Least Squares Updating for Kronecker Products

Raul Platero
Mathematics and Computer Science
Emory University
Atlanta, GA, USA

James Nagy, Emory University

Outline

Image Deblurring

Image deblurring-

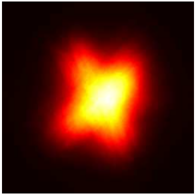
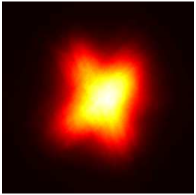
Given blurred image : 

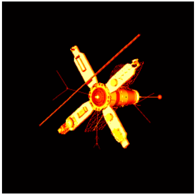
Image Deblurring

Image deblurring-

Given blurred image :



Compute estimate of true image:



Mathematical Model

General mathematical model for image formation:

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

where

- $\mathbf{b}$ = vector representing observed image
- $\mathbf{x}$ = vector representing true image
- A = matrix defining blurring operation
- η = unknown additive noise

Approximation of Blurring Matrix

The blurring matrix is defined by:

$$K = A_1 \otimes B_1 + A_2 \otimes B_2 + \cdots + A_n \otimes B_n \tag{1}$$

Approximation of Blurring Matrix

The blurring matrix is defined by:

$$K = A_1 \otimes B_1 + A_2 \otimes B_2 + \cdots + A_n \otimes B_n \qquad (1)$$

Which can be approximated using only the first two terms

$$K \approx A_1 \otimes A_2 + B_1 \otimes B_2 \,. \qquad (2)$$

Approximation of Blurring Matrix

The blurring matrix is defined by:

$$K = A_1 \otimes B_1 + A_2 \otimes B_2 + \cdots + A_n \otimes B_n \quad (1)$$

Which can be approximated using only the first two terms

$$K \approx A_1 \otimes A_2 + B_1 \otimes B_2 \,. \quad (2)$$

Second term can be approximated by a rank-one matrix[1]

$$K \approx A = A_1 \otimes A_2 + \mathbf{w}\mathbf{z}^T, \quad (3)$$

where $\mathbf{w} = \mathbf{w}_1 \otimes \mathbf{w}_2$ and $\mathbf{z} = \mathbf{z}_1 \otimes \mathbf{z}_2$ and are column vectors.

[1] M. Rezghi, S.M. Hosseini, and L. Elden, *Best Kronecker product approximation of the blurring operator in three dimensional image restoration problems* ,SIAM J. Matrix Anal. Appl. Vol. 35, No. 3, pp. 10861104

Tikhonov Regularization

Least squares problem

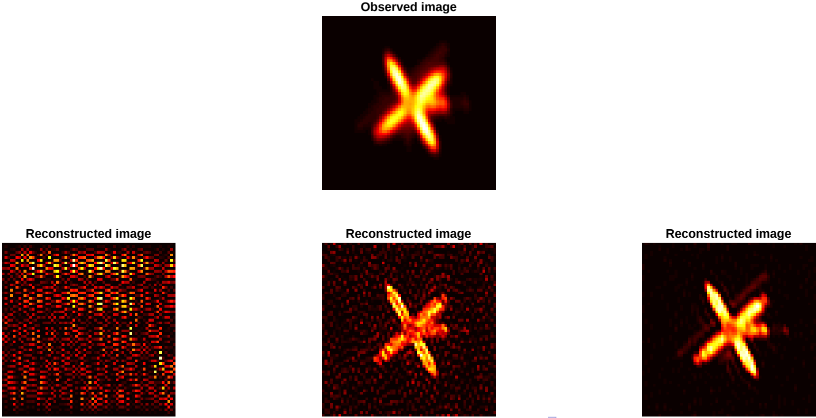
$$\min_{\mathbf{x}} ||A\mathbf{x} - \mathbf{b}||_2^2$$

Damped least squares problem

$$\min_{\mathbf{x}} \left\{ ||A\mathbf{x} - \mathbf{b}||_2^2 + \lambda^2 ||\mathbf{x}||_2^2 \right\} . \quad (4)$$

The regularization parameter λ controls the smoothness of the solution.

