(k+1)} + \mathbf{v}^{(k+1)}
\end{align*}
\]

- Convergent with inexact updates [NF14a]
- Slow \(x\)-update due to the highly shift-variant Hessian \(\mathbf{A}'\mathbf{W}\mathbf{A} + \eta\mathbf{C}'\mathbf{C}\)
Better conditioning with additional VS

The idea is ...
Better conditioning with additional VS

The idea is ...

- In 2D CT, $A'WA$ is highly shift-variant, but $A'A$ is not
Better conditioning with additional VS

The idea is ...

- In 2D CT, $A'WA$ is highly shift-variant, but $A'A$ is not.
- Replacing the weighted quadratic function in the $x$-update with an unweighted one removes most shift-variances of the Hessian.
Better conditioning with additional VS

The idea is ...

- In 2D CT, $A'WA$ is highly shift-variant, but $A'A$ is not
- Replacing the weighted quadratic function in the $x$-update with an unweighted one removes most shift-variances of the Hessian

Alternative formulation and ADMM

- Alternative formulation [RF12]:

$$ (\hat{x}, \hat{u}, \hat{v}) \in \arg \min_{x,u,v} \left\{ \frac{1}{2} \|y - u\|_W^2 + \Phi(v) \right\} \text{ s.t. } u = Ax, v = Cx $$
Better conditioning with additional VS

Alternative formulation and ADMM (cont’d)

- ADMM iterates [RF12]:

\[
\begin{align*}
  x^{(k+1)} &\in \arg \min_x \left\{ \frac{\rho}{2} \| A x - u^{(k)} - d^{(k)} \|_2^2 + \frac{\eta}{2} \| C x - v^{(k)} - e^{(k)} \|_2^2 \right\} \\
  u^{(k+1)} &\in \arg \min_u \left\{ \frac{1}{2} \| y - u \|_W^2 + \frac{\rho}{2} \| A x^{(k+1)} - u - d^{(k)} \|_2^2 \right\} \\
  v^{(k+1)} &\in \arg \min_v \left\{ \Phi(v) + \frac{\eta}{2} \| C x^{(k+1)} - v - e^{(k)} \|_2^2 \right\} \\
  d^{(k+1)} &= d^{(k)} - A x^{(k+1)} + u^{(k+1)} \\
  e^{(k+1)} &= e^{(k)} - C x^{(k+1)} + v^{(k+1)}
\end{align*}
\]
Alternative formulation and ADMM (cont’d)

- ADMM iterates [RF12]:

\[
\begin{align*}
\mathbf{x}^{(k+1)} & \in \arg \min_{\mathbf{x}} \left\{ \frac{\rho}{2} \| \mathbf{A}\mathbf{x} - \mathbf{u}^{(k)} - \mathbf{d}^{(k)} \|_2^2 + \frac{\eta}{2} \| \mathbf{C}\mathbf{x} - \mathbf{v}^{(k)} - \mathbf{e}^{(k)} \|_2^2 \right\} \\
\mathbf{u}^{(k+1)} & \in \arg \min_{\mathbf{u}} \left\{ \frac{1}{2} \| \mathbf{y} - \mathbf{u} \|_W^2 + \frac{\rho}{2} \| \mathbf{A}\mathbf{x}^{(k+1)} - \mathbf{u} - \mathbf{d}^{(k)} \|_2^2 \right\} \\
\mathbf{v}^{(k+1)} & \in \arg \min_{\mathbf{v}} \left\{ \Phi(\mathbf{v}) + \frac{\eta}{2} \| \mathbf{C}\mathbf{x}^{(k+1)} - \mathbf{v} - \mathbf{e}^{(k)} \|_2^2 \right\} \\
\mathbf{d}^{(k+1)} & = \mathbf{d}^{(k)} - \mathbf{A}\mathbf{x}^{(k+1)} + \mathbf{u}^{(k+1)} \\
\mathbf{e}^{(k+1)} & = \mathbf{e}^{(k)} - \mathbf{C}\mathbf{x}^{(k+1)} + \mathbf{v}^{(k+1)}
\end{align*}
\]
Better conditioning with additional VS

Alternative formulation and ADMM (cont’d)

- ADMM iterates [RF12]:

\[
\begin{align*}
\mathbf{x}^{(k+1)} &\in \arg \min_{\mathbf{x}} \left\{ \frac{\rho}{2} \| \mathbf{A} \mathbf{x} - \mathbf{u}^{(k)} - \mathbf{d}^{(k)} \|_2^2 + \frac{\eta}{2} \| \mathbf{C} \mathbf{x} - \mathbf{v}^{(k)} - \mathbf{e}^{(k)} \|_2^2 \right\} \\
\mathbf{u}^{(k+1)} &\in \arg \min_{\mathbf{u}} \left\{ \frac{1}{2} \| \mathbf{y} - \mathbf{u} \|_W^2 + \frac{\rho}{2} \| \mathbf{A} \mathbf{x}^{(k+1)} - \mathbf{u} - \mathbf{d}^{(k)} \|_2^2 \right\} \\
\mathbf{v}^{(k+1)} &\in \arg \min_{\mathbf{v}} \left\{ \Phi(\mathbf{v}) + \frac{\eta}{2} \| \mathbf{C} \mathbf{x}^{(k+1)} - \mathbf{v} - \mathbf{e}^{(k)} \|_2^2 \right\} \\
\mathbf{d}^{(k+1)} &= \mathbf{d}^{(k)} - \mathbf{A} \mathbf{x}^{(k+1)} + \mathbf{u}^{(k+1)} \\
\mathbf{e}^{(k+1)} &= \mathbf{e}^{(k)} - \mathbf{C} \mathbf{x}^{(k+1)} + \mathbf{v}^{(k+1)}
\end{align*}
\]
Better conditioning with additional VS

Alternative formulation and ADMM (cont’d)

- ADMM iterates [RF12]:

\[
\begin{align*}
\mathbf{x}^{(k+1)} &\in \arg \min_{\mathbf{x}} \left\{ \frac{\rho}{2} \| \mathbf{Ax} - \mathbf{u}^{(k)} - \mathbf{d}^{(k)} \|_2^2 + \frac{\eta}{2} \| \mathbf{Cx} - \mathbf{v}^{(k)} - \mathbf{e}^{(k)} \|_2^2 \right\} \\
\mathbf{u}^{(k+1)} &\in \arg \min_{\mathbf{u}} \left\{ \frac{1}{2} \| \mathbf{y} - \mathbf{u} \|_W^2 + \frac{\rho}{2} \| \mathbf{Ax}^{(k+1)} - \mathbf{u} - \mathbf{d}^{(k)} \|_2^2 \right\} \\
\mathbf{v}^{(k+1)} &\in \arg \min_{\mathbf{v}} \left\{ \Phi(\mathbf{v}) + \frac{\eta}{2} \| \mathbf{Cx}^{(k+1)} - \mathbf{v} - \mathbf{e}^{(k)} \|_2^2 \right\} \\
\mathbf{d}^{(k+1)} &= \mathbf{d}^{(k)} - \mathbf{Ax}^{(k+1)} + \mathbf{u}^{(k+1)} \\
\mathbf{e}^{(k+1)} &= \mathbf{e}^{(k)} - \mathbf{Cx}^{(k+1)} + \mathbf{v}^{(k+1)}
\end{align*}
\]

- Convergent with inexact updates [AB+11]
Better conditioning with additional VS

Alternative formulation and ADMM (cont’d)

- ADMM iterates [RF12]:

\[
\begin{align*}
\mathbf{x}^{(k+1)} &\in \arg\min_{\mathbf{x}} \left\{ \frac{\rho}{2} \left\| \mathbf{A}\mathbf{x} - \mathbf{u}^{(k)} - \mathbf{d}^{(k)} \right\|_2^2 + \frac{\eta}{2} \left\| \mathbf{C}\mathbf{x} - \mathbf{v}^{(k)} - \mathbf{e}^{(k)} \right\|_2^2 \right\} \\
\mathbf{u}^{(k+1)} &\in \arg\min_{\mathbf{u}} \left\{ \frac{1}{2} \left\| \mathbf{y} - \mathbf{u} \right\|_\mathbf{W}^2 + \frac{\rho}{2} \left\| \mathbf{A}\mathbf{x}^{(k+1)} - \mathbf{u} - \mathbf{d}^{(k)} \right\|_2^2 \right\} \\
\mathbf{v}^{(k+1)} &\in \arg\min_{\mathbf{v}} \left\{ \Phi(\mathbf{v}) + \frac{\eta}{2} \left\| \mathbf{C}\mathbf{x}^{(k+1)} - \mathbf{v} - \mathbf{e}^{(k)} \right\|_2^2 \right\} \\
\mathbf{d}^{(k+1)} &= \mathbf{d}^{(k)} - \mathbf{A}\mathbf{x}^{(k+1)} + \mathbf{u}^{(k+1)} \\
\mathbf{e}^{(k+1)} &= \mathbf{e}^{(k)} - \mathbf{C}\mathbf{x}^{(k+1)} + \mathbf{v}^{(k+1)}
\end{align*}
\]

- Convergent with inexact updates [AB⁺11]
- \( \rho\mathbf{A}'\mathbf{A} + \eta\mathbf{C}'\mathbf{C} \) can be well preconditioned by an appropriate circulant preconditioner in 2D CT
Better conditioning with additional VS

Figure: 2D NCAT: RMS errors as a function of iteration (left) and time (right).
Better conditioning with additional VS

Figure: 2D NCAT: RMS errors as a function of iteration (left) and time (right).

