Non-Iterative Reconstruction of Sparse Images from Limited Data

Professor Andrew E. Yagle
Dept. of Electrical Engineering and Computer Science
The University of Michigan
Ann Arbor, Michigan USA
Presentation Overview

- Why sparse images and non-iterative?
- Review of “valid” 2-d deconvolution
- Valid reconstruction of sparse images [3]:
  1. Valid deconvolution of bandlimited PSF;
  2. Slightly underdetermined reconstruction;
- Valid phase retrieval of sparse images [1]
- Conclusion
Presentation Overview

- Why sparse images and non-iterative?
- Review of “valid” 2-d deconvolution
- Valid reconstruction of sparse images [3]
- Valid phase retrieval of sparse images [1]
- Conclusion
Why are sparse images important?

- **DEF:** 1-d $x(n)$ is **K-sparse** if $x(n)$ is nonzero only at $K$ (unknown) values of index $n$.
- **EX:** $x(n) = \{2,0,0,0,3,0,0,1,0,0,0,0,4,0,0,1,0\}$ (all other values of $x(n)$ are 0) is **5-sparse**.
- Extension to 2-d (images) should be evident.
- **Example:** atoms in X-ray crystallography.
- **Example:** atoms in magnetic resonance force microscopy (MRFM).
What are sparsifiable images?

- $x(n)$ is **sparsifiable** if $x(n) = \sum c(n,m)z(m)$ for some known matrix of basis functions $c(n,m)$ and $z(n)$ is sparse ($x(n)$ sparse in some basis).
- **Example**: $c(n,m)$ are wavelet basis functions.
- **Extension to 2-d (images)** is evident (but this requires 4 indices; more if wavelets are used!)
- **Example**: block letters or symbols (next slide).
Using corner detector, we can sparsify block letters:

**Corner detector:** \( y(i,j) = x(i,j) - x(i,j-1) - x(i-1,j) + x(i-1,j-1) \)
Problems with iterative algorithms

- **Does** the algorithm converge at all?
- **How long** does it take to converge?
- To **what** does the algorithm converge?
- What bias is introduced by **stopping**?
- Can take long, unknown time to converge.
- Non-parallelizable in iteration number.
- **Non-iterative**: avoids all of these issues.
Presentation Overview

• Why sparse images and non-iterative?
• Review of “valid” 2-d deconvolution
• Valid reconstruction of sparse images [3]
• Valid phase retrieval of sparse images [1]
• Conclusion
**Complete 2-d convolution**

\[ y(i,j) = \sum \sum h(m,n)x(i-m,j-n). \]

**DFT:** \[ X(k) = \sum x(n)e^{-j2\pi nk/N} \] for \( k \).

**Deconvolution:** \[ X(k_1,k_2) = Y(k_1,k_2)/H(k_1,k_2) \] for \( |H(k_1,k_2)| > 0 \).

\[
\begin{array}{cccccc}
1 & 4 & 1 & 5 & 9 \\
2 & 6 & 5 & 3 & 5 \\
8 & 9 & 7 & 9 & 3 \\
2 & 3 & 8 & 4 & 6 \\
2 & 6 & 4 & 3 & 3 \\
\end{array}
\]

\[
\begin{array}{cccccc}
1 & 5 & 5 & 6 & 14 & 9 \\
3 & 13 & 16 & 14 & 22 & 14 \\
10 & 25 & 27 & 24 & 20 & 8 \\
10 & 22 & 27 & 28 & 22 & 9 \\
4 & 13 & 21 & 19 & 16 & 9 \\
2 & 8 & 10 & 7 & 6 & 3 \\
\end{array}
\]

\[ x(i,j) * h(i,j) = y(i,j) = h(i,j) * x(i,j) \]

\[ x(i,j) \]

\[ h(i,j) \]

\[ y(i,j) \]
Valid 2-d convolution

\[ y(i,j) = \sum \sum h(m,n) x(i-m,j-n). \text{ BUT: no image edge info used.} \]

Deconvolution: Underdetermined—need info about image.

\[
\begin{array}{cccccc}
1 & 4 & 1 & 5 & 9 \\
2 & 6 & 5 & 3 & 5 \\
8 & 9 & 7 & 9 & 3 \\
2 & 3 & 8 & 4 & 6 \\
2 & 6 & 4 & 3 & 3 \\
\end{array}
\]

\[
\begin{array}{cccccc}
* & * & * & * & * & * \\
* & 13 & 16 & 14 & 22 & * \\
* & 25 & 27 & 24 & 20 & * \\
* & 22 & 27 & 28 & 22 & * \\
* & 13 & 21 & 19 & 16 & * \\
* & * & * & * & * & * \\
\end{array}
\]

\[ \ast = \text{unknown values} \]
What 2-d convolution does to images

valid deconvolution
also downsampled:
Deconvolution: undo
Formulation of Basic Problem