Figure: Observed image, along with three reconstructed images where $\lambda = 0$, $\lambda = \lambda/1000$, and $\lambda = \lambda * 0.6$ respectively



Tikhonov Regularization

Damped least squares problem (equation (4)) is reformulated

$$\min_{\mathbf{x}} \left\| \begin{bmatrix} A \\ \lambda I \end{bmatrix} \mathbf{x} - \begin{bmatrix} \mathbf{b} \\ \mathbf{0} \end{bmatrix} \right\|_2^2 . \qquad (5)$$

Now, if we combine this equation with the approximation of the blurring matrix

$$\min_{\mathbf{x}} \left\| \begin{bmatrix} A_1 \otimes A_2 + \mathbf{w}\mathbf{z}^T \\ \lambda I \end{bmatrix} \mathbf{x} - \begin{bmatrix} \mathbf{b} \\ \mathbf{0} \end{bmatrix} \right\|_2^2 . \qquad (6)$$

LSQR

$$\min_{\mathbf{x}} \left\| \begin{bmatrix} A_1 \otimes A_2 + \mathbf{w}\mathbf{z}^T \\ \lambda I \end{bmatrix} \mathbf{x} - \begin{bmatrix} \mathbf{b} \\ \mathbf{0} \end{bmatrix} \right\|_2^2$$

Use LSQR to compute solution

QR Factorization

QR factorization (or decomposition):
If $A \in \mathbb{R}^{m \times n}$, then there exists matrices Q and R such that

$$A = QR$$

where $Q \in \mathbb{R}^{m \times m}$, $Q^T Q = I$ (i.e Q is an orthogonal matrix) and $R \in \mathbb{R}^{m \times n}$.

Givens Rotations

Givens rotations (or plane rotations):
A Givens rotation is a matrix that represents a clockwise rotation by an angle θ.

$$G = \begin{bmatrix} c & s \\ -s & c \end{bmatrix} = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix}$$

In general:

$$G_{ij}(\theta) = \begin{bmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & c & & s & & \\ & & & \ddots & & & \\ & & -s & & c & & \\ & & & & & \ddots & \\ & & & & & & 1 \end{bmatrix}$$

Example

Givens rotations example:

$$A = \begin{bmatrix} x & x & x \\ x & x & x \\ x & x & x \end{bmatrix}$$

Note

- zero entry rotated into a zero entry will remain zero

  non-zero entry rotated to a zero entry will change the zero entry into a non-zero entry

Example

Givens rotations example:

$$A = \begin{bmatrix} x & x & x \\ x & x & x \\ x & x & x \end{bmatrix}$$

Example

Givens rotations example:

$$A = \begin{bmatrix} x & x & x \\ x & x & x \\ x & x & x \end{bmatrix}$$

$$G_{23}(\theta_1)^T A = \begin{bmatrix} x & x & x \\ \bar{x} & \bar{x} & \bar{x} \\ 0 & \bar{x} & \bar{x} \end{bmatrix}$$

Example

$$G_{12}(\theta_2)^T G_{23}(\theta_1)^T A = \begin{bmatrix} \bar{x} & \bar{x} & \bar{x} \\ 0 & \bar{x} & \bar{x} \\ 0 & \bar{x} & \bar{x} \end{bmatrix}$$

Example

$$G_{12}(\theta_2)^T G_{23}(\theta_1)^T A = \begin{bmatrix} \bar{x} & \bar{x} & \bar{x} \\ 0 & \bar{x} & \bar{x} \\ 0 & \bar{x} & \bar{x} \end{bmatrix}$$

$$G_{23}(\theta_3)^T G_{12}(\theta_2)^T G_{23}(\theta_1)^T A = \begin{bmatrix} \bar{x} & \bar{x} & \bar{x} \\ 0 & \bar{x} & \bar{x} \\ 0 & 0 & \bar{x} \end{bmatrix}$$

