Iterative Methods for Image Deblurring

Math 561

Fall, 2006
Outline

Introduction

The Computational Problem

Regularization Review

Iterative Methods
Class of Difficult Problems

\[ b(s) = \int a(s, t)x(t)dt + e(s) \]

\[ b = Ax + e \]

where

- **A** - known (implicitly) large, ill-conditioned matrix
- **b** - known, given data
- **e** - noise, statistical properties may be known

Goal: Compute an approximation of \( x \).
Applications: Image Deblurring

\[ b = Ax + e \]

- Given blurred image, \( b \), information about the blurring, \( A \), and noise, \( e \).

- Goal: Compute approximation of true image, \( x \).
Applications: Image Deblurring

\[ b = Ax + e \]

- Given blurred image, \( b \), information about the blurring, \( A \), and noise, \( e \).
- Goal: Compute approximation of true image, \( x \).
Applications: Super Resolution for Iris Recognition

Basic idea: Given set of low resolution images,
Applications: Super Resolution for Iris Recognition

Basic idea: Given set of low resolution images,
Applications: Super Resolution for Iris Recognition

Basic idea: Given set of low resolution images, combine to get a high resolution image
Applications: Super Resolution for Iris Recognition

Basic idea: Given set of low resolution images, combine to get a high resolution image without distortions.
Applications: Super Resolution for Medical Imaging

Similar to iris recognition, but

- Registration (alignment) of images more difficult.
- Solving $\mathbf{b} = \mathbf{A}\mathbf{x} + \mathbf{e}$ is a subproblem of a nonlinear optimization method.
The Computational Problem

From the matrix-vector equation

\[ b = Ax + e \]

- Given \( b \) and \( A \), compute an approximation of \( x \)

- Regarding the noise, \( e \):
  - It is usually not known.
  - However, some statistical information may be known.
  - It is usually small, but it cannot be ignored!

That is, solving the linear algebra problem:

\[ Ax = b \quad \Rightarrow \quad x = A^{-1}b \]

usually does not work.
The Computational Problem

- Given $A$ and $b = Ax + e$

- Goal: Compute approximation of true image, $x$

- Naïve inverse solution

\[
\hat{x} = A^{-1}b \\
= A^{-1}(Ax + e) \\
= x + A^{-1}e \\
= x + \text{error}
\]

is corrupted with noise!
The Computational Problem

- Given $A$ and $b = Ax + e$

- Goal: Compute approximation of true image, $x$

- Naïve inverse solution

\[
\hat{x} = A^{-1}b \\
= A^{-1} (Ax + e) \\
= x + A^{-1}e \\
= x + \text{error}
\]

is corrupted with noise!
The Computational Problem

- Given $A$ and $b = Ax + e$

- Goal: Compute approximation of true image, $x$

- Naïve inverse solution

  $$\hat{x} = A^{-1}b$$
  $$= A^{-1} (Ax + e)$$
  $$= x + A^{-1}e$$
  $$= x + \text{error}$$

  is corrupted with noise!
Regularization

**Basic Idea:** Modify the inversion process to compute:

\[ x_{\text{reg}} = A_{\text{reg}}^{-1} b \]

so that

\[ \hat{x} = A_{\text{reg}}^{-1} b \]

\[ = A_{\text{reg}}^{-1} (Ax + e) \]

\[ = A_{\text{reg}}^{-1} Ax + A_{\text{reg}}^{-1} e \]

where

\[ A_{\text{reg}}^{-1} Ax \approx x \quad \text{and} \quad A_{\text{reg}}^{-1} e \text{ is not too large} \]
Regularization

Examples of well known regularization methods:

- Tikhonov
- Truncated Singular Value Decomposition (TSVD)
- Total Variation

Difficulty: Computing $A_{\text{reg}}^{-1}b$ is often very computationally expensive when $A$ is large.