• **GIVEN:** Underdetermined linear system.
• \( y = Hx \). **Data** \( y \): known M-vector.
• **Solution** \( x \): unknown N-vector.
• Infinite number of solutions, since \( M < N \).
• Compute the unique K-sparse solution \( x \).
• Assume it exists (*a priori* knowledge).
Why not use $\ell_1$ minimization?

- Minimize $\sum |x(n)|$ such that $y = Hx$ (constrained)
- Min $||y - Hx||^2 + \lambda \sum |x(n)|$ (LASSO functional)
- Min $||y - Hx||_1 + \lambda \sum |x(n)|$ (LAD functional)
- Min $||y - Hx||^2 + \lambda \sum |x(n) - x(n-1)|$ (total variation)

- Minimizing $\sum |x(n)|$ tends to sparsify $x(n)$ IF
- $H$ is a random matrix (or other conditions)
Why not use $\ell_1$ minimization?

- Minimize functionals using: gradient, or linear programming or coordinate descent (all iterative methods; may take long time)

- BUT: H matrix in image reconstruction is NOT a random matrix! $\ell_1$ doesn’t work!
Alternative to $\ell_1$ norm minimization

• Suppose $x(n)$ has length $N$ and is $K$-sparse.
• Then there is an indicator function $s(n)$ s.t.:
  \[ s(n)x(n) = 0 \] and DFT $S(k)$ has length $= K+1$.

• \textbf{DFT:} $X(k) = \sum x(n)e^{-j2\pi nk/N}$ for $N$ values of $k$.

• Locations of nonzero $x(n)$: \{n$_1$, n$_2$, n$_3$... n$_K$\}.
• Polynomial $\sum S(k)z^k$ has $K$ zeros at locations \{exp(-j2\pi n$_1$/N)...exp(-j2\pi n$_K$/N)\}. 

Example: Indicator function

- \( x(n) = \{0, 0, 2, 0, 3, 0, 0, 0\} \). Length=8; 2-sparse.
- \( s(n) = \{(1+j)/4, 0.177j, 0, 0.073, 0, -0.177j, (1-j)/4, 0.427\} \)
- \( S(k) = \{1, 1+j, j, 0, 0, 0, 0, 0\} \). Roots: \{-j, -1\}.

- \( x(n) \ K\text{-sparse} \rightarrow s(n)x(n) = 0 \rightarrow S(k)X(k) = 0. \)
- K+1 unknowns \( S(k) \) impose sparsity on \( x(n) \).
- Use this in the following NEW algorithms.
Presentation Overview

• Why sparse images and non-iterative?
• Review of “valid” 2-d deconvolution
• Valid reconstruction of sparse images [1/3]:
  1. Valid deconvolution of bandlimited PSF;
  2. Slightly underdetermined reconstruction;
• Valid phase retrieval of sparse images [1]
• Conclusion
1. Bandlimited matrix $H$: Needs

- **ASSUME**: Each row of $H$ *bandlimited* to $M$.
- **THEN**: $K$-sparse $x(n)$ computed by solving:
  - (1) $M \times M$ system to compute $X(k), 0 \leq k \leq M/2$;
  - (2) $K \times K$ Toeplitz to compute $s(n)$, locations $n_i$
  - (3) $K \times K$ system to compute $x(n)$ values.

- **APPLICATION**: Deconvolving bandlimited point-spread functions (PSF) $h(i,j)$ from $x(i,j)$.
- **EXAMPLE**: 2-d Gaussian PSF.
1. Bandlimited matrix $H$: Procedure

- Let $H(i,k) = \sum h(i,n)e^{-j2\pi nk/N} = \text{DFT\{rows of } h(i,j)\text{\}}$
- $y(i) = \sum h(i,j)x(j) = \sum H(i,k) * X(k)/N$ (Parseval).
- $H(i,k)$ rows bandlimited to $M$ implies $H(i,k)=0$ for $M/2<k<N-M/2$ for each row $#i$ of $H(i,k)$.