Example

$$G_{12}(\theta_2)^T G_{23}(\theta_1)^T A = \begin{bmatrix} \bar{x} & \bar{x} & \bar{x} \\ 0 & \bar{x} & \bar{x} \\ 0 & \bar{x} & \bar{x} \end{bmatrix}$$

$$G_{23}(\theta_3)^T G_{12}(\theta_2)^T G_{23}(\theta_1)^T A = \begin{bmatrix} \bar{x} & \bar{x} & \bar{x} \\ 0 & \bar{x} & \bar{x} \\ 0 & 0 & \bar{x} \end{bmatrix}$$

Because the product of orthogonal matrices is an orthogonal matrix, we let $Q = G_{23}(\theta_1)G_{12}(\theta_2)G_{23}(\theta_3)$, and we obtain

$$Q^T A = R \implies A = QR.$$

Updating Problem

Updating problem:
Given a solution to a mathematical problem, efficiently compute a new solution when the problem is slightly modified.

Updating Problem

Updating problem:
Given a solution to a mathematical problem, efficiently compute a new solution when the problem is slightly modified.

- appending a row: Given QR factorization of A, compute QR factorization of
$$\tilde{A} = \begin{bmatrix} A \\ \mathbf{u}^T \end{bmatrix}$$
- appending a column: Given QR factorization of A, compute QR factorization of
$$\tilde{A} = \left[\begin{array}{c|c} A & \mathbf{u}^T \end{array}\right].$$
- adding a rank-one matrix: Given QR factorization of A, compute QR factorization of
$$\tilde{A} = A + \mathbf{w}\mathbf{z}^T,$$
where $\mathbf{w}$, $\mathbf{z}$ are column vectors

Kronecker Product

Kronecker product: generalized outer product that results in a block matrix.

$$K = A \otimes B = \begin{bmatrix} a_{11}B & a_{12}B & \dots & a_{1n}B \\ a_{21}B & a_{22}B & \dots & a_{2n}B \\ \vdots & \vdots & & \vdots \\ a_{m1}B & a_{m2}B & \dots & a_{mn}B \end{bmatrix}$$

Example

If $A = \begin{bmatrix} 1 & 2 & 3 \\ 3 & 4 & 5 \end{bmatrix}$ and $B = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, then

Example

If $A = \begin{bmatrix} 1 & 2 & 3 \\ 3 & 4 & 5 \end{bmatrix}$ and $B = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, then

$$A \otimes B = \begin{bmatrix} 1B & 2B & 3B \\ 3B & 4B & 5B \end{bmatrix} = \begin{bmatrix} 1 & 0 & 2 & 0 & 3 & 0 \\ 0 & 1 & 0 & 2 & 0 & 3 \\ 3 & 0 & 4 & 0 & 5 & 0 \\ 0 & 3 & 0 & 4 & 0 & 5 \end{bmatrix}$$

Properties

Property 1

$$(A \otimes B)^T = A^T \otimes B^T \tag{7}$$

Property 2

If A and B are invertible, then $A \otimes B$ is invertible and

$$(A \otimes B)^{-1} = A^{-1} \otimes B^{-1} \tag{8}$$

Property 3

If A and B are orthogonal, then $A \otimes B$ is also orthogonal. (9)

Property 4

$$(A \otimes B)(C \otimes D) = AC \otimes BD \tag{10}$$

Preconditioner

What is a preconditoner?

Preconditioner

What is a preconditoner?
application of a transformation to make a problem more suitable for solving methods

Rank-one Update

Given QR factorization of A, compute QR factorization of

$$\tilde{A} = A + \mathbf{w}\mathbf{z}^T,$$

where $\mathbf{w}$, $\mathbf{z}$ are column vectors

Rank-one Update

Given QR factorization of A, compute QR factorization of

$$\tilde{A} = A + \mathbf{w}\mathbf{z}^T,$$

where $\mathbf{w}$, $\mathbf{z}$ are column vectors

$$\begin{aligned}
\tilde{A} &= A + \mathbf{w}\mathbf{z}^T \\
&= QR + \mathbf{w}\mathbf{z}^T \\
&= Q[R + Q^T\mathbf{w}\mathbf{z}^T] \\
&= Q[R + \overline{Q}\,\overline{Q}^T Q^T\mathbf{w}\mathbf{z}^T] \\
&= Q\overline{Q}[\overline{Q}^T R + c\mathbf{e}_1\mathbf{z}^T]
\end{aligned}$$

