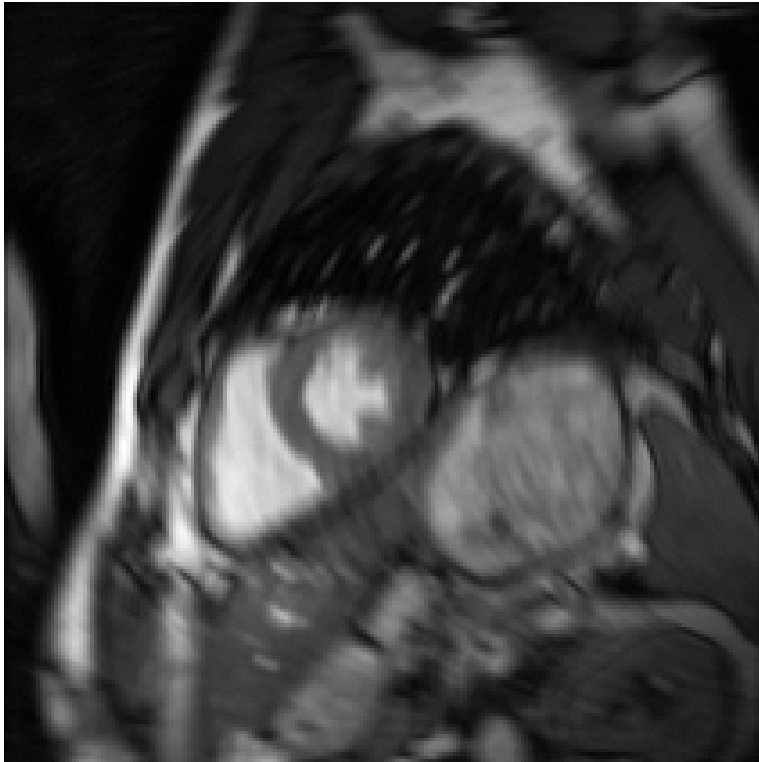

Fast Algorithms
for the
Removal of Non-Uniform Motion Blurs

James G. Nagy
Emory University
Atlanta, GA

Joint work with Julianne Chung, Eldad Haber, John Oshinski

Motivating Example

Motion in MRI cardiac image



Restored MRI cardiac image

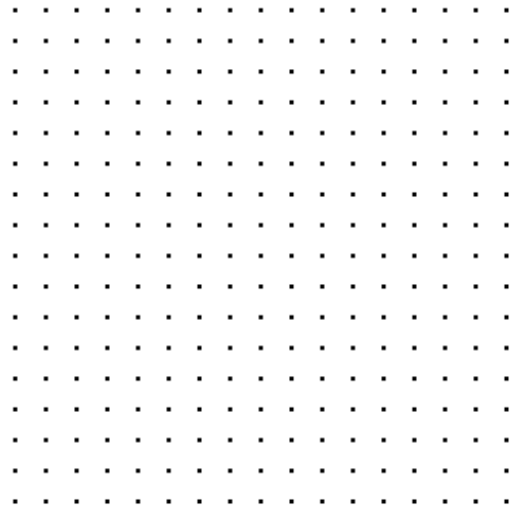


Outline

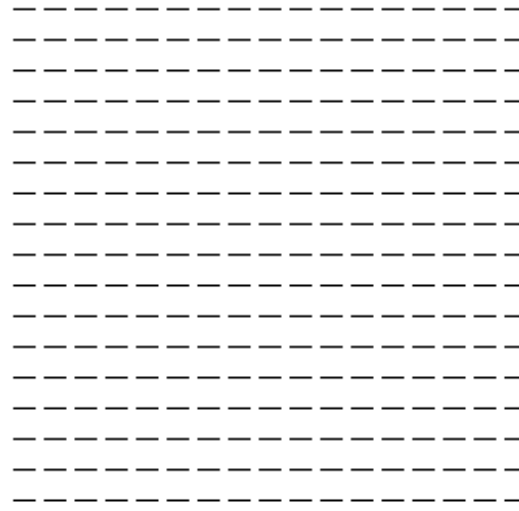
- Uniform motion blurs, matrix models
- Non-uniform motion blurs, matrix models
- Algorithms
- Difficulties