The fact is ...

$A' A$ is still highly shift-variant in 3D CT due to the different geometries and scan trajectories, so this method is still slow in 3D CT
Outline

1. Background
   - Model-based CT reconstruction
   - Fast (2D) CT reconstruction using ADMM

2. OS-LALM: a splitting-based OS algorithm for PWLS problems
   - Linearized AL method with OS acceleration
   - Deterministic downward continuation approach
   - Low-memory OS-LALM with additional variable splits

3. Experimental results
   - Low-dose CT with edge-preserving regularizers
   - Sparse-view CT with TV-like regularizers

4. Conclusion and future work

Hung Nien (U of M)  
Model-based X-ray CT image recon using VS methods with OS  
05/19/2014  22 / 58
Outline

1 Background
   - Model-based CT reconstruction
   - Fast (2D) CT reconstruction using ADMM

2 OS-LALM: a splitting-based OS algorithm for PWLS problems
   - Linearized AL method with OS acceleration
   - Deterministic downward continuation approach
   - Low-memory OS-LALM with additional variable splits

3 Experimental results
   - Low-dose CT with edge-preserving regularizers
   - Sparse-view CT with TV-like regularizers

4 Conclusion and future work
Motivation

What’s wrong with ADMM in CT recon?

\[
\begin{align*}
\hat{x}, \hat{u} & \in \arg \min_{x, u} \left\{ \frac{1}{2} \| \tilde{y} - u \|_2^2 + h(x) \right\} \\
& \text{s.t. } u = \tilde{A}x
\end{align*}
\]
Motivation

What’s wrong with ADMM in CT recon?

- Image update is non-trivial
Motivation

What’s wrong with ADMM in CT recon?

- Image update is non-trivial
- Memory burden of difference images is high
Motivation

What’s wrong with ADMM in CT recon?

- Image update is non-trivial
- Memory burden of difference images is high
- Ordered-subsets (OS) acceleration is not applicable
Motivation

What’s wrong with ADMM in CT recon?

- Image update is non-trivial
- Memory burden of difference images is high
- Ordered-subsets (OS) acceleration is not applicable

Possible solution?

- Proposed formulation [NF13]:

\[
(\hat{x}, \hat{u}) \in \arg \min_{x \in \Omega, u} \left\{ \frac{1}{2} \| W^{1/2} y - u \|_2^2 + R(x) \right\} \quad \text{s.t.} \quad u = W^{1/2} Ax
\]
Motivation

What’s wrong with ADMM in CT recon?

- Image update is non-trivial
- Memory burden of difference images is high
- Ordered-subsets (OS) acceleration is not applicable

Possible solution?

- Proposed formulation [NF13]:

\[
(\hat{x}, \hat{u}) \in \arg \min_{x,u} \left\{ \frac{1}{2} \| \tilde{y} - u \|_2^2 + h(x) \right\} \quad \text{s.t.} \quad u = \tilde{A}x
\]
**Motivation**

What’s wrong with ADMM in CT recon?
- Image update is non-trivial
- Memory burden of difference images is high
- Ordered-subsets (OS) acceleration is not applicable

Possible solution?
- Proposed formulation [NF13]:
  \[
  (\hat{x}, \hat{u}) \in \arg\min_{x,u} \left\{ \frac{1}{2} \| \tilde{y} - u \|^2_2 + h(x) \right\} \quad \text{s.t. } u = \tilde{A}x
  \]
- Image update:
  \[
  x^{(k+1)} \in \arg\min_x \left\{ h(x) + \frac{\rho}{2} \| \tilde{A}x - u^{(k)} - d^{(k)} \|^2_2 \right\}
  \]
Motivation

What’s wrong with ADMM in CT recon?
- Image update is non-trivial
- Memory burden of difference images is high
- Ordered-subsets (OS) acceleration is not applicable

Possible solution?
- Proposed formulation [NF13]:
  \[(\hat{x}, \hat{u}) \in \text{arg min}_{x,u} \left\{ \frac{1}{2} \| \tilde{y} - u \|_2^2 + h(x) \right\} \text{ s.t. } u = \tilde{A}x\]
  
- Image update:
  \[x^{(k+1)} \in \text{arg min}_x \left\{ h(x) + \frac{\rho}{2} \| \tilde{A}x - u^{(k)} - d^{(k)} \|_2^2 \right\} \]
Inexact linearized AL method

The idea is ...

Linearized AL method and proposed variants
Inexact linearized AL method

The idea is ...

- Majorizing $\theta_k(x) \triangleq \frac{\rho}{2} \| \tilde{A}x - u^{(k)} - d^{(k)} \|^2_2$ simplifies image updates

Linearized AL method and proposed variants
Inexact linearized AL method

The idea is ...

- Majorizing $\theta_k(x) \triangleq \frac{\rho}{2} \|\tilde{A}x - u^{(k)} - d^{(k)}\|^2_2$ simplifies image updates

Linearized AL method and proposed variants

- Linearized AL method
  
  $\begin{cases} x^{(k+1)} \in \arg \min_x \left\{ h(x) + \tilde{\theta}_k(x; x^{(k)}) \right\} \\
  u^{(k+1)} \in \arg \min_u \left\{ \frac{1}{2} \|\tilde{y} - u\|^2_2 + \frac{\rho}{2} \|\tilde{A}x^{(k+1)} - u - d^{(k)}\|^2_2 \right\} \\
  d^{(k+1)} = d^{(k)} - \tilde{A}x^{(k+1)} + u^{(k+1)} \end{cases}$
Inexact linearized AL method

The idea is ...

- Majorizing $\theta_k(x) \triangleq \frac{\rho}{2} \| \tilde{A}x - u^{(k)} - d^{(k)} \|^2_2$ simplifies image updates

Linearized AL method and proposed variants

- Linearized AL method

\[
\begin{align*}
\begin{cases} 
    x^{(k+1)} \in \arg \min_x \left\{ h(x) + \tilde{\theta}_k(x; x^{(k)}) \right\} \\
    u^{(k+1)} \in \arg \min_u \left\{ \frac{1}{2} \| \tilde{y} - u \|^2_2 + \frac{\rho}{2} \| \tilde{A}x^{(k+1)} - u - d^{(k)} \|^2_2 \right\} \\
    d^{(k+1)} = d^{(k)} - \tilde{A}x^{(k+1)} + u^{(k+1)}
\end{cases}
\end{align*}
\]

\[
\tilde{\theta}_k(x; x^{(k)}) \triangleq \theta_k(x^{(k)}) + \langle \nabla \theta_k(x^{(k)}) , x - x^{(k)} \rangle + \frac{\rho L}{2} \| x - x^{(k)} \|^2_2
\]
Inexact linearized AL method

The idea is ...

- Majorizing $\theta_k(x) \triangleq \frac{\rho}{2} \| \tilde{A}x - u^{(k)} - d^{(k)} \|_2^2$ simplifies image updates

Linearized AL method and proposed variants

- Linearized AL method

$$
\begin{align*}
    x^{(k+1)} &\in \arg\min_x \left\{ h(x) + \frac{\rho L}{2} \| x - (x^{(k)} - (\rho^{-1} t) \nabla \theta_k(x^{(k)})) \|_2^2 \right\} \\
u^{(k+1)} &\in \arg\min_u \left\{ \frac{1}{2} \| \tilde{y} - u \|_2^2 + \frac{\rho}{2} \| \tilde{A}x^{(k+1)} - u - d^{(k)} \|_2^2 \right\} \\
d^{(k+1)} &= d^{(k)} - \tilde{A}x^{(k+1)} + u^{(k+1)}
\end{align*}
$$

$$
\tilde{\theta}_k(x; x^{(k)}) \triangleq \theta_k(x^{(k)}) + \langle \nabla \theta_k(x^{(k)}) , x - x^{(k)} \rangle + \frac{\rho L}{2} \| x - x^{(k)} \|_2^2
$$
Inexact linearized AL method

The idea is ...