Rank-one Update

$$R = \begin{bmatrix} x & x & \dots & x & x \\ & x & \ddots & & \vdots \\ & & \ddots & & x \\ & & & & x \end{bmatrix}$$

Rank-one Update

$$R = \begin{bmatrix} x & x & \dots & x & x \\ & x & \ddots & & \vdots \\ & & \ddots & & x \\ & & & & x \end{bmatrix}$$

$$\overline{Q}^T R = H = \begin{bmatrix} x & x & \dots & x & x \\ x & x & & & \vdots \\ & x & \ddots & x & x \\ & & \ddots & x & x \end{bmatrix}$$

Rank-one Update

$$
\begin{aligned}
&= Q\overline{Q}[\overline{Q}^T R + c\mathbf{e}_1\mathbf{z}^T] \\
&= Q\overline{Q}[H] \\
&= Q\overline{Q}\hat{Q}\tilde{R} \\
&= \tilde{Q}\tilde{R},
\end{aligned}
$$

where $\tilde{Q} = Q\overline{Q}\hat{Q}$

Rank-one Update for Kronecker Products

Rank-one update for Kronecker products:
Suppose we are given a matrix A_1 and A_2 and their corresponding QR factorizations,

$$A = A_1 \otimes A_2 + \mathbf{w}\mathbf{z}^T = (Q_1 \otimes Q_2)(R_1 \otimes R_2) + (\mathbf{w}_1 \otimes \mathbf{w}_2)(\mathbf{z}_1 \otimes \mathbf{z}_2)^T.$$

This problem is restated as

$$A = (Q_1 \otimes Q_2)[(R_1 \otimes R_2) + (Q_1^T \mathbf{w}_1 \otimes Q_2^T \mathbf{w}_2)(\mathbf{z}_1 \otimes \mathbf{z}_2)^T] \quad (11)$$

Rank-one Update for Kronecker Products

$$A = (Q_1 \otimes Q_2)[(R_1 \otimes R_2) + (Q_1^T \mathbf{w}_1 \otimes Q_2^T \mathbf{w}_2)(\mathbf{z}_1 \otimes \mathbf{z}_2)^T]$$

Rank-one Update for Kronecker Products

$$A = (Q_1 \otimes Q_2)[(R_1 \otimes R_2) + (Q_1^T \mathbf{w}_1 \otimes Q_2^T \mathbf{w}_2)(\mathbf{z}_1 \otimes \mathbf{z}_2)^T]$$

$$A = (Q_1 \otimes Q_2)(\bar{Q}_1 \otimes \bar{Q}_2)[(\bar{Q}_1{}^T \otimes \bar{Q}_2{}^T)(R_1 \otimes R_2) + v(\mathbf{e}_1 \otimes \mathbf{e}_1)(\mathbf{z}_1 \otimes \mathbf{z}_2)^T],$$

where v is a scalar.

Rank-one Update for Kronecker Products

$$A = (Q_1 \otimes Q_2)[(R_1 \otimes R_2) + (Q_1^T \mathbf{w}_1 \otimes Q_2^T \mathbf{w}_2)(\mathbf{z}_1 \otimes \mathbf{z}_2)^T]$$

$$A = (Q_1 \otimes Q_2)(\bar{Q}_1 \otimes \bar{Q}_2)[(\bar{Q}_1{}^T \otimes \bar{Q}_2{}^T)(R_1 \otimes R_2) + v(\mathbf{e}_1 \otimes \mathbf{e}_1)(\mathbf{z}_1 \otimes \mathbf{z}_2)^T],$$

where v is a scalar.

$$A = (\tilde{Q}1 \otimes \tilde{Q}_2)[(H_1 \otimes H_2) + v(\mathbf{e}_1 \otimes \mathbf{e}_1)(\mathbf{z}_1 \otimes \mathbf{z}_2)^T] \qquad (12)$$

Preconditioner

What makes a good preconditioner?
Find M that has the following properties:

- M can be computed efficiently.
- Solving linear systems with M and M^T can be done efficiently.
- M has the property that $M^T M \approx A^T A + \lambda^2 I$. Ideally, if $M^T M - (A^T A + \lambda^2 I)$ is a matrix of rank r, then LSQR will converge in at most r iterations[2].