Uniform Motion Blur

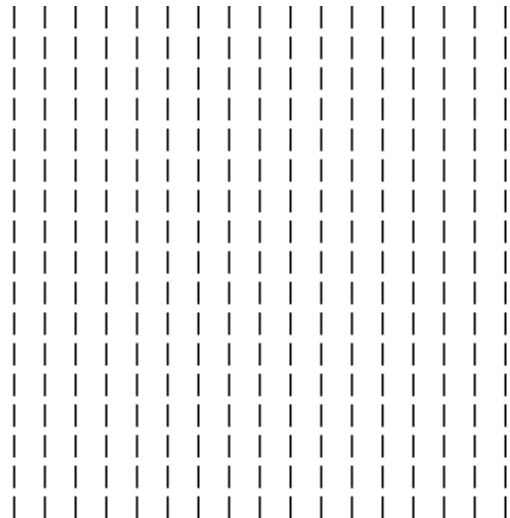
True Image



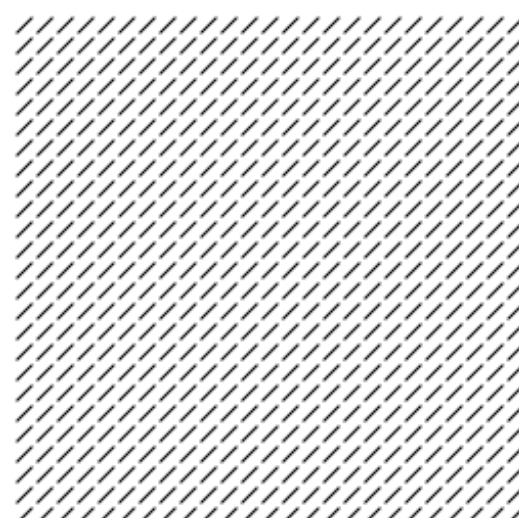
Horizontal Blur



Vertical Blur

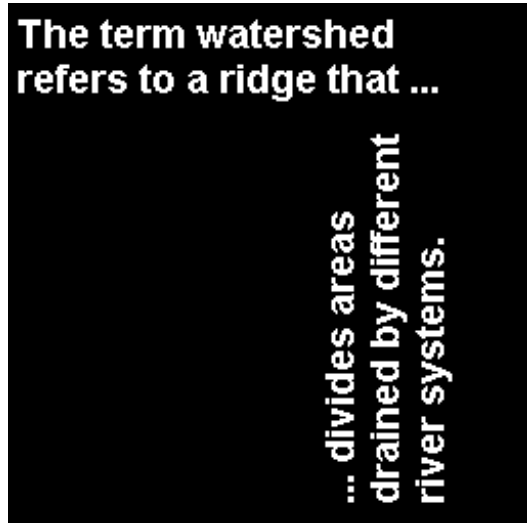


Angled Blur

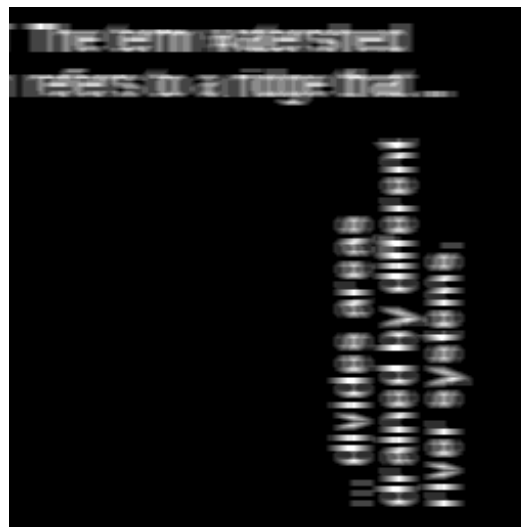


Uniform Motion Blur

True Image



Horizontal Blur



Vertical Blur



Angled Blur



Uniform Motion Blur

$$\text{Linear model: } \mathbf{b} = A\mathbf{x} + \mathbf{n}$$

The matrix A is structured:

$$\text{Horizontal blur} \Rightarrow A = T \otimes I$$

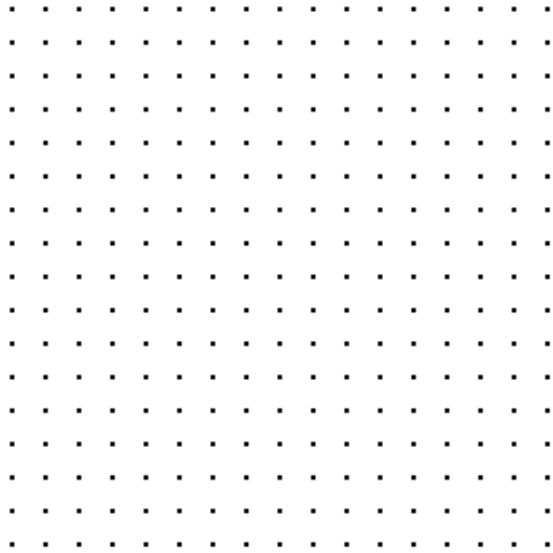
$$\text{Vertical blur} \Rightarrow A = I \otimes T$$

$$\text{where } T = \begin{bmatrix} \frac{1}{d} & & & & & & \\ & \vdots & & & & & \\ & & \frac{1}{d} & & & & \\ & & & \dots & & & \\ & & & & \dots & & \\ & & & & & \frac{1}{d} & \dots & \frac{1}{d} \end{bmatrix}$$

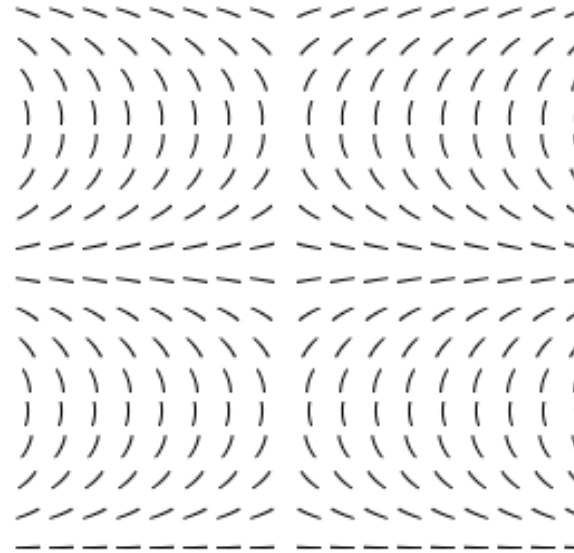
$$\text{Angled blur} \Rightarrow A = \text{block Toeplitz, Toeplitz blocks}$$

Non-Uniform Motion Blur

True Image

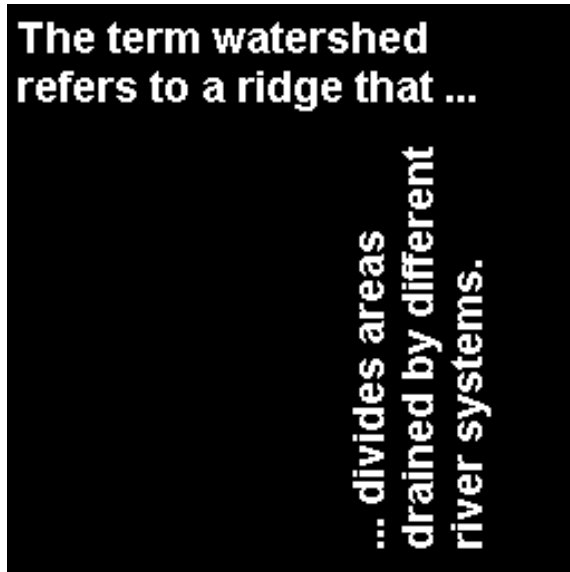


Non-Uniform Blur



Non-Uniform Motion Blur

True Image



Non-Uniform Blur



We still have linear model

$$\mathbf{b} = \mathbf{A}\mathbf{x} + \mathbf{n}$$

However, \mathbf{A} has no obvious structure.