- Solve $M \times M$ linear system for $X(k)$, $0 \leq k \leq M/2$.
- Solve $K \times K$ Toeplitz equations $S(k)*X(k)=0$.
- Solve $K \times K$ linear system for values of $x(n)$. 
1. Bandlimited matrix $H$: Example

- **IMAGE** $x(i,j)$: $72 \times 72$ and sparsifiable.
- **PSF**: 2-d Gaussian $h(i,j) = 0.98^{(i^2+j^2)}$ bandlimited
- **DATA**: $y(i,j) = h(i,j) \ast x(i,j)$ & downsampled, since $h(i,j)$ bandlimited implies $y(i,j)$ also bandlimited.

- **GOAL**: Compute $x(i,j)$ from downsampled $y(i,j)$.
- **NOTE**: Clearly underdetermined linear problem (see next slide for numerical details).
1. Bandlimited matrix $H$: Example

- **Unknowns**: $72^2 = 5184$ pixels $x(i,j)$.
- **Knowns**: $36^2 = 1296$ values $y(i,j)$ of downsampled $h(i,j) \times x(i,j)$ (cyclic $*$).

- **Side information**: $x(i,j)$ sparsifiable by $z(i,j) = x(i,j) - x(i,j-1) - x(i-1,j) + x(i-1,j-1)$.
- $y(i,j)$ known at $19 \times 19$ lowest wavenumbers.
- **NEED**: sparsified $z(i,j)$ is $10^2 - 1 = 99$-sparse.
1. Bandlimited matrix H: Example

- **Computational requirements:**
  - Null of $100 \times 100$ Toeplitz-block-Toeplitz;
  - $72 \times 72$ 2-d DFT of $10 \times 10$ rearrangement of null vector of Toeplitz-block-Toeplitz;

- Solution of $98 \times 98$ to compute $z(i,j)$ values;

- Deconvolve corner detector: $z(i,j) \rightarrow x(i,j)$. Requires knowledge of 2 edges of $x(i,j)$.
1. Bandlimited matrix $H$: Example

Blurred and downsampled image data.

2-d Gaussian blurring PSF $h(i,j)$

Can you guess the original image?
1. Bandlimited matrix $H$: Example

Reconstructed sparsified image $z(i,j)$

Reconstructed original image $x(i,j)$
Presentation Overview

- Why sparse images and non-iterative?
- Review of “valid” 2-d deconvolution
- Valid reconstruction of sparse images [2/3]
  1. Valid deconvolution of bandlimited PSF;
  2. Slightly underdetermined reconstruction;
- Valid phase retrieval of sparse images [1]
- Conclusion
2. Slightly Underdetermined H: Needs

- **ASSUME**: $y = Hx$ only *slightly underdetermined*:
- $N > (N - M)(K) = (#\text{underdetermined})(#\text{nonzero } x(n))$
- Actually need $N > (N - M + 1)(K + 1)$ (counting issues).

- **APPLICATION**: Valid deconvolution of PSFs that are *spatially varying*; other underdetermined linear transformations of $K$-sparse signals.
2. Slightly Underdetermined H: Procedure

- **THEN**: \( y = Hx \rightarrow [H - y][x^T 1]^T = 0 \) (include \( y \) in \( H \)).
- Rename: \( H = [H - y] \) and \( x^T = [x^T 1]^T \) in the sequel.

- Now \( y = Hx \) has become \( 0 = Hx \). Using Parseval:
  - \( 0 = Hx = \sum h(i,j)x(j) = \sum H(i,k)^*X(k) = Hx \) (DFT of \( H, x \)).

- \( x = Gw \) where \( G \) spans right nullspace of \( H \).
2. Slightly Underdetermined H: Procedure

- **BUT**: $G$ and vector $w$ have dimensions $N-M$.

- **SO**: $S(k) \sum G(i,k)w(k) = 0$ is $N$ equations in $(N-M)$ unknowns $w(k)$ and $K$ unknowns $S(k)$. Becomes: $N$ **linear** equations in $(N-M)(K)$ unknowns $S(k_1)w(k_2)$. 
2. Slightly Underdetermined $H$: Example

- **IMAGE** $x(i,j): 30 \times 30$; sparsifiable to 12-sparse.
- **LINEAR TRANSFORMATION $H$**: Random $832 \times 900$ matrix times inverse corner detector.
- **DATA**: $y = Hx$ where $x(i,j)$ unwrapped by rows.