[2] S. Karimi, D. K. Salkuyeh and F. Toutounian, *A preconditioner for LSQR algorithm*, 9J. Appl. Math. and Informatics Vol. 26(2008), No. 1 - 2, pp. 213 - 222

Rank-one Updating Scheme for Preconditioner

$$A = (\tilde{Q}1 \otimes \tilde{Q}_2)[(H_1 \otimes H_2) + v(\mathbf{e}_1 \otimes \mathbf{e}_1)(\mathbf{z}_1 \otimes \mathbf{z}_2)^T] \qquad (13)$$

Consider the singular value decomposition of H_1 and H_2:

$$H_1 = U_1 \Sigma_1 V_1^T \quad \text{and} \quad H_2 = U_2 \Sigma_2 V_2^T .$$

We will use the following as our preconditioner:

$$M = D(V_1 \otimes V_2)^T , \quad \text{where } D = (\Sigma_1^2 \otimes \Sigma_2^2 + \lambda^2 I)^{1/2} .$$

Rank-one Updating Scheme for Preconditioner

$$M = D(V_1 \otimes V_2)^T, \quad \text{where } D = (\Sigma_1^2 \otimes \Sigma_2^2 + \lambda^2 I)^{1/2}.$$

$$\begin{aligned}
M^T M &= [D(V_1 \otimes V_2)^T]^T D(V_1 \otimes V_2)^T \\
&= (V_1 \otimes V_2) D D (V_1 \otimes V_2)^T \\
&= (V_1 \otimes V_2)(\Sigma_1^2 \otimes \Sigma_2^2 + \lambda^2 I)(V_1 \otimes V_2)^T \\
&= (V_1 \otimes V_2)(\Sigma_1^2 \otimes \Sigma_2^2 + \lambda^2 (I_1 \otimes I_2))(V_1 \otimes V_2)^T \\
&= V_1 \Sigma_1^2 V_1^T \otimes V_2 \Sigma_2^2 V_2^T + \lambda^2 V_1 V_1^T \otimes V_2 V_2^T \\
&= V_1 \Sigma_1^T U_1^T U_1 \Sigma_1 V_1^T \otimes V_2 \Sigma_2^T U_2^T U_2 \Sigma_2 V_2^T + \lambda^2 I \\
&= H_1^T H_1 \otimes H_2^T H_2 + \lambda^2 I \\
&= (H_1 \otimes H_2)^T (H_1 \otimes H_2) + \lambda^2 I \\
&= H^T H + \lambda^2 I.
\end{aligned}$$

Rank-one Updating Scheme for Preconditioner

Now if $A = A_1 \otimes A_2 + \mathbf{w}\mathbf{z}^T = \tilde{Q}[H + v(\mathbf{e}_1 \otimes \mathbf{e}_1)\mathbf{z}^T]$ and $H = H_1 \otimes H_2$, we get

$$\begin{aligned} A^T A &= [H^T + v(\mathbf{z}_1\mathbf{e}_1^T \otimes \mathbf{z}_2\mathbf{e}_1^T)]Q^T Q[H + v(\mathbf{e}_1 \otimes \mathbf{e}_1)\mathbf{z}^T] \\ &= H^T H + v(H_1^T\mathbf{e}_1\mathbf{z}_1^T \otimes H_2^T\mathbf{e}_1\mathbf{z}_2^T) \\ &\qquad + v(\mathbf{z}_1\mathbf{e}_1^T H_1 \otimes \mathbf{z}_2\mathbf{e}_1^T H_2) + v^2(\mathbf{z}_1\mathbf{e}_1^T\mathbf{e}_1\mathbf{z}_1^T \otimes \mathbf{z}_2\mathbf{e}_1^T\mathbf{e}_1\mathbf{z}_2^T) \\ &= H^T H + R \end{aligned}$$

where R is the sum of the three remaining rank-one matrices. Now if we add $\lambda^2 I$ to both sides,

$$\begin{aligned} A^T A + \lambda^2 I &= H^T H + \lambda^2 I + R \\ &= M^T M + R \end{aligned}$$