Non-Uniform Motion Blur

To explicitly construct A :

- Estimate direction and magnitude of motion at each pixel
- Construct corresponding column of A
- Use sparse data structure for A

Problem: It may be difficult to estimate motion at every pixel

Non-Uniform Motion Blur

To approximate A :

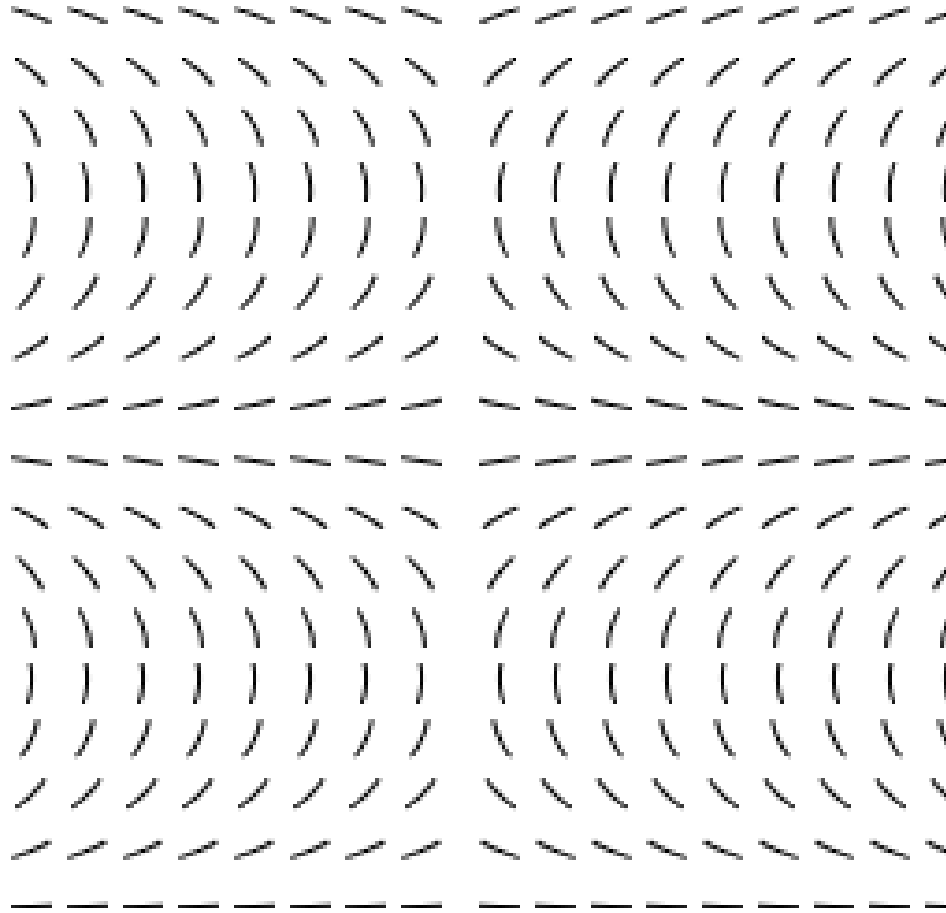
- Partition image into regions
- Assume motion is uniform in each region
- Estimate direction and magnitude of "uniform" motion in each region
- Use interpolation:

$$A \approx \sum I_k A_k$$

where A_k is defined by uniform motion in k th region and I_k is diagonal, with $\sum I_k = I$

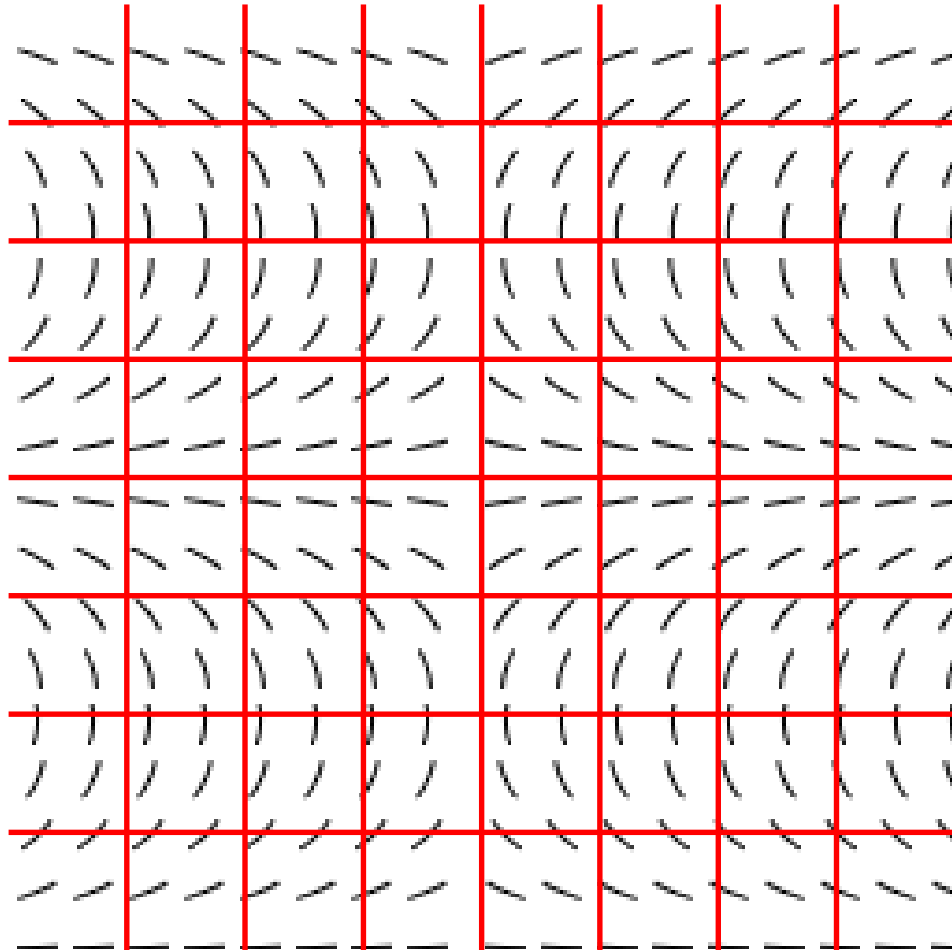
Region Partitioning

One region



Region Partitioning

64 regions



Deblurring Algorithms

Given A (or its approximation), need to solve

$$\mathbf{b} = A\mathbf{x} + \mathbf{n}$$

where

- A is ill-conditioned
⇒ need regularization
- A is large
⇒ usually need iterative method