- Majorizing $\theta_k(x) \triangleq \frac{\rho}{2} \| \tilde{A}x - u^{(k)} - d^{(k)} \|^2_2$ simplifies image updates
- Quadratic data-fitting term makes the $u$-updates linear

Linearized AL method and proposed variants

- Linearized AL method

$$
\begin{aligned}
\begin{cases}
    x^{(k+1)} \in \arg \min_x \left\{ h(x) + \frac{\rho L}{2} \| x - (x^{(k)} - (\rho^{-1} t) \nabla \theta_k(x^{(k)})) \|^2_2 \right\} \\
    u^{(k+1)} \in \arg \min_u \left\{ \frac{1}{2} \| \tilde{y} - u \|^2_2 + \frac{\rho}{2} \| \tilde{A}x^{(k+1)} - u - d^{(k)} \|^2_2 \right\} \\
    d^{(k+1)} = d^{(k)} - \tilde{A}x^{(k+1)} + u^{(k+1)}
\end{cases}
\end{aligned}
$$

$$
\tilde{\theta}_k(x; x^{(k)}) \triangleq \theta_k(x^{(k)}) + \langle \nabla \theta_k(x^{(k)}) , x - x^{(k)} \rangle + \frac{\rho L}{2} \| x - x^{(k)} \|^2_2
$$
Inexact linearized AL method

The idea is ...

- Majorizing $\theta_k(x) \triangleq \frac{\rho}{2} \| \tilde{A}x - u(k) - d(k) \|^2_2$ simplifies image updates
- Quadratic data-fitting term makes the $u$-updates linear

Linearized AL method and proposed variants

- Linearized AL method

$$
\begin{aligned}
\begin{cases}
    x^{(k+1)} \in \arg \min_x \left\{ h(x) + \frac{\rho L}{2} \| x - (x^{(k)} - (\rho^{-1} t) \nabla \theta_k(x^{(k)})) \|^2_2 \right\} \\
    u^{(k+1)} = \frac{\rho}{\rho+1} (\tilde{A}x^{(k+1)} - d^{(k)}) + \frac{1}{\rho+1} \tilde{y} \\
    d^{(k+1)} = d^{(k)} - \tilde{A}x^{(k+1)} + u^{(k+1)}
\end{cases}
\end{aligned}
$$

- $\tilde{\theta}_k(x; x^{(k)}) \triangleq \theta_k(x^{(k)}) + \langle \nabla \theta_k(x^{(k)}), x - x^{(k)} \rangle + \frac{\rho L}{2} \| x - x^{(k)} \|^2_2$
Inexact linearized AL method

Linearized AL method and proposed variants (cont’d)

- Gradient-based linearized AL method [NF14b]:

\[
\begin{aligned}
    s^{(k+1)} &= \rho \nabla \ell(x^{(k)}) + (1 - \rho) g^{(k)} \\
    x^{(k+1)} &\in \text{prox}_{(\rho^{-1}t)h}(x^{(k)} - (\rho^{-1}t) s^{(k+1)}) \\
    g^{(k+1)} &= \frac{\rho}{\rho+1} \nabla \ell(x^{(k+1)}) + \frac{1}{\rho+1} g^{(k)}
\end{aligned}
\]
Inexact linearized AL method

Linearized AL method and proposed variants (cont’d)

- Gradient-based linearized AL method [NF14b]:

\[
\begin{align*}
    s^{(k+1)} &= \rho \nabla \ell(x^{(k)}) + (1 - \rho) g^{(k)} \\
    x^{(k+1)} &\in \text{prox}_{(\rho^{-1} t) h}(x^{(k)} - (\rho^{-1} t) s^{(k+1)}) \\
    g^{(k+1)} &= \frac{\rho}{\rho+1} \nabla \ell(x^{(k+1)}) + \frac{1}{\rho+1} g^{(k)}
\end{align*}
\]
Inexact linearized AL method

Linearized AL method and proposed variants (cont’d)

- Gradient-based linearized AL method [NF14b]:
  \[
  \begin{align*}
  s^{(k+1)} &= \rho \nabla \ell(x^{(k)}) + (1 - \rho) g^{(k)} \\
  x^{(k+1)} &\in \text{prox}_{(\rho^{-1}t)h}(x^{(k)} - (\rho^{-1}t) s^{(k+1)}) \\
  g^{(k+1)} &= \frac{\rho}{\rho+1} \nabla \ell(x^{(k+1)}) + \frac{1}{\rho+1} g^{(k)}
  \end{align*}
  \]

- Linearized AL method with OS acceleration ($M$ subsets) [NF14b]:
  \[
  \begin{align*}
  s^{(k,m+1)} &= \rho M \nabla \ell_m(x^{(k,m)}) + (1 - \rho) g^{(k,m)} \\
  x^{(k,m+1)} &\in \text{prox}_{(\rho^{-1}t)h}(x^{(k,m)} - (\rho^{-1}t) s^{(k,m+1)}) \\
  g^{(k,m+1)} &= \frac{\rho}{\rho+1} M \nabla \ell_{m+1}(x^{(k,m+1)}) + \frac{1}{\rho+1} g^{(k,m)}
  \end{align*}
  \]
Inexact linearized AL method

Linearized AL method and proposed variants (cont’d)

- Gradient-based linearized AL method [NF14b]:
  \[
  \begin{align*}
  s^{(k+1)} &= \rho \nabla \ell(x^{(k)}) + (1 - \rho) g^{(k)} \\
  x^{(k+1)} &\in \text{prox}_{(\rho^{-1}t)h}(x^{(k)} - (\rho^{-1}t) s^{(k+1)}) \\
  g^{(k+1)} &= \frac{\rho}{\rho + 1} \nabla \ell(x^{(k+1)}) + \frac{1}{\rho + 1} g^{(k)}
  \end{align*}
  \]

- Linearized AL method with OS acceleration (\(M\) subsets) [NF14b]:
  \[
  \begin{align*}
  s^{(k,m+1)} &= \rho M \nabla \ell_m(x^{(k,m)}) + (1 - \rho) g^{(k,m)} \\
  x^{(k,m+1)} &\in \text{prox}_{(\rho^{-1}t)h}(x^{(k,m)} - (\rho^{-1}t) s^{(k,m+1)}) \\
  g^{(k,m+1)} &= \frac{\rho}{\rho + 1} M \nabla \ell_{m+1}(x^{(k,m+1)}) + \frac{1}{\rho + 1} g^{(k,m)}
  \end{align*}
  \]

Inexact linearized AL method

Linearized AL method and proposed variants (cont’d)

- Gradient-based linearized AL method [NF14b]:
  \[
  \begin{align*}
  s^{(k+1)} &= \rho \nabla \ell(x^{(k)}) + (1 - \rho) g^{(k)} \\
  x^{(k+1)} &\in \text{prox}_{(\rho^{-1} t) h}(x^{(k)} - (\rho^{-1} t) s^{(k+1)}) \\
  g^{(k+1)} &= \frac{\rho}{\rho+1} \nabla \ell(x^{(k+1)}) + \frac{1}{\rho+1} g^{(k)}
  \end{align*}
  \]

- Linearized AL method with OS acceleration (M subsets) [NF14b]:
  \[
  \begin{align*}
  s^{(k,m+1)} &= \rho M \nabla \ell_m(x^{(k,m)}) + (1 - \rho) g^{(k,m)} \\
  x^{(k,m+1)} &\in \text{prox}_{(\rho^{-1} t) h}(x^{(k,m)} - (\rho^{-1} t) s^{(k,m+1)}) \\
  g^{(k,m+1)} &= \frac{\rho}{\rho+1} M \nabla \ell_{m+1}(x^{(k,m+1)}) + \frac{1}{\rho+1} g^{(k,m)}
  \end{align*}
  \]

Inexact linearized AL method

Linearized AL method and proposed variants (cont’d)

- Gradient-based linearized AL method [NF14b]:

\[
\begin{align*}
s^{(k+1)} &= \rho \nabla \ell(x^{(k)}) + (1 - \rho) g^{(k)} \\
x^{(k+1)} &\in \text{prox}_{(\rho^{-1} t) h}(x^{(k)} - (\rho^{-1} t) s^{(k+1)}) \\
g^{(k+1)} &= \frac{\rho}{\rho+1} \nabla \ell(x^{(k+1)}) + \frac{1}{\rho+1} g^{(k)}
\end{align*}
\]

- Linearized AL method with OS acceleration ($M$ subsets) [NF14b]:

\[
\begin{align*}
s^{(k,m+1)} &= \rho M \nabla \ell_m(x^{(k,m)}) + (1 - \rho) g^{(k,m)} \\
x^{(k,m+1)} &\in \text{prox}_{(\rho^{-1} t) h}(x^{(k,m)} - (\rho^{-1} t) s^{(k,m+1)}) \\
g^{(k,m+1)} &= \frac{\rho}{\rho+1} M \nabla \ell_{m+1}(x^{(k,m+1)}) + \frac{1}{\rho+1} g^{(k,m)}
\end{align*}
\]

Inexact linearized AL method

Linearized AL method and proposed variants (cont’d)

- Gradient-based linearized AL method [NF14b]:
  \[
  \begin{align*}
  s^{(k+1)} &= \rho \nabla \ell(x^{(k)}) + (1 - \rho) g^{(k)} \\
  x^{(k+1)} &\in \text{prox}_{(\rho^{-1}t)h}(x^{(k)} - (\rho^{-1}t) s^{(k+1)}) \\
  g^{(k+1)} &= \frac{\rho}{\rho+1} \nabla \ell(x^{(k+1)}) + \frac{1}{\rho+1} g^{(k)}
  \end{align*}
  \]