Efficient Preconditioner Criteria

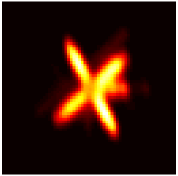
$$M = D(V_1 \otimes V_2)^T, \quad \text{where } D = (\Sigma_1^2 \otimes \Sigma_2^2 + \lambda^2 I)^{1/2}$$

Find M that has the following properties:

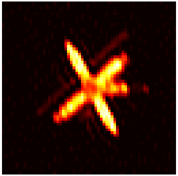
- M can be computed efficiently.
- Solving linear systems with M and M^T can be done efficiently.
- M has the property that $M^T M \approx A^T A + \lambda^2 I$. Ideally, if $M^T M - (A^T A + \lambda^2 I)$ is a matrix of rank r, then LSQR will converge in at most r iterations.

64 $\times$ 64 pixel Satellite Image

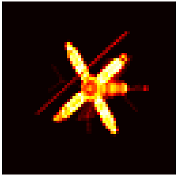
Observed image



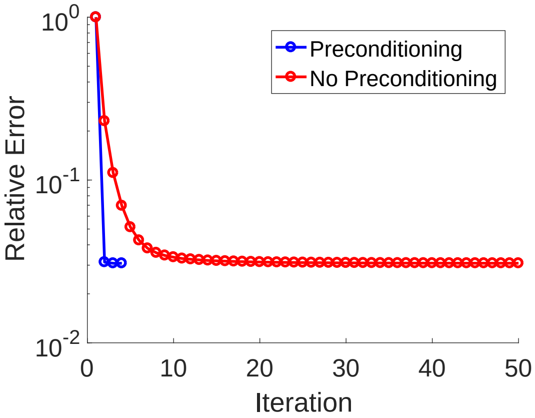
Reconstructed image



True image

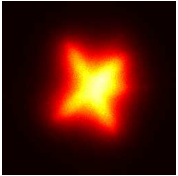


64 × 64 pixel Satellite Image

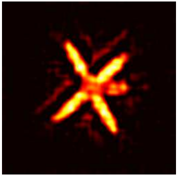


256×256 pixel Satellite Image

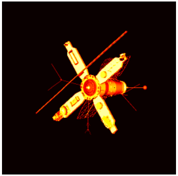
Observed image



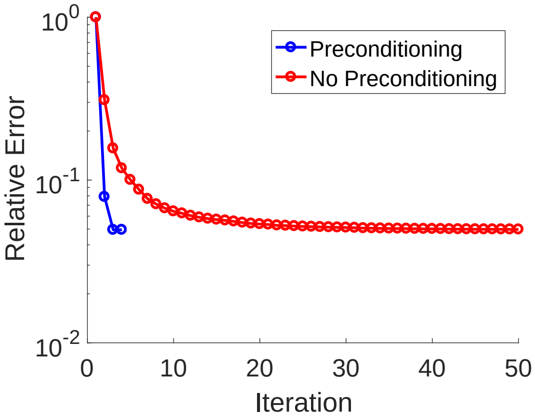
Reconstructed image



True image



256×256 pixel Satellite Image



Time Comparisons

	Preconditioner			No Preconditioner		
size	Time (s)	rel.error	# Itr.	Time (s)	rel.error	# Itr.
$4,096$	0.0707	$8.2326 \cdot 10^{-5}$	3	0.5155	$8.2369 \cdot 10^{-5}$	50
$65,536$	0.5140	$3.5307 \cdot 10^{-4}$	3	11.5040	$3.5505 \cdot 10^{-4}$	50

Conclusion

Remarks:

- Rank-one updating scheme provides an efficient preconditioner for image deblurring problem
- Guaranteed to converge in at most 3 iterations
- Increased speed over benchmark method

Future Work:

- Extending our approach to rank-k modifications, where $k > b$
- Comparisons with other fast methods