Deblurring Algorithms

Possible regularization methods

- Truncated singular value decomposition (TSVD)

$$A = U\Sigma V^T \quad \Rightarrow \quad \mathbf{x}_{\text{tsvd}} = \sum_{i=1}^k \frac{\mathbf{u}_i^T \mathbf{b}}{\sigma_i} \mathbf{v}_i$$

(can use efficiently for horizontal or vertical blurs)

- Tikhonov:

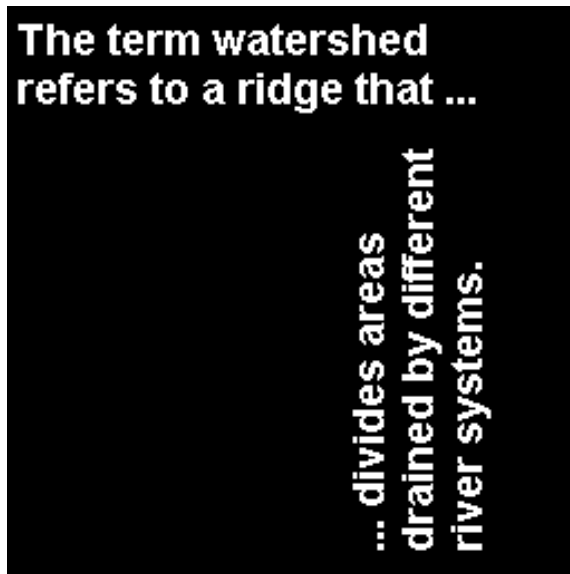
$$\min \left\{ \|\mathbf{b} - A\mathbf{x}\|^2 + \mu^2 \|L\mathbf{x}\|^2 \right\}$$

- Iterative: Terminate iterations early
(e.g., conjugate gradients)

Numerical Examples

First consider text data:

True Image

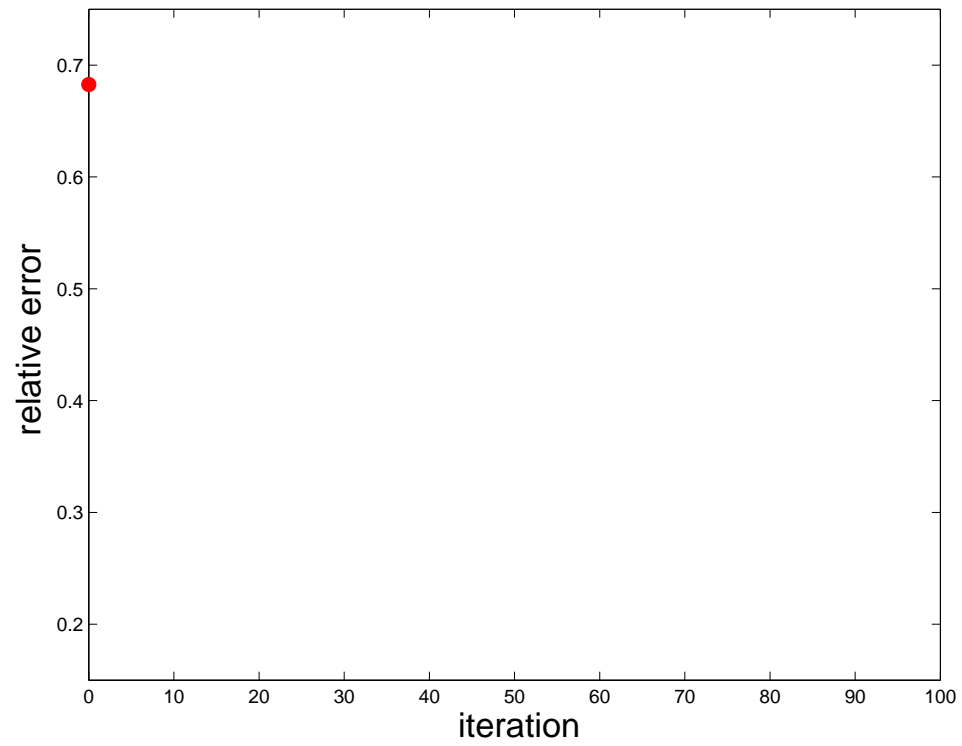


Non-Uniform Blur



- Use CGLS, iterative regularization.
- For A ,
 - Approximate motion on regions
 - Use motion information at every pixel