- **GOAL**: Compute $x$ from $y$. Underdetermined.
- **NEED**: $N > (N-M+1)(K+1)$ not $(N-M)K$ (counting)
- **HAVE**: $900 > 897 = (900-832+1)(12+1)$ so can do it.
2. Slightly Underdetermined H: Example

- Computational requirements:
  - Null of $900 \times 897$ Toeplitz-blocks matrix;
  - Rearrange null vector into $69 \times 13$ matrix;
  - Rank-one factorization of this matrix;
  - $900$-point DFT of length $= 13$ rank-one factor;
  - $12$ values of this were zero; these specified locations of nonzero elements of sparsified $x$. 
2. Slightly Underdetermined H: Example

832×900 is only *slightly* underdetermined linear system. Can’t we just use least-squares to find 12 nonzero values?

This is the least-squares solution. Find 12 nonzero values:

This is the least-squares SOLUTION, *not* the data! Only 12 of these pixels are supposed to be nonzero!

Can you pick out the 12?

HINT: They aren’t the brightest pixels you see.
2. Slightly underdetermined $H$: Example

Reconstructed sparsified image $z(i,j)$

Reconstructed original image $x(i,j)$
Presentation Overview

• Why sparse images and non-iterative?
• Review of “valid” 2-d deconvolution
• Valid reconstruction of sparse images [3/3]

1. **Valid deconvolution of bandlimited PSF**;
2. **Slightly underdetermined reconstruction**;
3. **Kronecker-product-based reconstruction**.

• Valid phase retrieval of sparse images [1]
• **Conclusion**

- **GIVEN**: $y = Hx$ where $H = H_1 \times H_2$
- $H_1 \times H_2 = \text{Kronecker product of } H_1 \& H_2$.
- $y$ is an $M^2$ vector & $x$ is an $N^2$ vector.
- $x$ is $(M-1)$ sparse or less; *very* sparse.

- **GOAL**: Compute $x$ from $y$. Note $M^2 \ll N^2$.
- **ADVANTAGE**: Much less computation.
3. Kronecker Product H: Review

\[
\begin{bmatrix}
1 & 2 \\
3 & 4
\end{bmatrix}
\times
\begin{bmatrix}
5 & 6 \\
7 & 8
\end{bmatrix}
= 
\begin{bmatrix}
5 & 6 & 10 & 12 \\
7 & 8 & 14 & 16 \\
15 & 18 & 20 & 24 \\
21 & 24 & 28 & 32
\end{bmatrix}
\]

Relevant properties of the Kronecker product here:

\[(A \times B)(C \times D) = (AC) \times (BD).\]

\[\text{vec}(AXB) = (B^T \times A)\text{vec}(X).\]

\[\text{vec}(Y) = (H_1 \times H_2)\text{vec}(X) \text{ means } Y = H_2 X H_1^T.\]
3. Kronecker Product H: Applications

- **2-d deconvolution with separable 2-d PSF.**
  Example: 2-d Gaussian PSF is separable.

- **2-d reconstruction from partial DFT data.**
  2-d DFT is Kronecker product of 1-d DFTs.

- **Image sparsifiable by separable 2-d transform.**
  2-d wavelet transform is usually separable.

- $y = (H_1 \times H_2)x$ same as $Y = H_2XH_1^T$ where:
- $\text{vec}(X) = x$ and $\text{vec}(Y) = y$. $X$ is $N \times N$; $Y$ is $M \times M$.
- SVD’s: $H_1 = U_1 S_1 V_1$ and $H_2 = U_2 S_2 V_2$ (identical?)

- $Y = H_2XH_1^T = (U_2 S_2 V_2)X(U_1 S_1 V_1)^T$ becomes
- $V_2 XV_1^T = (S_2)^{-1}U_2^TYU_1(S_1)^{-1}$ computed from $y$.  


- $V_2XV_1^T$ is $M \times N$ but has rank at most $M-1$, since at most $M-1$ entries of $X$ are nonzero.

- Can have more than $M-1$ nonzero entries of $X$ if some lie on same row or column: Need at most $M-1$ nonzero-containing rows and $M-1$ columns.

- Null $n$ of $V_2XV_1^T$ is same as null of $XV_1^T$. 

• $i^{th}$ row of $X$ all zeros $\rightarrow i^{th}$ element of $XV_1^Tn=0$.

• $(i,j)^{th}$ element of $X$ nonzero $\rightarrow (j^{th}$ row of $V_1^T)n=0$.

• Zeros of $V_1^Tn \rightarrow X$ columns with nonzero element.