- Linearized AL method with OS acceleration ($M$ subsets) [NF14b]:
  \[
  \begin{align*}
  s^{(k,m+1)} &= \rho M \nabla \ell_m(x^{(k,m)}) + (1 - \rho) g^{(k,m)} \\
  x^{(k,m+1)} &\in \text{prox}_{(\rho^{-1}t)h}(x^{(k,m)} - (\rho^{-1}t) s^{(k,m+1)}) \\
  g^{(k,m+1)} &= \frac{\rho}{\rho+1} M \nabla \ell_{m+1}(x^{(k,m+1)}) + \frac{1}{\rho+1} g^{(k,m)}
  \end{align*}
  \]

Outline

1. Background
   - Model-based CT reconstruction
   - Fast (2D) CT reconstruction using ADMM

2. OS-LALM: a splitting-based OS algorithm for PWLS problems
   - Linearized AL method with OS acceleration
   - Deterministic downward continuation approach
   - Low-memory OS-LALM with additional variable splits

3. Experimental results
   - Low-dose CT with edge-preserving regularizers
   - Sparse-view CT with TV-like regularizers

4. Conclusion and future work
Faster convergence using continuation

An inconvenient truth ... of ADMM
Faster convergence using continuation

An inconvenient truth ... of ADMM

- ADMM is convergent for any fixed penalty parameter
Faster convergence using continuation

An inconvenient truth ... of ADMM

- ADMM is **convergent** for any fixed penalty parameter
- But, it is **fast** only if the penalty parameter is **chosen appropriately**
Faster convergence using continuation

An inconvenient truth ... of ADMM

- ADMM is convergent for any fixed penalty parameter
- But, it is fast only if the penalty parameter is chosen appropriately
- Choosing the optimal parameter is still an open problem
Faster convergence using continuation

An inconvenient truth ... of ADMM
- ADMM is convergent for any fixed penalty parameter
- But, it is fast only if the penalty parameter is chosen appropriately
- Choosing the optimal parameter is still an open problem

Any alternative?
Faster convergence using continuation

An inconvenient truth … of ADMM
- ADMM is convergent for any fixed penalty parameter
- But, it is fast only if the penalty parameter is chosen appropriately
- Choosing the optimal parameter is still an open problem

Any alternative?
- Yes, continuation!
Faster convergence using continuation

An inconvenient truth ... of ADMM
- ADMM is convergent for any fixed penalty parameter
- But, it is fast only if the penalty parameter is chosen appropriately
- Choosing the optimal parameter is still an open problem

Any alternative?
- Yes, continuation!
- Recall the proximal-gradient image update:
  \[ x^{(k+1)} \in \text{prox}_{(\rho^{-1} t)h}\left(x^{(k)} - (\rho^{-1} t) s^{(k+1)}\right) \]
Faster convergence using continuation

An inconvenient truth ... of ADMM

- ADMM is convergent for any fixed penalty parameter
- But, it is fast only if the penalty parameter is chosen appropriately
- Choosing the optimal parameter is still an open problem

Any alternative?

- Yes, continuation!
- Recall the proximal-gradient image update:

\[
x^{(k+1)} \in \text{prox}_{(\rho^{-1}t)h}(x^{(k)} - (\rho^{-1}t)s^{(k+1)})
\]
Faster convergence using continuation

An inconvenient truth ... of ADMM
- ADMM is convergent for any fixed penalty parameter
- But, it is fast only if the penalty parameter is chosen appropriately
- Choosing the optimal parameter is still an open problem

Any alternative?
- Yes, continuation!
- Recall the proximal-gradient image update:
  \[ x^{(k+1)} \in \text{prox}_{(\rho^{-1}t)h}(x^{(k)} - (\rho^{-1}t)s^{(k+1)}) \]
- We can adjust the step size by varying the penalty parameter
Faster convergence using continuation

An inconvenient truth ... of ADMM

- ADMM is convergent for any fixed penalty parameter
- But, it is fast only if the penalty parameter is chosen appropriately
- Choosing the optimal parameter is still an open problem

Any alternative?

- Yes, continuation!
- Recall the proximal-gradient image update:

\[ x^{(k+1)} \in \text{prox}_{(\rho^{-1}t)h}(x^{(k)} - (\rho^{-1}t)s^{(k+1)}) \]

- We can adjust the step size by varying the penalty parameter (How?)
Faster convergence using continuation

Deterministic downward continuation

Decreasing $\rho_k$ compensates the shrinkage of step length.

Decreasing $\rho_k$ too fast could make the algorithm unstable or diverge.

The designed sequence $[NF14d]$:

$$\rho_k = \begin{cases} 1, & \text{if } k = 0 \\ \pi_k + 1 \sqrt{1 - (\pi_k^2 + 2)^2}, & \text{otherwise} \end{cases}$$

Inspired by a second-order recursive system analysis.

Derivation: An adaptive restart condition takes care of the dependence on $\tilde{A}$. 

Hung Nien (U of M) Model-based X-ray CT image recon using VS methods with OS 05/19/2014 29 / 58
Faster convergence using continuation

**Deterministic downward continuation**
- Decreasing $\rho_k$ compensates the shrinkage of step length
Faster convergence using continuation

Deterministic downward continuation

- Decreasing $\rho_k$ compensates the shrinkage of step length
- Decreasing $\rho_k$ too fast could make the algorithm unstable or diverge
Faster convergence using continuation

Deterministic downward continuation

- Decreasing $\rho_k$ compensates the shrinkage of step length
- Decreasing $\rho_k$ too fast could make the algorithm unstable or diverge
- The designed sequence [NF14d]:

$$\rho_k = \begin{cases} 1, & \text{if } k = 0 \\ \frac{\pi}{k+1} \sqrt{1 - \left(\frac{\pi}{2k+2}\right)^2}, & \text{otherwise} \end{cases}$$
OS-LALM: a splitting-based OS algorithm for PWLS problems

Deterministic downward continuation approach

Faster convergence using continuation

Deterministic downward continuation

- Decreasing $\rho_k$ compensates the shrinkage of step length
- Decreasing $\rho_k$ too fast could make the algorithm unstable or diverge
- The designed sequence [NF14d]:

$$\rho_k = \begin{cases} 
1, & \text{if } k = 0 \\
\frac{\pi}{k+1} \sqrt{1 - \left(\frac{\pi}{2k+2}\right)^2}, & \text{otherwise}
\end{cases}$$

- Inspired by a second-order recursive system analysis

Derivation
Faster convergence using continuation

Deterministic downward continuation

- Decreasing $\rho_k$ compensates the shrinkage of step length
- Decreasing $\rho_k$ too fast could make the algorithm unstable or diverge
- The designed sequence [NF14d]:

$$\rho_k = \begin{cases} 1, & \text{if } k = 0 \\ \frac{\pi}{k+1} \sqrt{1 - \left(\frac{\pi}{2k+2}\right)^2}, & \text{otherwise} \end{cases}$$

- Inspired by a second-order recursive system analysis
- An adaptive restart condition takes care of the dependence on $\tilde{A}$
Outline

1. Background
   - Model-based CT reconstruction
   - Fast (2D) CT reconstruction using ADMM

2. OS-LALM: a splitting-based OS algorithm for PWLS problems
   - Linearized AL method with OS acceleration
   - Deterministic downward continuation approach
   - Low-memory OS-LALM with additional variable splits

3. Experimental results
   - Low-dose CT with edge-preserving regularizers
   - Sparse-view CT with TV-like regularizers

4. Conclusion and future work
OS-LALM with an additional VS

Is additional VS really beneficial?
OS-LALM with an additional VS

Is additional VS really beneficial?

- $\ell_1$-regularization (compressed sensing)
OS-LALM with an additional VS

Is additional VS really beneficial?

- $\ell_1$-regularization (compressed sensing)
- TV-regularization (sparse-view CT)
OS-LALM with an additional VS

Is additional VS really beneficial?

- $\ell_1$-regularization (compressed sensing)
- TV-regularization (sparse-view CT)
- Smooth regularizer with very high curvature (corner-rounding)
OS-LALM with an additional VS

Is additional VS really beneficial?

- $\ell_1$-regularization (compressed sensing)
- TV-regularization (sparse-view CT)
- Smooth regularizer with very high curvature (corner-rounding)

CT reconstruction with “high-memory” VS

- PWLS formulation:
  \[
  \hat{x} \in \arg\min_{x \in \Omega} \left\{ \frac{1}{2} \|\tilde{y} - \tilde{A}x\|_2^2 + \Phi(Cx) \right\}
  \]
OS-LALM with an additional VS

Is additional VS really beneficial?