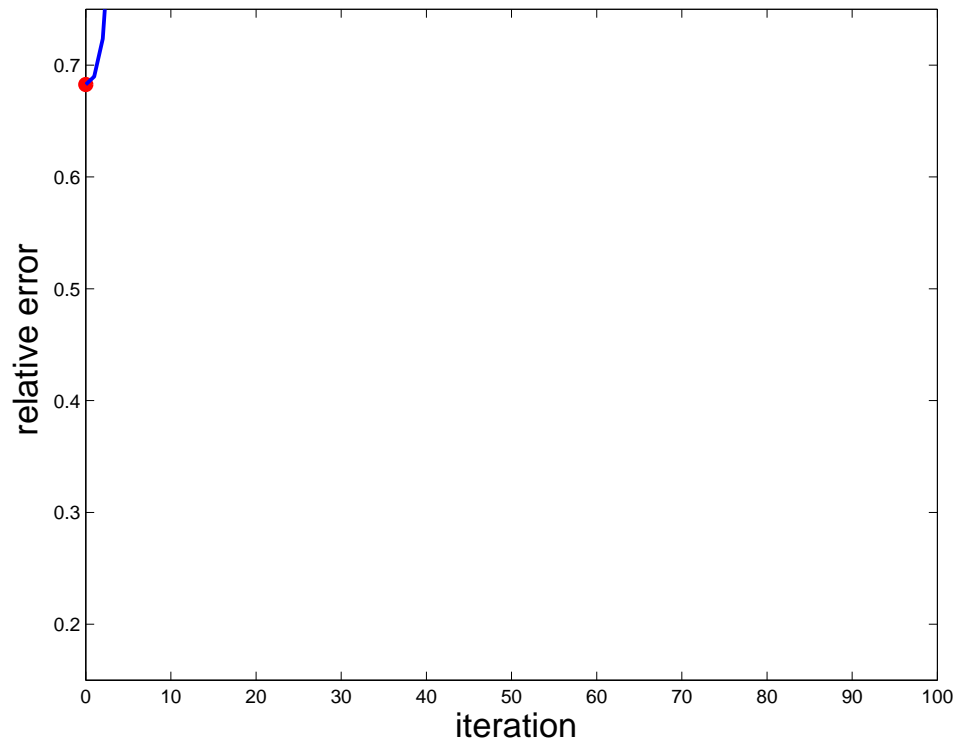
Numerical Examples



Blurred image



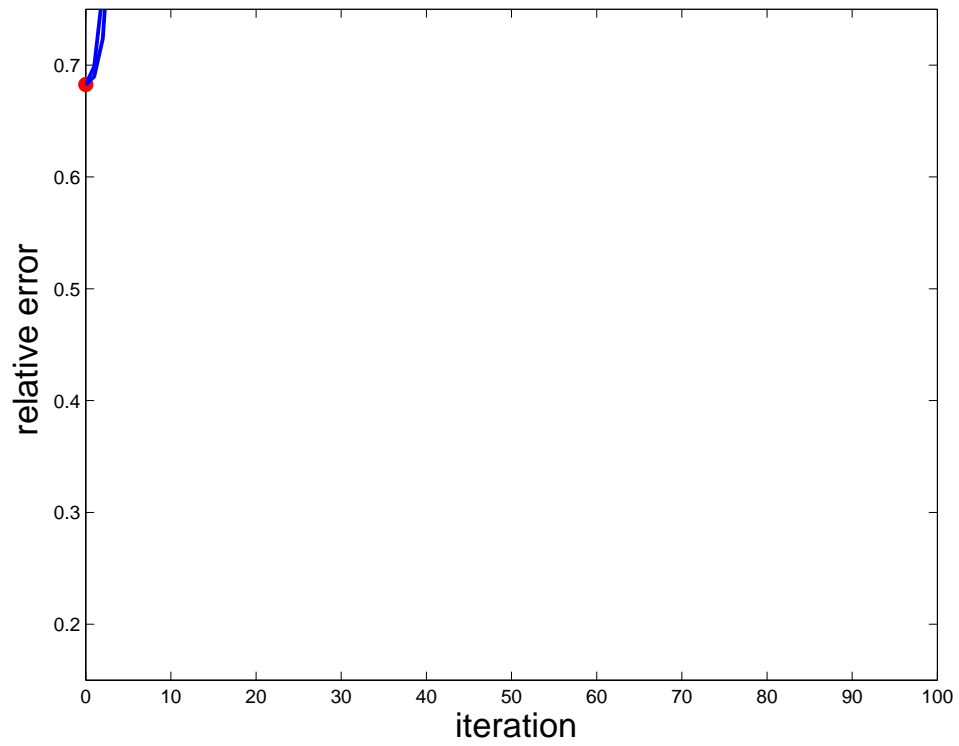
Numerical Examples



1 region, 0 iterations



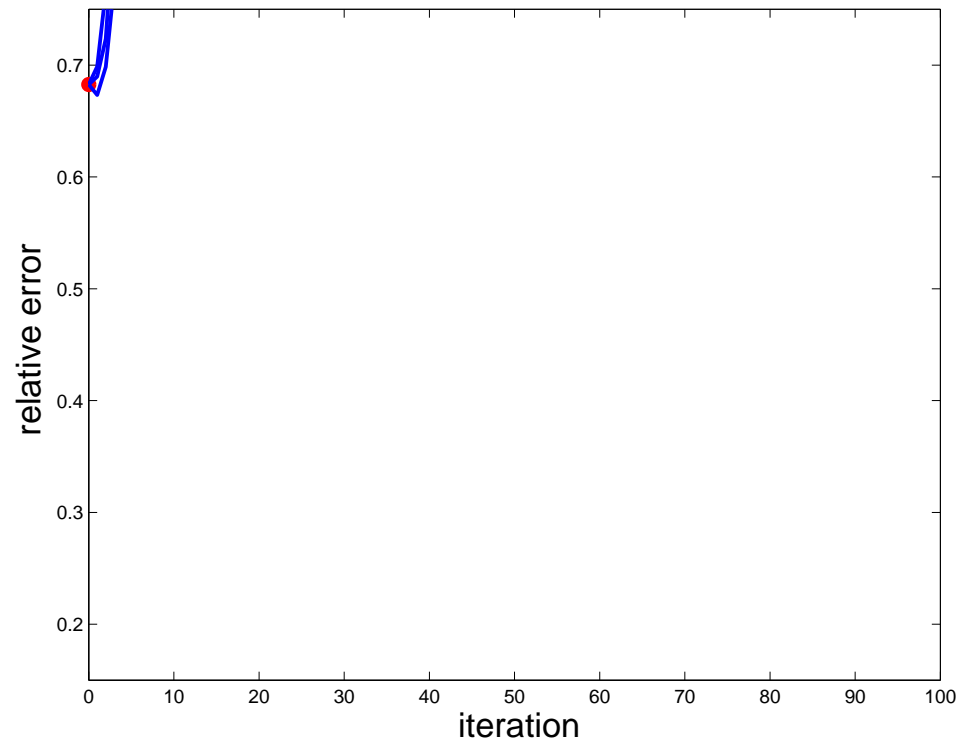
Numerical Examples



4 region, 0 iterations



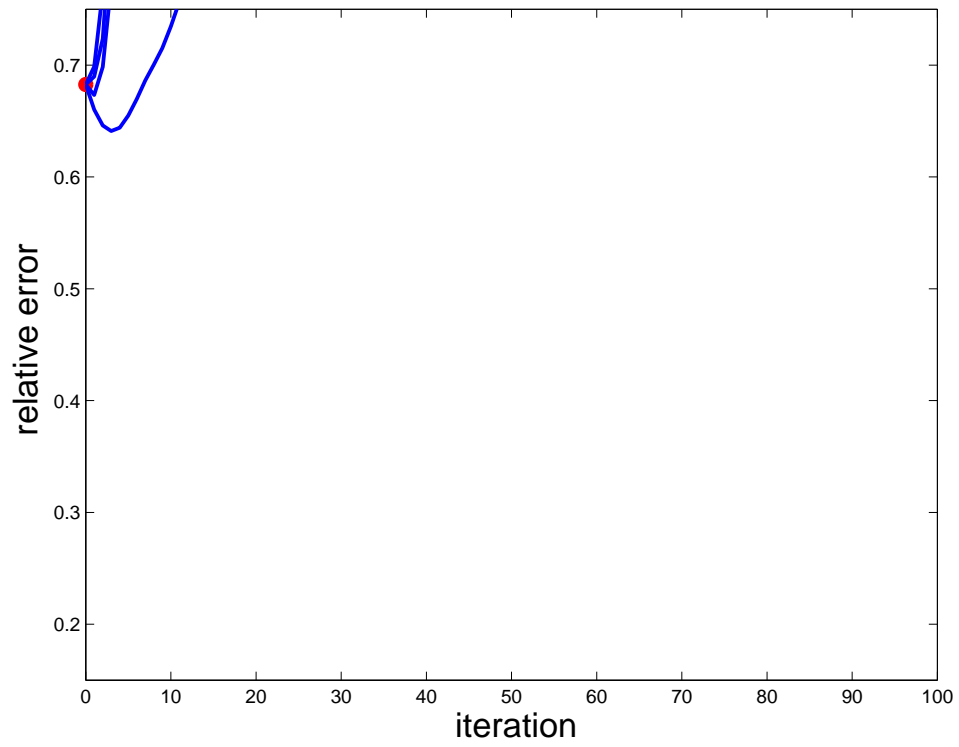
Numerical Examples



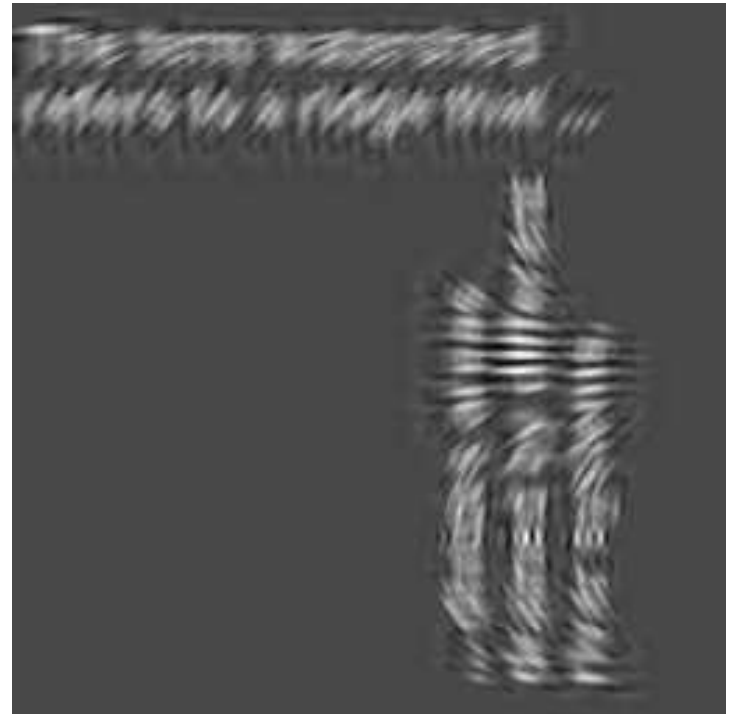
16 region, 1 iterations



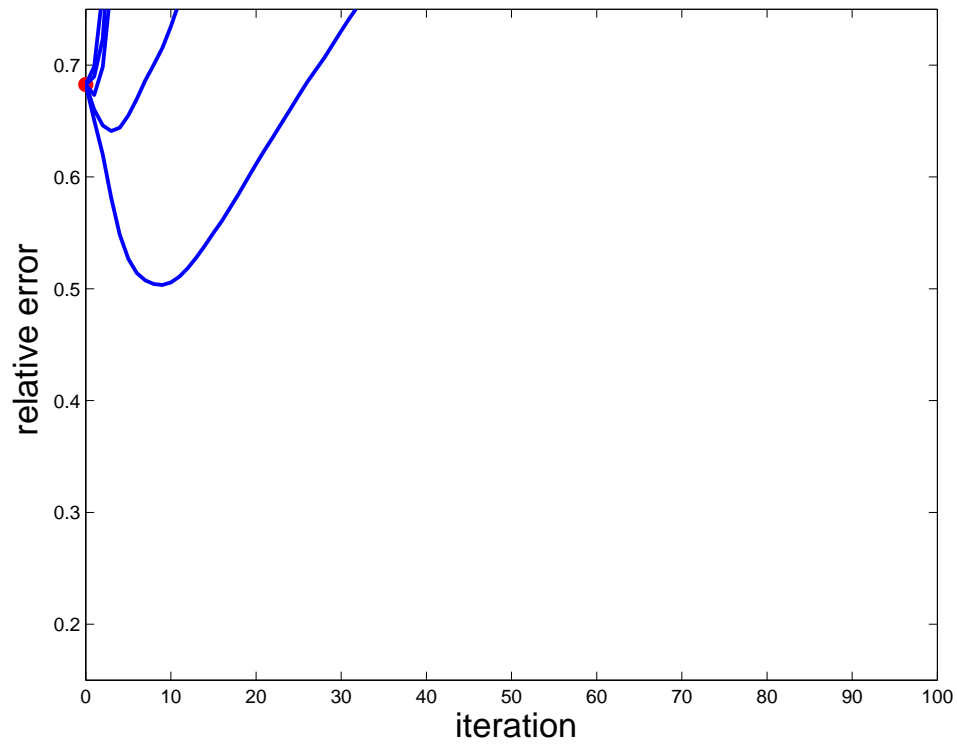