• Repeat with $(V_2XV_1^T)^T \rightarrow X$ rows with nonzeros.
3. Kronecker Product H: Example

- $256 \times 256$ sparse image $X$ with 21 nonzero pixels.
- $2^2 = 484 \times 65536 = 256^2$ random system matrix $H$.
- $H =$ Kronecker product of two $22 \times 256$ matrices.

**Goal:** Compute unknown 65536-element $x$ from known 484-element $y$. Know that $x$ is 21-sparse.
3. Kronecker Product H: Example

- **Computational requirements:**
- SVD’s of two $22 \times 256$ matrices (maybe identical). Can precompute these for a given imaging system.

- Left and right nulls of $22 \times 22$ data matrix.
- Compute $V_1^T n$ from null $n$; repeat for $V_2^T n$.
- *Very* little computation for this big a problem!
3. Kronecker Product $H$: Example

Original data arranged into a 22 by 22 array. Can you guess locations of 21 nonzero pixels?
3. Kronecker Product $H$: Example

Locations of possible nonzero pixels

Original image with 21 nonzero pixels
Presentation Overview

• **Why sparse images and non-iterative?**
• **Review of “valid” 2-d deconvolution**
• **Valid reconstruction of sparse images [3]**
• **Valid phase retrieval of sparse images [1]**
• **Conclusion**
4. Phase Retrieval of Sparse Images

• **GIVEN**: 2-d DFT Fourier magnitude $|X(k_1,k_2)|$.

• **BUT**: Don’t have Fourier phase $\arg\{X(k_1,k_2)\}$.

• **GOAL**: Compute $x(i,j)$ from 2-d DFT $|X(k_1,k_2)|$.

• **HENCE**: “Phase retrieval” (from magnitude).

• **APPLICATIONS**: X-ray crystallography; optics (measure only diffraction patterns); astronomy.
4. Phase Retrieval: Problem set-up

- **ASSUME**: \( x(i,j) \) is either *sparse* or *sparsifiable* by an LTI transformation (e.g., corner detector).

- **GIVEN**: Autocorrelation \( y(i,j) = \text{DFT}^{-1}\{|X(k_1,k_2)|^2\} \)

- **UNWRAP**: 2-d problem to 1-d using either:
  - Kronecker transformation: substitute \( y = x^M \) in:
    \[
    Y(x,y) = X(x,y)X(x^{-1},y^{-1}) \rightarrow Y(x,x^M) = X(x,x^M)X(x^{-1},x^{-M});
    \]
  - Agarwal-Cooley convolution (residue # system).
4. Phase retrieval: Ambiguities

- **SCALE FACTOR**: Solution $x(n) \rightarrow -x(n)$ also.
- **TRANSLATION**: Solution $x(n) \rightarrow x(n-d)$ also.
- **REVERAL**: Solution $x(n) \rightarrow x(-n)$ also a solution.
- These extend to 2-d case in obvious fashion.

- All of these will appear in the sparse algorithm!
4. Phase Retrieval: Problem set-up

- **GOAL**: Solve 1-d phase retrieval; rewrap to 2-d.

- **SOLVE**: \( y(n) = y(-n) = x(n) \cdot x(-n) = \sum x(i) x(i+n) \).

- If \( x(n) \) sparsifiable to \( z(n) \) by \( x(n) = h(n) \cdot z(n) \) for some known function \( h(n) \) (e.g., corner detector):

  - \( y(n) = [h(n) \cdot h(-n)] \cdot [z(n) \cdot z(-n)] \). Then deconvolve \( h(n) \cdot h(-n) \) from \( y(n) \rightarrow \) sparse problem \( z(n) \cdot z(-n) \).
4. Phase Retrieval: Algorithm [1/5]

- **ASSUME**: Each $y(n)$ is a single $x(i)x(i+n)$ term. True if $x(n)$ is sparse and sampling of $x(n)$ fine.
- **THEN**: Can replace nonzero $x(n)$ and $y(n)$ with 1 to find locations of nonzero $x(n)$. Then actual $x(n)$ computed from rank-one decomposition of $r(n)$.

- **GIVEN**: $r(n)=r(-n)$ support $-M \leq n \leq M$ for some $M$.
- **Initialize**: $x(0)=x(M)=1$ since $r(M)=1$.
- **NOTE**: This resolves translation ambiguity!
4. Phase Retrieval: Algorithm [2/5]

• **Recursion#1**: Let $n_1$ be next *largest* $n$ s.t. $r(n) \neq 0$.

• Either $x(n_1)$ or $x(M-n_1) \neq 0$, but which one?