- $\ell_1$-regularization (compressed sensing)
- TV-regularization (sparse-view CT)
- Smooth regularizer with very high curvature (corner-rounding)

CT reconstruction with “high-memory” VS

- PWLS formulation:
  \[
  \hat{x} \in \arg \min_{x \in \Omega} \left\{ \frac{1}{2} \| \tilde{y} - \tilde{A}x \|_2^2 + \Phi(Cx) \right\}
  \]

- Equivalent formulation [NF14c]:
  \[
  (\hat{x}, \hat{u}, \hat{v}) \in \arg \min_{x \in \Omega, u, v} \left\{ \frac{1}{2} \| \tilde{y} - u \|_2^2 + \Phi(v) \right\} \quad \text{s.t.} \quad u = \tilde{A}x, v = Cx
  \]
OS-LALM: a splitting-based OS algorithm for PWLS problems

Low-memory OS-LALM with additional variable splits

OS-LALM with an additional VS

“High-memory” OS-LALM

\[
\begin{align*}
\mathbf{s}^{(k+1)} &= \rho \nabla \ell(x^{(k)}) + (1 - \rho) g^{(k)} \\
\sigma^{(k+1)} &= \tilde{\eta} C' \left( Cx^{(k)} - v^{(k)} - e^{(k)} \right) \\
x^{(k+1)} &= \left[ x^{(k)} - \frac{1}{\rho L_1 + \tilde{\eta} L_2} \left( \mathbf{s}^{(k+1)} + \sigma^{(k+1)} \right) \right]_{\Omega} \\
g^{(k+1)} &= \frac{\rho}{\rho + 1} \nabla \ell(x^{(k+1)}) + \frac{1}{\rho + 1} g^{(k)} \\
v^{(k+1)} &\in \text{prox}_{\tilde{\eta}^{-1} \Phi} \left( Cx^{(k+1)} - e^{(k)} \right) \\
e^{(k+1)} &= e^{(k)} - Cx^{(k+1)} + v^{(k+1)}
\end{align*}
\]
OS-LALM with an additional VS

“High-memory” OS-LALM

\[
\begin{aligned}
\mathbf{s}^{(k+1)} &= \rho \nabla \ell (\mathbf{x}^{(k)}) + (1 - \rho) \mathbf{g}^{(k)} \\
\mathbf{\sigma}^{(k+1)} &= \bar{\eta} \mathbf{C}^\prime (\mathbf{C} \mathbf{x}^{(k)} - \mathbf{v}^{(k)} - \mathbf{e}^{(k)}) \\
\mathbf{x}^{(k+1)} &= \left[ \mathbf{x}^{(k)} - \frac{1}{\rho L_1 + \bar{\eta} L_2} (\mathbf{s}^{(k+1)} + \mathbf{\sigma}^{(k+1)}) \right] \Omega \\
\mathbf{g}^{(k+1)} &= \frac{\rho}{\rho+1} \nabla \ell (\mathbf{x}^{(k+1)}) + \frac{1}{\rho+1} \mathbf{g}^{(k)} \\
\mathbf{v}^{(k+1)} &\in \text{prox}_{\bar{\eta}^{-1} \Phi} (\mathbf{C} \mathbf{x}^{(k+1)} - \mathbf{e}^{(k)}) \\
\mathbf{e}^{(k+1)} &= \mathbf{e}^{(k)} - \mathbf{C} \mathbf{x}^{(k+1)} + \mathbf{v}^{(k+1)}
\end{aligned}
\]

- Gradient descent-like algorithm with adjustable step sizes
OS-LALM with an additional VS

“High-memory” OS-LALM

\[
\begin{align*}
\mathbf{s}^{(k+1)} &= \rho \nabla \ell (\mathbf{x}^{(k)}) + (1 - \rho) \mathbf{g}^{(k)} \\
\mathbf{\sigma}^{(k+1)} &= \bar{\eta} \mathbf{C}' (\mathbf{C} \mathbf{x}^{(k)} - \mathbf{v}^{(k)} - \mathbf{e}^{(k)}) \\
\mathbf{x}^{(k+1)} &= \left[ \mathbf{x}^{(k)} - \frac{1}{\rho L_1 + \bar{\eta} L_2} (\mathbf{s}^{(k+1)} + \mathbf{\sigma}^{(k+1)}) \right] \Omega \\
\mathbf{g}^{(k+1)} &= \frac{\rho}{\rho + 1} \nabla \ell (\mathbf{x}^{(k+1)}) + \frac{1}{\rho + 1} \mathbf{g}^{(k)} \\
\mathbf{v}^{(k+1)} &\in \text{prox}_{\bar{\eta}^{-1} \Phi} (\mathbf{C} \mathbf{x}^{(k+1)} - \mathbf{e}^{(k)}) \\
\mathbf{e}^{(k+1)} &= \mathbf{e}^{(k)} - \mathbf{C} \mathbf{x}^{(k+1)} + \mathbf{v}^{(k+1)}
\end{align*}
\]

- Gradient descent-like algorithm with \textbf{adjustable} step sizes
- \( \Phi \) can be either smooth or non-smooth (with efficient \text{prox}_\Phi)
OS-LALM with an additional VS

“High-memory” OS-LALM

\[
\begin{align*}
\mathbf{s}^{(k+1)} &= \rho \nabla \ell(\mathbf{x}^{(k)}) + (1 - \rho) \mathbf{g}^{(k)} \\
\sigma^{(k+1)} &= \bar{\eta} \mathbf{C}'(\mathbf{C} \mathbf{x}^{(k)} - \mathbf{v}^{(k)} - \mathbf{e}^{(k)}) \\
\mathbf{x}^{(k+1)} &= \left[\mathbf{x}^{(k)} - \frac{1}{\rho L_1 + \bar{\eta} L_2} (\mathbf{s}^{(k+1)} + \sigma^{(k+1)})\right]_\Omega \\
\mathbf{g}^{(k+1)} &= \frac{\rho}{\rho + 1} \nabla \ell(\mathbf{x}^{(k+1)}) + \frac{1}{\rho + 1} \mathbf{g}^{(k)} \\
\mathbf{v}^{(k+1)} &\in \text{prox}_{\bar{\eta}^{-1} \Phi}(\mathbf{C} \mathbf{x}^{(k+1)} - \mathbf{e}^{(k)}) \\
\mathbf{e}^{(k+1)} &= \mathbf{e}^{(k)} - \mathbf{C} \mathbf{x}^{(k+1)} + \mathbf{v}^{(k+1)}
\end{align*}
\]

- Gradient descent-like algorithm with adjustable step sizes
- \(\Phi\) can be either smooth or non-smooth (with efficient \text{prox}_\Phi)
- Requires two extra image volumes for each direction
OS-LALM with an additional VS

“High-memory” OS-LALM

\[
\begin{align*}
\mathbf{s}^{(k+1)} & = \rho \nabla \ell(\mathbf{x}^{(k)}) + (1 - \rho) \mathbf{g}^{(k)} \\
\mathbf{\sigma}^{(k+1)} & = \overline{\eta} \mathbf{C}'(\mathbf{C}\mathbf{x}^{(k)} - \mathbf{v}^{(k)} - \mathbf{e}^{(k)}) \\
\mathbf{x}^{(k+1)} & = \left[ \mathbf{x}^{(k)} - \frac{1}{\rho L_1 + \overline{\eta} L_2} \left( \mathbf{s}^{(k+1)} + \mathbf{\sigma}^{(k+1)} \right) \right]_\Omega \\
\mathbf{g}^{(k+1)} & = \frac{\rho}{\rho + 1} \nabla \ell(\mathbf{x}^{(k+1)}) + \frac{1}{\rho + 1} \mathbf{g}^{(k)} \\
\mathbf{v}^{(k+1)} & \in \text{prox}_{\overline{\eta}^{-1} \Phi}(\mathbf{C}\mathbf{x}^{(k+1)} - \mathbf{e}^{(k)}) \\
\mathbf{e}^{(k+1)} & = \mathbf{e}^{(k)} - \mathbf{C}\mathbf{x}^{(k+1)} + \mathbf{v}^{(k+1)}
\end{align*}
\]

- Gradient descent-like algorithm with adjustable step sizes
- \( \Phi \) can be either smooth or non-smooth (with efficient \text{prox}_\Phi)
- Requires two extra image volumes for each direction
- Remarkable memory and computational overhead
The idea is ...

The auxiliary variables $v$ and $e$ can be large, but $C'v$ and $C'e$ are not. "Low-memory" OS-LALM makes all $v$- and $e$-related updates linear!

At the $k$th iteration, replace $\Phi$ by $\tilde{\Phi}(v; Cx(k+1)) \propto v' \nabla \Phi(Cx(k+1)) + L\Phi \|v - Cx(k+1)\|^2_2$

The $v$-update has a linear approximate solution:

$$v(k+1) \approx Cx(k+1) - (\bar{\eta} \bar{\eta} + L\Phi e(k) + L\Phi \bar{\eta} + L\Phi L^{-1}\Phi \nabla \Phi(Cx(k+1)))$$
The idea is ...

- The auxiliary variables $v$ and $e$ can be large, but $C'v$ and $C'e$ are not...

Majoirzing $\Phi$ makes all $v$- and $e$-related updates linear!