Numerical Examples



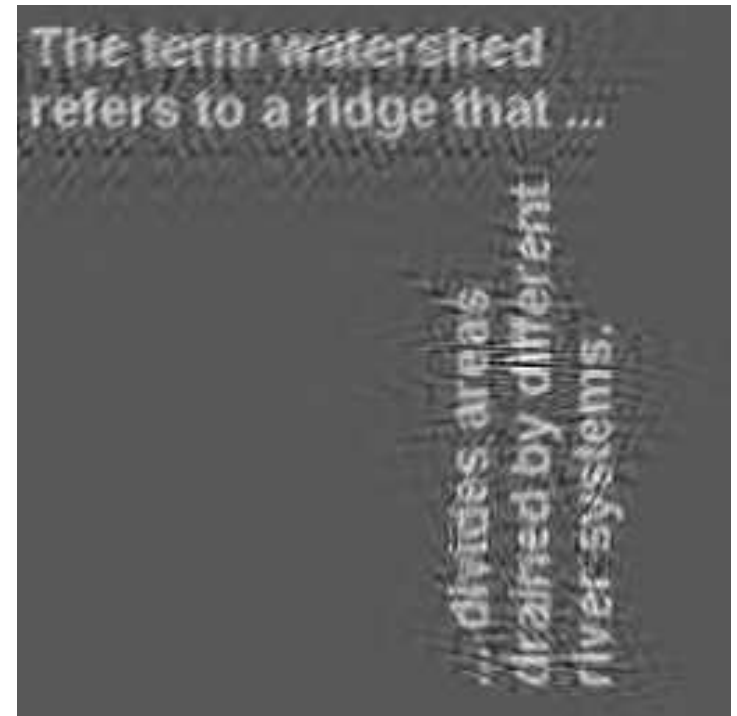
64 region, 3 iterations



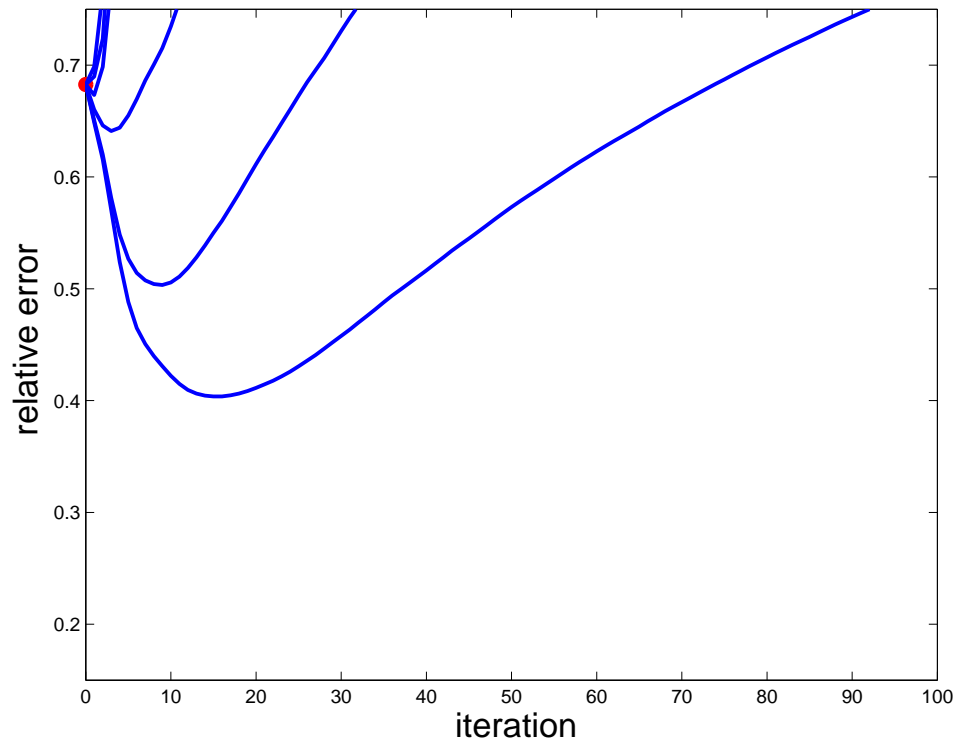
Numerical Examples



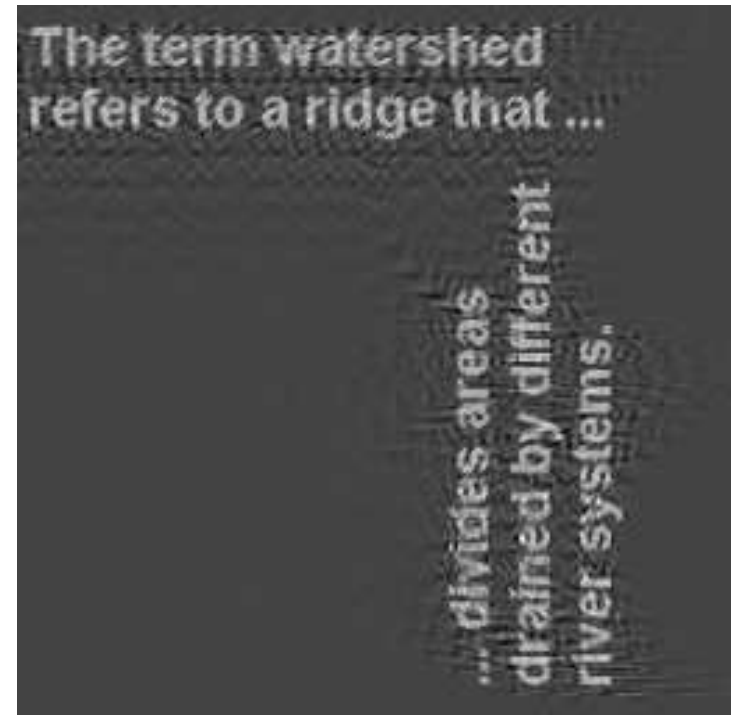
256 region, 9 iterations



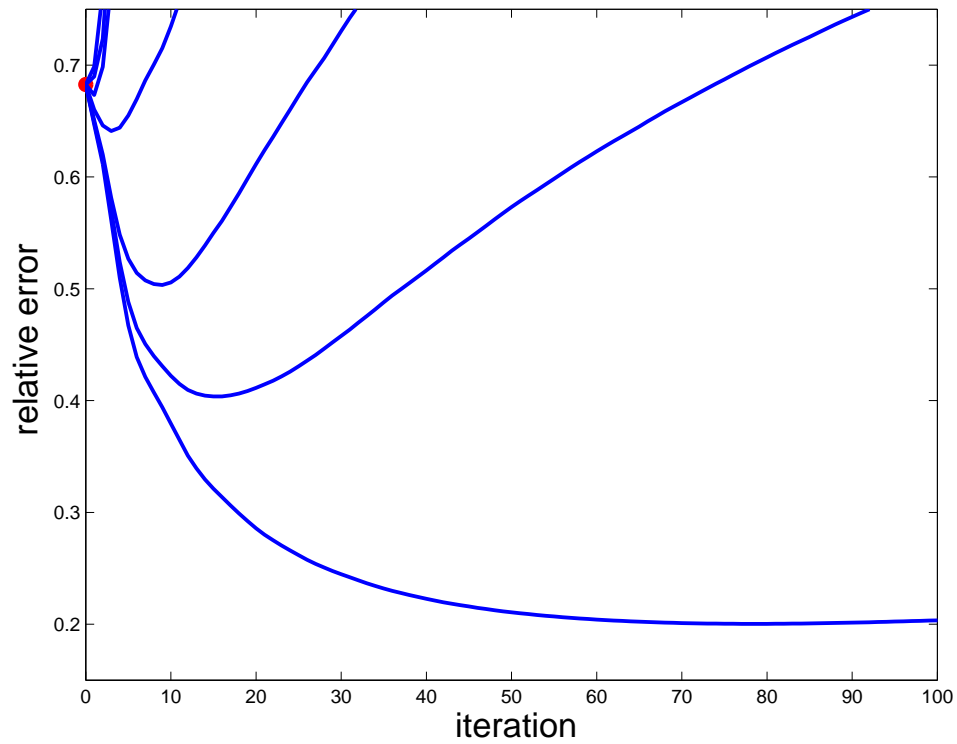
Numerical Examples



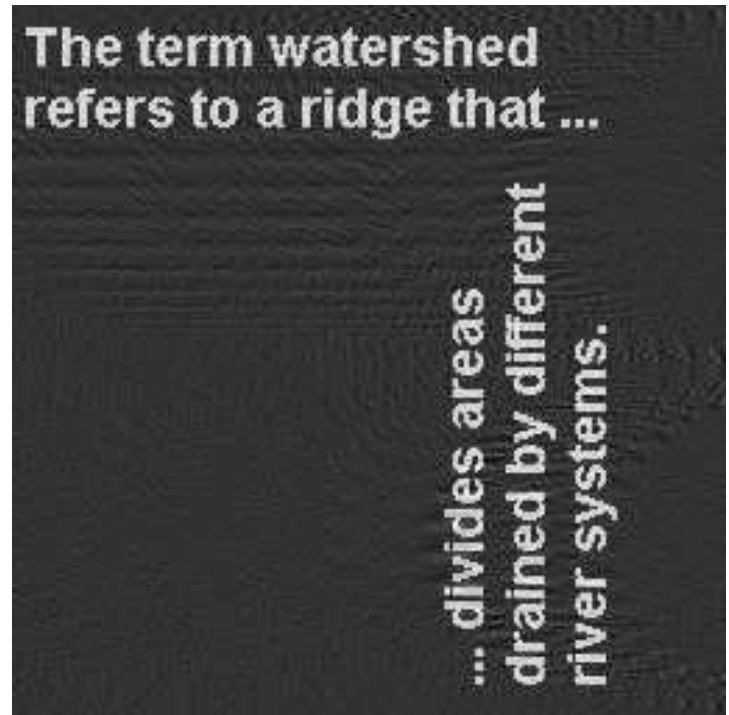
1024 region, 15 iterations



Numerical Examples



Every pixel, 79 iterations



Computational Issues

- Choosing a regularization parameter
- Fast matrix-vector multiplication
should scale well with number of regions
- Preconditioning
especially when using a lot of regions, or all pixel
information
- How to get motion information?

Preconditioning using Kronecker Products

Using the model:
$$A = \sum_{k=1}^p I_k A_k$$

- Each A_k can be decomposed as:

$$A_k = \sum_{j=1}^r C_j^{(k)} \otimes D_j^{(k)}$$

- Therefore, our model for A is:

$$A = \sum_{k=1}^p I_k \left(\sum_{j=1}^r C_j^{(k)} \otimes D_j^{(k)} \right)$$

The End!

- It is possible to efficiently implement iterative methods for non-uniform motion blurs.
- More information provides substantially better restorations.
- However, more information results in slower convergence of iterative methods.

www.mathcs.emory.edu/~nagy