# Non-Iterative Superresolution Phase Retrieval of Sparse Images without Support Constraints

Andrew E. Yagle
Department of EECS, The University of Michigan, Ann Arbor, MI 48109-2122

*Abstract*— **We propose a new non-iterative algorithm for phase retrieval of a sparse image from low-wavenumber values of its Fourier transform magnitude. No image support constraint is needed. The algorithm uses the sparsity of the image autocorrelation to reconstruct it exactly from low-wavenumber Fourier magnitude data (superresolution) using a variation of MUSIC. The sparsity of the image is then used to reconstruct it recursively from its autocorrelation. Applications include X-ray crystallography and astronomy. Three numerical examples illustrate the algorithm.**

*Keywords*— **Superresolution, phase retrieval**
**Phone: 734-763-9810. Fax: 734-763-1503.**
**Email: aey@eecs.umich.edu.** **EDICS: 2-REST.**

## I. INTRODUCTION

### *A. Background*

In many optical problems featuring diffraction and scattering, Fourier phase information is distorted or never acquired. Thus Fourier phase must be computed from magnitude (phase retrieval). X-ray crystallography and astronomy are two such problems.

Many algorithms have been proposed to solve this. Most of them iteratively impose constraints on the Fourier magnitude values and image support. These are known to almost surely uniquely determine the image to some trivial ambiguities, discussed below.

However, high-wavenumber data are often unavailable, for reasons discussed below, and the image support may be unknown. But the image may be sparse (mostly zero-valued), with the locations of its nonzero values unknown. This is the case in X-ray crystallography and astronomy in particular.

### *B. Problem Statement*

The goal is to reconstruct $\{x_n, 0 \leq n \leq M-1\}$ from some low-wavenumber values $\{|X_k|, |k| \leq K^2\}$ of its N-point discrete Fourier transform (DFT)

$$X_k = \sum_{n=0}^{N-1} x_n e^{-j2\pi nk/N}, M \leq N \tag{1}$$

- $x_n$ is known to be K-sparse (K nonzero values);
- The locations $\{n_i, 1 \leq i \leq K\}$ of its nonzero values are unknown (hence no known support constraint);
- The 2-D problem has been unwrapped to a 1-D one using Kronecker or Agarwal-Cooley;
- M is for convenience of presentation:
  - The finite support case N≥2M+1 is solved first;
  - Then extended to the no support case M=N.

The phase retrieval problem in all dimensions is known to have the following three trivial ambiguities:

- **Translation:** If $x_n$ is a solution, then $x_{n-D}$ is also a solution for any integer D. For the no support case $x_{n-D}$ represents a circular shift (viz., (n-D)mod(N));
- **Reversal:** If $x_n$ is a solution, then $x_{N-n}$ is also a solution. The signal can only be reconstructed to a mirror-image ambiguity;
- **Sign:** If $x_n$ is a solution, then $-x_n$ is a solution. If $x_n$ may be complex, then $e^{j\theta}x_n$ is also a solution.

The problem is considered solved when $x_n$ is determined to within these three ambiguities.

The autocorrelation $r_n$ of $x_n$ is defined by

$$r_n = \sum_{i=0}^{M-1} x_i x_{i-n}; \quad |X_k|^2 = \sum_{n=0}^{N-1} r_n e^{-j2\pi nk/N} \tag{2}$$

So knowledge of $r_n$ is equivalent to knowledge of Fourier transform magnitude squared. If there is no support information for $x_n$, $r_n$ is the cyclic autocorrelation–all indices are reduced mod(N).

### *C. Relevant Other Approaches*

Sparse signal reconstruction is often accomplished by finding the signal satisfying all other constraints that has minimum $\ell_1$ norm ($\sum |x_n|$). This can be computed using linear programming.

However, that approach is inapplicable here, since even if all of the Fourier magnitude data were available, this would only determine the autocorrelation $r_n$ of $x_n$, which is only the sum of $\{x_i x_j, i-j=n\}$.

Iterative algorithms such as the hybrid input-output algorithm, converge to the solution if all Fourier magnitude data are available and a support constraint is known. The 1-D solution is not unique, but the 2-D problem unwraps to a 1-D problem with bands of zeros, which does have a unique solution.

However, that approach is also inapplicable here, since although most $x_n$ are zero, the locations $n_i$ of the nonzero values are unknown, so there is no known support constraint. And if only some of the Fourier magnitude data are available, convergence to unique solution is not guaranteed for iterative algorithms.

### *D. New Approach of This Paper*

We solve the problem formulated above in 3 steps:

1. The sparsity of the autocorrelation $r_n$, whose nonzero locations are also unknown, is used to compute $r_n$ from low-wavenumber Fourier magnitudes $\{|X_k|^2, |k| \leq K < N\}$, using a variation of MUSIC;
2. If the problem is 2-D, it is unwrapped to a 1-D problem using the Kronecker transform or Agarwal-Cooley fast convolution (for no known support);
3. The sparse signal $x_n$ is computed recursively from the sparse $r_n$. At each recursion, the locations of already-determined nonzero $x_n$ are used to determine whether a nonzero $x_n$ is located at n or M–n or N–n for the no known support case. Then the $r_n$ due to the newly-determined $x_n$ are all eliminated.