"Low-memory" OS-LALM

At the $k$th iteration, replace $\Phi$ by $\tilde{\Phi}(v; Cx(k+1)) \propto v' \nabla \Phi(Cx(k+1)) + L_\Phi 2 \| v - Cx(k+1) \|^2$

The $v$-update has a linear approximate solution:

$$v(k+1) \approx Cx(k+1) - (\bar{\eta}\bar{\eta} + L_\Phi e(k) + L_\Phi L^{-1} \Phi \nabla \Phi(Cx(k+1)))$$
Low-memory OS-LALM with “compressed” VS

The idea is ...

- The auxiliary variables $\mathbf{v}$ and $\mathbf{e}$ can be large, but $\mathbf{C}'\mathbf{v}$ and $\mathbf{C}'\mathbf{e}$ are not.
- Majorizing $\Phi$ makes all $\mathbf{v}$- and $\mathbf{e}$-related updates linear!
OS-LALM: a splitting-based OS algorithm for PWLS problems

Low-memory OS-LALM with additional variable splits

**Low-memory OS-LALM with “compressed” VS**

The idea is ... 

- The auxiliary variables \( \mathbf{v} \) and \( \mathbf{e} \) can be large, but \( \mathbf{C}' \mathbf{v} \) and \( \mathbf{C}' \mathbf{e} \) are not.
- Majorizing \( \Phi \) makes all \( \mathbf{v} \)- and \( \mathbf{e} \)-related updates linear!

“Low-memory” OS-LALM

- At the \( k \)th iteration, replace \( \Phi \) by

\[
\bar{\Phi}(\mathbf{v}; \mathbf{C}_x^{(k+1)}) \propto \mathbf{v}' \nabla \Phi(\mathbf{C}_x^{(k+1)}) + \frac{L_\Phi}{2} \left\| \mathbf{v} - \mathbf{C}_x^{(k+1)} \right\|^2_2
\]
The idea is ...

- The auxiliary variables $v$ and $e$ can be large, but $C'v$ and $C'e$ are not.
- Majorizing $\Phi$ makes all $v$- and $e$-related updates linear!

"Low-memory" OS-LALM

- At the $k$th iteration, replace $\Phi$ by

$$\tilde{\Phi}(v;Cx^{(k+1)}) \propto v'\nabla\Phi(Cx^{(k+1)}) + \frac{L\Phi}{2}\|v - Cx^{(k+1)}\|^2_2$$

- The $v$-update has a linear approximate solution:

$$v^{(k+1)} \approx Cx^{(k+1)} - \left(\frac{\tilde{\eta}}{\tilde{\eta} + L\Phi}e^{(k)} + \frac{L\Phi}{\tilde{\eta} + L\Phi}L^{-1}\Phi(Cx^{(k+1)})\right)$$
Low-memory OS-LALM with “compressed” VS

“Low-memory” OS-LALM (cont’d)

\[
\begin{align*}
    s^{(k+1)} &= \rho \nabla \ell(x^{(k)}) + (1 - \rho) g^{(k)} \\
    \sigma^{(k+1)} &= \eta (\bar{v}^{(k)} - \tilde{e}^{(k)}) \\
    x^{(k+1)} &= \left[ x^{(k)} - \frac{1}{\rho L_1 + \eta L_2} \left( s^{(k+1)} + \sigma^{(k+1)} \right) \right] \Omega \\
    g^{(k+1)} &= \frac{\rho}{\rho + 1} \nabla \ell(x^{(k+1)}) + \frac{1}{\rho + 1} g^{(k)} \\
    \bar{v}^{(k+1)} &= \frac{\eta}{\eta + 1} \tilde{e}^{(k)} + \frac{1}{\eta + 1} \nabla R(x^{(k+1)}) \\
    \tilde{e}^{(k+1)} &= \tilde{e}^{(k)} - \bar{v}^{(k+1)}
\end{align*}
\]

Suppose \( \Phi \) is smooth, and \( \nabla \Phi \) is \( \mathcal{L}_\Phi \)-Lipschitz.

No explicit proximal mapping is in the updates.

Requires only two extra image volumes for all directions.
OS-LALM: a splitting-based OS algorithm for PWLS problems

Low-memory OS-LALM with additional variable splits

Low-memory OS-LALM with “compressed” VS

“Low-memory” OS-LALM (cont’d)

\[
\begin{align*}
    \mathbf{s}^{(k+1)} &= \rho \nabla \ell(\mathbf{x}^{(k)}) + (1 - \rho) \mathbf{g}^{(k)} \\
    \mathbf{\sigma}^{(k+1)} &= \eta \left( \mathbf{\bar{v}}^{(k)} - \mathbf{\tilde{e}}^{(k)} \right) \\
    \mathbf{x}^{(k+1)} &= \left[ \mathbf{x}^{(k)} - \frac{1}{\rho L_1 + \eta L_\Phi} \left( \mathbf{s}^{(k+1)} + \mathbf{\sigma}^{(k+1)} \right) \right] \Omega \\
    \mathbf{g}^{(k+1)} &= \frac{\rho}{\rho + 1} \nabla \ell(\mathbf{x}^{(k+1)}) + \frac{1}{\rho + 1} \mathbf{g}^{(k)} \\
    \mathbf{\bar{v}}^{(k+1)} &= \frac{\eta}{\eta + 1} \mathbf{\tilde{e}}^{(k)} + \frac{1}{\eta + 1} \nabla R(\mathbf{x}^{(k+1)}) \\
    \mathbf{\tilde{e}}^{(k+1)} &= \mathbf{\tilde{e}}^{(k)} - \mathbf{\bar{v}}^{(k+1)}
\end{align*}
\]

- Suppose $\Phi$ is smooth, and $\nabla \Phi$ is $L_\Phi$-Lipschitz
Low-memory OS-LALM with “compressed” VS

“Low-memory” OS-LALM (cont’d)

\[
\begin{align*}
    s^{(k+1)} &= \rho \nabla \ell(x^{(k)}) + (1 - \rho) g^{(k)} \\
    \sigma^{(k+1)} &= \eta (\bar{v}^{(k)} - \tilde{e}^{(k)}) \\
    x^{(k+1)} &= \left[ x^{(k)} - \frac{1}{\rho L_1 + \eta L_2} \left( s^{(k+1)} + \sigma^{(k+1)} \right) \right] \Omega \\
    g^{(k+1)} &= \frac{\rho}{\rho + 1} \nabla \ell(x^{(k+1)}) + \frac{1}{\rho + 1} g^{(k)} \\
    \bar{v}^{(k+1)} &= \frac{\eta}{\eta + 1} \tilde{e}^{(k)} + \frac{1}{\eta + 1} \nabla R(x^{(k+1)}) \\
    \tilde{e}^{(k+1)} &= \tilde{e}^{(k)} - \bar{v}^{(k+1)}
\end{align*}
\]

- Suppose \( \Phi \) is smooth, and \( \nabla \Phi \) is \( L_\Phi \)-Lipschitz
- No explicit proximal mapping is in the updates
Suppose $\Phi$ is smooth, and $\nabla \Phi$ is $L_\Phi$-Lipschitz.

No explicit proximal mapping is in the updates.

Requires only two extra image volumes for all directions.
Outline

1 Background
   - Model-based CT reconstruction
   - Fast (2D) CT reconstruction using ADMM

2 OS-LALM: a splitting-based OS algorithm for PWLS problems
   - Linearized AL method with OS acceleration
   - Deterministic downward continuation approach
   - Low-memory OS-LALM with additional variable splits

3 Experimental results
   - Low-dose CT with edge-preserving regularizers
   - Sparse-view CT with TV-like regularizers

4 Conclusion and future work
Experimental results

Setup and notation

OS configuration

OS with the bit-reversal order

Separable quadratic surrogate with Hessian $D_L \equiv \text{diag}\left\{A^\prime W A_1\right\}$

Naming conventions

OS-SQS-M: the standard OS algorithm [EF99]

OS-Nes05-M: the state-of-the-art OS + momentum algorithm [KR13]

OS-LALM-M-ρ-n: the proposed one-split algorithm [NF14b; NF14d]

OS-LALM-M-c-η (low-mem): the proposed two-split algorithm

Hung Nien (U of M)
Setup and notation

OS configuration

- OS with the bit-reversal order
- Separable quadratic surrogate with Hessian $D_L \triangleq \text{diag}\{A'WA_1\}$
Setup and notation

OS configuration
- OS with the bit-reversal order
- Separable quadratic surrogate with Hessian $D_L \triangleq \text{diag}\{A'WA1\}$

Naming conventions
- OS-SQS-M: the standard OS algorithm [EF99]
- OS-Nes05-M: the state-of-the-art OS+momentum algorithm [KR+13]
- OS-LALM-M-$\rho$-$n$: the proposed one-split algorithm [NF14b; NF14d]
- OS-LALM-M-$c$-$\eta$ (low-mem): the proposed two-split algorithm
Outline