• Can’t tell at this point—this is *reversal ambiguity*!

• Pick, without loss of generality, $x(n_1) \neq 0$. 

- **Recursion #2**: Let $n_2$ be next *largest* $n$ s.t. $r(n) \neq 0$.
- Either $x(n_2)$ or $x(M-n_2) \neq 0$, but which one?
- Now *can* tell! Check the following two cases:
  - If $x(n_2) \neq 0$, then $r(n_1-n_2) \neq 0$ and $r(|M-n_1-n_2|) = 0$.
  - If $x(M-n_2) \neq 0$, then $r(|M-n_1-n_2|) \neq 0$ and $r(n_1-n_2) = 0$.
- This specifies which of $x(n_2)$ or $x(M-n_2) \neq 0$. 

- **Recursion #3**: Let $n_3$ be next largest $n$ s.t. $r(n)\neq 0$.
- Either $x(n_3)$ or $x(M-n_3)\neq 0$, but which one?
- Suppose $x(n_2)$, not $x(M-n_2)$, was $\neq 0$. Then:
  - If $x(n_3)\neq 0$, then $r(n_1-n_3)\neq 0$ and $r(n_2-n_3)\neq 0$.
  - If $x(M-n_3)\neq 0$, $r(|M-n_3-n_1|)\neq 0$ and $r(M-n_3-n_2)=0$.
- This specifies which of $x(n_3)$ or $x(M-n_3)\neq 0$.

- **NOTE**: As recursions progress, more checks.
4. Phase Retrieval: Algorithm [5/5]

- **AT END**: Have all indices \( n_j \) at which \( x(n_j) \neq 0 \).
- **THEN**: Each \( r(n_i) = x(n_j)x(n_j + n_i) \) for a known \( n_j \).

- **SO**: Form symmetric matrix of nonzero \( r(n_i) \). Rank-one factorization \( \rightarrow \) actual \( x(n_j) \) values.

- **BUT**: Sign ambiguity in outer product: This is *Scale factor ambiguity*!
4. Phase Retrieval: Example #1

- Sparse 100×99 image; 16 nonzero pixels.
- **GIVEN**: 100×99 cyclic autocorrelation; no image support constraint; just sparse.
- **GOAL**: reconstruct sparse 100×99 image.

- **NOTE**: Agarwal-Cooley used to map to 1-d.
- **NOTE**: Cyclic autocorrelation is 240-sparse; 16 values x(n)≠0→16(16-1)=240 values r(n)≠0.
4. Phase Retrieval: Example #1

Autocorrelation (zeroth lag suppressed)  
Reconstructed 16-sparse image
4. Phase Retrieval: Example #2

- **Sparsifiable** (by corner detector) 30×29 image.
- **GIVEN**: 30×29 cyclic autocorrelation of image; no image support constraint; just sparsifiable.
- **GOAL**: reconstruct sparsifiable 30×29 image.

- **NOTE**: Agarwal-Cooley used to map to 1-d.
- **NOTE**: 1\textsuperscript{st} deconvolve the corner detector from cyclic autocorrelation; then it is 132-sparse: 12 values \(x(n)\neq 0\rightarrow 12(12-1)=132\) values \(r(n)\neq 0\).
4. Phase Retrieval: Example #2

Fourier transform magnitude

Reconstructed sparsifiable image

note no support constraint known
4. Phase Retrieval: Example #2

- **NOTE**: Weird-looking block letter “E.” Why?
  Need to ensure that after deconvolving corner detector: Each $y(n)$ is a single $x(i)x(i+n)$ term.

- **NOTE**: In a realistic-size problem, this is not likely to be an issue (use fine discretization).

- Used small-size problem to illustrate the issue.

- **NOTE**: Do need a small support constraint:
  2 edges of image are row and column of zeros, so can deconvolve corner detector from image.
CONCLUSION

• Non-iterative algorithms are fast: Most of these require only solution of an $M \times M$ linear system.
  
• 1st: For bandlimited valid image deconvolution

• 2nd: For non-bandlimited valid deconvolution with non-separable PSF; valid linear transform

• 3rd: For separable valid linear transforms of very sparse or sparsifiable images; VERY fast.

• Phase retrieval of sparse or sparsifiable images
THANK YOU FOR LISTENING!

- Papers and Matlab code for small examples at: http://www.eecs.umich.edu/~aey/sparse.html

- I would like to thank Jison for his hospitality (and for being such a good Ph.D student!)

- Any questions?