(In our problems, $A$ can easily be $10^6 \times 10^6$)
For large matrices use iterative methods

\[ x_0 = \text{initial estimate of } x \]

\text{for } k = 1, 2, 3, \ldots

- \quad x_k = \text{computations involving } x_{k-1}, A, b, \text{ a preconditioner matrix, } M, \text{ and other intermediate quantities}
- \quad \text{determine if stopping criteria are satisfied}

\text{end}

\text{▶ } x_k \text{ converges to } A^{-1}b

\text{▶ Expensive computations at each iteration:}
  \text{▶ } \text{Multiplying } A \text{ times a vector.}
  \text{▶ } \text{Applying the preconditioner.}

These computations are usually cheap for sparse and structured matrices.
Properties of iterative methods for our problem:

- Early iterations:
  - $x_k$ begins to reconstruct $x$
  - error term is small

- But, eventually iteration converges to $\hat{x} = A^{-1}b = x + \text{error}$

- Thus, as $k$ gets large, $x_k$ is dominated by error

- Goal is to stop iteration when:
  - $x_k$ is a good approximation of $x$
  - error is not too large
Properties of iterative methods for our problem:

- Early iterations:
  - $x_k$ begins to reconstruct $x$
  - error term is small

- But, eventually iteration converges to $\hat{x} = A^{-1}b = x + \text{error}$

- Thus, as $k$ gets large, $x_k$ is dominated by error

- Goal is to stop iteration when:
  - $x_k$ is a good approximation of $x$
  - error is not too large
Properties of iterative methods for our problem:

- Early iterations:
  - $x_k$ begins to reconstruct $x$
  - error term is small

- But, eventually iteration converges to $\hat{x} = A^{-1}b = x + \text{error}$

- Thus, as $k$ gets large, $x_k$ is dominated by error

- Goal is to stop iteration when:
  - $x_k$ is a good approximation of $x$
  - error is not too large
Properties of iterative methods for our problem:

- Early iterations:
  - $x_k$ begins to reconstruct $x$
  - error term is small

- But, eventually iteration converges to $\hat{x} = A^{-1}b = x + \text{error}$

- Thus, as $k$ gets large, $x_k$ is dominated by error

- Goal is to stop iteration when:
  - $x_k$ is a good approximation of $x$
  - error is not too large
Example of iterative method LSQR
Example of iterative method LSQR
Example of iterative method LSQR
Example of iterative method LSQR
Example of iterative method LSQR
Example of iterative method LSQR
Example of iterative method LSQR
Example of iterative method LSQR
Difficulty with iterative methods

- Difficult to find an appropriate stopping iteration, $k_{\text{stop}}$.
  - The error plot in the previous example gives $k_{\text{stop}} = k_{\text{opt}}$.
  - However, can only plot errors if true solution is known!

- Other methods can be used to estimate $k_{\text{stop}}$.
  - But they often choose $k_{\text{stop}} > k_{\text{opt}}$.
    Thus, computed solution contains too much error!
Hybrid Method

Basic Idea:

- Use iterative method for $b = Ax + e$

- At each iteration:
  - Project very large problem to a very small problem.
  - Use sophisticated regularization methods to solve very small problem.
  - Project solution of very small problem back to very large problem.

- Advantage: error term does not grow!

- Thus, it is okay to have $k_{\text{stop}} > k_{\text{opt}}$. 
Example of hybrid method
Example of hybrid method
Example of hybrid method
Example of hybrid method
Example of hybrid method
Example of hybrid method
Example of hybrid method
Hybrid Methods

In any case, the hybrid methods still have the form:

\[ x_0 = \text{initial estimate of } x \]
\[ \text{for } k = 1, 2, 3, \ldots \]
\[ \bullet \quad x_k = \text{computations involving } x_{k-1}, A, b, \]
\[ \quad \text{a preconditioner matrix, } M, \text{ and } \]
\[ \quad \text{other intermediate quantities} \]
\[ \bullet \quad \text{determine if stopping criteria are satisfied} \]

end

\[ \rightarrow x_k \text{ converges to } A^{-1}_{\text{reg}} b \]

\[ \rightarrow \text{Expensive computations at each iteration:} \]
\[ \quad \rightarrow \text{Multiplying } A \text{ times a vector.} \]
\[ \quad \rightarrow \text{Applying the preconditioner.} \]