1 Background
   - Model-based CT reconstruction
   - Fast (2D) CT reconstruction using ADMM

2 OS-LALM: a splitting-based OS algorithm for PWLS problems
   - Linearized AL method with OS acceleration
   - Deterministic downward continuation approach
   - Low-memory OS-LALM with additional variable splits

3 Experimental results
   - Low-dose CT with edge-preserving regularizers
   - Sparse-view CT with TV-like regularizers

4 Conclusion and future work
Shoulder region helical scan

Specification

- **Image size:** $512 \times 512 \times 109$
- **Sinogram size:** $888 \times 32 \times 7146$ (about 7 turns, pitch = 0.5)
Shoulder region helical scan

Specification

- Image size: $512 \times 512 \times 109$
- Sinogram size: $888 \times 32 \times 7146$ (about 7 turns, pitch $= 0.5$)

**Figure:** Shoulder: the initial FBP image (left), the reference reconstruction (middle), and the reconstructed image using OS-LALM after 30 iterations (right).
Shoulder region helical scan

Figure: Shoulder: RMS differences as a function of iteration using different OS-based algorithms with 20 subsets (left) and 40 subsets (right).
**Figure:** Shoulder: RMS differences as a function of iteration using different OS-based algorithms with 20 subsets (left) and 40 subsets (right).
Shoulder region helical scan

**Figure:** Shoulder: RMS differences as a function of iteration using different OS-based algorithms with 20 subsets (left) and 40 subsets (right).
Shoulder region helical scan

**Figure:** Shoulder: RMS differences as a function of iteration using different OS-based algorithms with 20 subsets (left) and 40 subsets (right).
Shoulder region helical scan

**Figure:** Shoulder: RMS differences as a function of iteration using different OS-based algorithms with 20 subsets (left) and 40 subsets (right).
Shoulder region helical scan

Figure: Shoulder: RMS differences as a function of iteration using different OS-based algorithms with 20 subsets (left) and 40 subsets (right).
**Shoulder region helical scan**

*Figure:* Shoulder: RMS differences as a function of iteration using different OS-based algorithms with 20 subsets (left) and 40 subsets (right).
Shoulder region helical scan

**Figure:** Shoulder: RMS differences as a function of iteration using different OS-based algorithms with 20 subsets (left) and 40 subsets (right).
Shoulder region helical scan

Figure: Shoulder: reconstructed images using different OS-based algorithms after 30 iterations.
**Shoulder region helical scan**

**Figure:** Shoulder: difference images of the reconstructed images using different OS-based algorithms after 30 iterations.
GE performance phantom axial scan

Specification
- Image size: $1024 \times 1024 \times 90$
- Sinogram size: $888 \times 64 \times 984$ (less view redundancy, cf. helical scan)
GE performance phantom axial scan

Specification

- Image size: $1024 \times 1024 \times 90$
- Sinogram size: $888 \times 64 \times 984$ (less view redundancy, cf. helical scan)

Figure: GEPP: the initial FBP image (left), the reference reconstruction (middle), and the reconstructed image using OS-LALM after 30 iterations (right).
**GE performance phantom axial scan**

**Figure:** GEPP: RMS differences as a function of iteration using different OS-based algorithms with 12 subsets (left) and 24 subsets (right).
Figure: GEPP: difference images of the reconstructed images using different OS-based algorithms with 12 subsets after 30 iterations.
Figure: GEPP: difference images of the reconstructed images using different OS-based algorithms with 24 subsets after 30 iterations.
Outline

1 Background
   - Model-based CT reconstruction
   - Fast (2D) CT reconstruction using ADMM

2 OS-LALM: a splitting-based OS algorithm for PWLS problems
   - Linearized AL method with OS acceleration
   - Deterministic downward continuation approach
   - Low-memory OS-LALM with additional variable splits

3 Experimental results
   - Low-dose CT with edge-preserving regularizers
   - Sparse-view CT with TV-like regularizers

4 Conclusion and future work
Chest region half scan

Specification

- Image size: $718 \times 718 \times 122$
- Sinogram size: $888 \times 64 \times 81$ (about 12.6% views are used)
Chest region half scan

Specification

- Image size: $718 \times 718 \times 122$
- Sinogram size: $888 \times 64 \times 81$ (about 12.6% views are used)

Figure: Chest: the initial FBP image (left), the reference reconstruction (middle), and the reconstructed image using OS-LALM after 100 iterations (right).
Chest region half scan

Figure: Chest: RMS differences as a function of iteration (left) and time (right) using OS-LALM with $M = 5$ and different values of $\eta$. 
Chest region half scan

**Figure:** Chest: RMS differences as a function of iteration (left) and time (right) using OS-LALM with different values of $M$ and $\eta = 0.05$. 
Chest region half scan

Figure: Chest: RMS differences as a function of iteration (left) and time (right) using different OS-based algorithms with $M = 5$. 
Chest region half scan

Figure: Chest: reconstructed images using different OS-based algorithms with $M = 5$ after 100 iterations.
Outline

1. Background
   - Model-based CT reconstruction
   - Fast (2D) CT reconstruction using ADMM

2. OS-LALM: a splitting-based OS algorithm for PWLS problems
   - Linearized AL method with OS acceleration
   - Deterministic downward continuation approach
   - Low-memory OS-LALM with additional variable splits

3. Experimental results
   - Low-dose CT with edge-preserving regularizers
   - Sparse-view CT with TV-like regularizers

4. Conclusion and future work
Conclusion

We proposed a splitting-based OS algorithm, OS-LALM, for solving PWLS X-ray CT image reconstruction problems. A deterministic downward continuation approach for accelerating the proposed algorithm. A low-memory variant of the proposed algorithm when considering additional variable splits. Experimental results showed that the proposed algorithm significantly accelerates the convergence of X-ray CT image reconstruction with negligible overhead. The proposed algorithm is stable when using many subsets for OS acceleration.
Conclusion

We proposed ...

- A splitting-based OS algorithm, OS-LALM, for solving PWLS X-ray CT image reconstruction problems
- A deterministic downward continuation approach for accelerating the proposed algorithm
- A low-memory variant of the proposed algorithm when considering additional variable splits
Conclusion

We proposed ...

- A splitting-based OS algorithm, OS-LALM, for solving PWLS X-ray CT image reconstruction problems
- A deterministic downward continuation approach for accelerating the proposed algorithm
- A low-memory variant of the proposed algorithm when considering additional variable splits

Experimental results showed that ...

- The proposed algorithm significantly accelerates the convergence of X-ray CT image reconstruction with negligible overhead
- The proposed algorithm is stable when using many subsets for OS acceleration
Future work

Future work

- Convergence analysis of OS-LALM when $M > 1$
- Convergence analysis of OS-LALM with downward continuation
- Optimal downward continuation and restart condition
- Convergence analysis of low-mem OS-LALM
- Parameter selection for low-mem OS-LALM

Extension

- Non-quadratic data-fitting term
- Low-mem OS-LALM with non-smooth potential functions
Future work

Theory

- Convergence analysis of OS-LALM when $M > 1$
- Convergence analysis of OS-LALM with downward continuation
- Optimal downward continuation and restart condition
- Convergence analysis of low-mem OS-LALM
- Parameter selection for low-mem OS-LALM
Future work

Theory

- Convergence analysis of OS-LALM when $M > 1$
- Convergence analysis of OS-LALM with downward continuation
- Optimal downward continuation and restart condition
- Convergence analysis of low-mem OS-LALM
- Parameter selection for low-mem OS-LALM

Extension

- Non-quadratic data-fitting term
- Low-mem OS-LALM with non-smooth potential functions
References I


References III


THANK YOU!

any question?
Second-order recursive system analysis

Minimize a quadratic function

\[ \hat{x} \in \arg \min_x x_1^2 \|Ax\|_2^2 \]

Let \( V\Lambda V' \) be the EVD of \( A' A \), where \( 0 < \mu = \lambda_1 \leq \ldots \leq \lambda_n = L \).

LALM iterates (with a fixed \( \rho \)):

\[
\begin{align*}
\{x(k+1) &= x(k) - \frac{1}{L} (V\Lambda V' x(k) + (\rho - 1 - 1)g(k)) \\
g(k+1) &= \frac{\rho}{\rho + 1} V\Lambda V' x(k+1) + \frac{1}{\rho + 1} g(k)
\end{align*}
\]

The diagonalized system \( \bar{x} \equiv V' x \) and \( \bar{g} \equiv V' g \) satisfies a 2nd-order recursive system determined by the characteristic polynomial:

\[
(1 + \rho) r^2 - 2 (1 - \lambda_i/L + \rho/2) r + (1 - \lambda_i/L) = 0
\]
Second-order recursive system analysis

Minimize a quadratic function

- Simple quadratic programming:

\[ \hat{x} \in \arg \min_x \frac{1}{2} \|Ax\|_2^2 \]
Second-order recursive system analysis

Minimize a quadratic function

- Simple quadratic programming:

\[ \hat{x} \in \arg\min_x \frac{1}{2} \|Ax\|^2 \]

- Let \( V\Lambda V' \) be the EVD of \( A'A \), where \( 0 < \mu = \lambda_1 \leq \ldots \leq \lambda_n = L \).
Second-order recursive system analysis

Minimize a quadratic function

- Simple quadratic programming:

\[ \hat{x} \in \arg\min_x \frac{1}{2} \|Ax\|_2^2 \]

- Let \( V\Lambda V' \) be the EVD of \( A'\!A \), where \( 0 < \mu = \lambda_1 \leq \ldots \leq \lambda_n = L \)

- LALM iterates (with a fixed \( \rho \)):

\[
\begin{align*}
\mathbf{x}^{(k+1)} &= \mathbf{x}^{(k)} - \left(\frac{1}{L}\right) \left( V\Lambda V'\mathbf{x}^{(k)} + \left( \frac{\rho^{-1}}{2} - 1 \right) \mathbf{g}^{(k)} \right) \\
\mathbf{g}^{(k+1)} &= \frac{\rho}{\rho + 1} V\Lambda V'\mathbf{x}^{(k+1)} + \frac{1}{\rho + 1} \mathbf{g}^{(k)}
\end{align*}
\]
Second-order recursive system analysis

Minimize a quadratic function

- Simple quadratic programming:
  \[ \hat{x} \in \arg\min_x \frac{1}{2} \|Ax\|^2 \]

- Let \( V\Lambda V' \) be the EVD of \( A'A \), where \( 0 < \mu = \lambda_1 \leq \ldots \leq \lambda_n = L \)

- LALM iterates (with a fixed \( \rho \)):
  \[
  \begin{cases}
    x^{(k+1)} = x^{(k)} - \left(\frac{1}{L}\right) (V\Lambda V'x^{(k)} + (\rho^{-1} - 1) g^{(k)}) \\
    g^{(k+1)} = \frac{\rho}{\rho+1} V\Lambda V'x^{(k+1)} + \frac{1}{\rho+1} g^{(k)}
  \end{cases}
  \]

- The diagonalized system \( \tilde{x} \triangleq V'x \) and \( \tilde{g} \triangleq V'g \) satisfies a 2nd-order recursive system determined by the characteristic polynomial:
  \[
  (1 + \rho) r^2 - 2 \left(1 - \frac{\lambda_i}{L} + \frac{\rho}{2}\right) r + \left(1 - \frac{\lambda_i}{L}\right)
  \]
Second-order recursive system analysis

System behaviors based on the value of $\rho$

Let $\rho \equiv 2\sqrt{\lambda_i} L (1 - \lambda_i) L$

$\rho = \rho^\star_i$: critically damped
$\rho > \rho^\star_i$: over-damped
$\rho < \rho^\star_i$: under-damped, oscillates with frequency $\psi_i \approx \sqrt{\lambda_i / L}$

We also observed that in practice, the asymptotic convergence rate of the system is determined by the eigencomponent with the smallest eigenvalue $\lambda_1 = \mu$. For $\lambda_i < L / 2$, the critically damped system converges fastest.
Second-order recursive system analysis

System behaviors based on the value of $\rho$

Let

$$\rho_i^* \triangleq 2 \sqrt{\frac{\lambda_i}{L}} \left(1 - \frac{\lambda_i}{L}\right)$$

$\rho_i^*$ critically damped

$\rho > \rho_i^*$: over-damped

$\rho < \rho_i^*$: under-damped, oscillates with frequency $\psi_i \approx \sqrt{\frac{\lambda_i}{L}}$
Second-order recursive system analysis

System behaviors based on the value of $\rho$

Let

$$\rho_i^* \triangleq 2 \sqrt{\frac{\lambda_i}{L}} \left(1 - \frac{\lambda_i}{L}\right)$$

- $\rho = \rho_i^*$: critically damped

In practice, the asymptotic convergence rate of the system is determined by the eigencomponent with the smallest eigenvalue $\lambda_1$. For $\lambda_i < L/2$, the critically damped system converges fastest.
Second-order recursive system analysis

System behaviors based on the value of $\rho$

Let

$$\rho_i^* \triangleq 2\sqrt{\frac{\lambda_i}{L}} \left(1 - \frac{\lambda_i}{L}\right)$$

- $\rho = \rho_i^*$: critically damped
- $\rho > \rho_i^*$: over-damped
Second-order recursive system analysis

System behaviors based on the value of $\rho$

Let

$$\rho_i^* \triangleq 2 \sqrt{\frac{\lambda_i}{L} \left(1 - \frac{\lambda_i}{L}\right)}$$

- $\rho = \rho_i^*$: critically damped
- $\rho > \rho_i^*$: over-damped
- $\rho < \rho_i^*$: under-damped, oscillates with frequency $\psi_i \approx \sqrt{\lambda_i/L}$

We also observed that in practice, the asymptotic convergence rate of the system is determined by the eigencomponent with the smallest eigenvalue $\lambda_1 = \mu$.

For $\lambda_i < L/2$, the critically damped system converges fastest.
Second-order recursive system analysis

System behaviors based on the value of $\rho$

Let

$$\rho_i^* \triangleq 2 \sqrt{\frac{\lambda_i}{L} \left(1 - \frac{\lambda_i}{L}\right)}$$

- $\rho = \rho_i^*$: critically damped
- $\rho > \rho_i^*$: over-damped
- $\rho < \rho_i^*$: under-damped, oscillates with frequency $\psi_i \approx \sqrt{\frac{\lambda_i}{L}}$

We also observed that

- In practice, the asympt. convergence rate of the system is determined by the eigencomponent with the smallest eigenvalue $\lambda_1 = \mu$
Second-order recursive system analysis

System behaviors based on the value of $\rho$

Let

$$\rho_i^* \triangleq 2 \sqrt{\frac{\lambda_i}{L}} \left(1 - \frac{\lambda_i}{L}\right)$$

- $\rho = \rho_i^*$: critically damped
- $\rho > \rho_i^*$: over-damped
- $\rho < \rho_i^*$: under-damped, oscillates with frequency $\psi_i \approx \sqrt{\frac{\lambda_i}{L}}$

We also observed that

- In practice, the asymp. convergence rate of the system is determined by the eigencomponent with the smallest eigenvalue $\lambda_1 = \mu$
- For $\lambda_i < L/2$, the critically damped system converges fastest
Second-order recursive system analysis

System behaviors based on the value of $\rho$ (cont’d)

To attain the fastest asymptotic convergence rate, we would like to choose

$$\rho^* = \rho_1^* = 2\sqrt{\frac{\mu}{L}} \left(1 - \frac{\mu}{L}\right)$$
Second-order recursive system analysis

System behaviors based on the value of $\rho$ (cont’d)

To attain the fastest asymptotic convergence rate, we would like to choose

$$
\rho^* = \rho_1^* = 2 \sqrt{\frac{\mu}{L} (1 - \frac{\mu}{L})}
$$

**Figure:** Asymptotic convergence rate of a system with 6 distinct eigenvalues.
Second-order recursive system analysis

Design a decreasing sequence $\rho_k \rightarrow \rho^*$
Design a decreasing sequence $\rho_k \to \rho^*$

We know that

- As the algorithm proceeds, only the component oscillating at the frequency $\psi_1 \approx \sqrt{\mu/L}$ persists
Design a decreasing sequence $\rho_k \to \rho^*$

We know that

- As the algorithm proceeds, only the component oscillating at the frequency $\psi_1 \approx \sqrt{\mu/L}$ persists.

- In this case, $\xi(k) \triangleq (g^{(k)} - \nabla \ell(x^{(k+1)}))' (\nabla \ell(x^{(k+1)}) - \nabla \ell(x^{(k)}))$ oscillates at the frequency $2\sqrt{\mu/L}$.
Design a decreasing sequence $\rho_k \to \rho^*$

We know that

- As the algorithm proceeds, only the component oscillating at the frequency $\psi_1 \approx \sqrt{\mu/L}$ persists.
- In this case, $\xi(k) \triangleq (g^{(k)} - \nabla \ell(x^{(k+1)}))' (\nabla \ell(x^{(k+1)}) - \nabla \ell(x^{(k)}))$ oscillates at the frequency $2\sqrt{\mu/L}$.
- If the algorithm is restarted when $\xi(k) > 0$, we shall observe the next restart signal after a further $(\pi/2)\sqrt{L/\mu}$ iterations.
Design a decreasing sequence $\rho_k \to \rho^*$

We know that

- As the algorithm proceeds, only the component oscillating at the frequency $\psi_1 \approx \sqrt{\mu/L}$ persists
- In this case, $\xi(k) \triangleq (g^{(k)} - \nabla \ell(x^{(k+1)}))' (\nabla \ell(x^{(k+1)}) - \nabla \ell(x^{(k)}))$ oscillates at the frequency $2\sqrt{\mu/L}$
- If the algorithm is restarted when $\xi(k) > 0$, we shall observe the next restart signal after a further $(\pi/2)\sqrt{L/\mu}$ iterations
- We can easily design a decreasing sequence $\rho_k$ that reaches $\rho^*$ every time we restart the algorithm!