### *E. Organization*

This paper is organized as follows. Section II reviews diffraction scattering theory to explain why only low-wavenumber Fourier transform magnitude data is available in many optics problems. It then presents the superresolution algorithm for reconstructing the sparse autocorrelation from low-wavenumber Fourier magnitude data only. Section III presents the algorithms for recursively reconstructing the sparse signal $x_n$ from the sparse autocorrelaiton $r_n$, first for the finite-support case and then for the no known support case. Section IV presents two illustative numerical examples, one for each support case.

## II. Superresolution and Scattering

The first subsection reviews general diffraction imaging in optics; the second specializes to crystals. The third reviews how MUSIC can be adapted to perform superresolution and reconstruct the sparse autocorrelation from low-wavenumber Fourier magnitude data. We also provide a glossary linking X-ray crystallography terms to signal processing terms.

### *A. Diffraction Imaging*

Consider an object $\{o(x), x \in \mathcal{R}^3\}$ known to be zero outside the sphere $|x| \leq R$ for some finite radius $R$ (so $o(x)$ has compact support) illuminated with a plane wave $\delta(t - \vec{e_I} \cdot x/c)$ in a direction specified by the unit vector $\vec{e_I}$, travelling at wave speed $c$. Taking temporal Fourier transforms to replace time dependence with frequency dependence results in

$$\mathcal{F}_{t\to\omega}\{\delta(t-\vec{e_I}\cdot x/c)\} = \int \delta(t-\vec{e_I}\cdot x/c)e^{-i\omega t}dt$$
$$= e^{-i\omega(\vec{e_I}\cdot x)/c} = e^{-i2\pi(\vec{e_I}\cdot x)/\lambda} \quad \lambda = \text{wavelength} \quad (3)$$

Here wavelength replaces frequency over wave speed.

For each $x \in \mathcal{R}^3$, this plane wave is scattered by $o(x)$, producing a spherically-spreading scattered field. In the direction specified by the unit vector $\vec{e_S}$, the scattered field in the far field (large $|x|$) is

$$e^{-i\frac{2\pi}{\lambda}\vec{e_I}\cdot x} o(x) \frac{1}{4\pi|x|} e^{i\frac{2\pi}{\lambda}\vec{e_S}\cdot x}$$

We now make the *Born approximation*, which is that the scattered field is not further scattered by $o(x)$ at other values of $x$. This amounts to assuming that $|o(x)| << 1$, so that $o(x_1)o(x_2)$ is negligible vs. $o(x)$. This linearizes the problem, and allows us to state that the total measured field from all of $o(x)$ is

$$\underbrace{e^{-i\frac{2\pi}{\lambda}\vec{e_I}\cdot x}}_{\text{INCIDENT}} + \frac{1}{4\pi|x|}\underbrace{\int e^{-i\frac{2\pi}{\lambda}(\vec{e_I}-\vec{e_S})\cdot x} o(x)dx}_{\text{SCATTERED}} \quad (4)$$

The goal is to reconstruct $o(x)$ from this field. In the sequel we subtract off the incident plane wave and omit the geometric spreading factor $1/(4\pi|x|)$.

As the incident $\vec{e_I}$ and scattered $\vec{e_S}$ directions sweep over the unit sphere, the 3-D Fourier transform $O(k)$ of $o(x)$ is determined over a set of spheres of radius $\frac{2\pi}{\lambda}$ centered on another sphere of radius $\frac{2\pi}{\lambda}$. This is also the set of all spheres of radius $\frac{2\pi}{\lambda}$ that pass through the origin.

Some thought (picture a flyball governor spinning around all possible axes) shows that this sweeps over all wavenumbers with magnitudes $\leq \frac{4\pi}{\lambda}$. In fact, we can omit half of the incident or scattered directions and still recover $\{O(k), |k| \leq \frac{4\pi}{\lambda}\}$ (picture an anemometer spinning around all possible axes and use reciprocity). Hence we can recover only a low-wavenumber-filtered version of $o(x)$; the shorter the wavelength $\lambda$, the higher the $o(x)$ cutoff wavenumber. This is important in the development to follow.

For more details see any paper on diffraction scattering or tomography. We have found the papers of A.J. Devaney to be particularly helpful to us.

### *B. X-Ray Crystallography*

We now specialize to the case of $o(x)$ is a crystal:

- $o(x)$ is *periodic*: $o(x) = o(x + [L_x, L_y, L_z])$ for some lengths $L_x, L_y, L_z$. Each period of $o(x)$ is a unit cell;

• $o(x)$ is *atomic*: $o(x)=\sum_{n=1}^{M}o_n\delta(x-x_n)$ for some values and locations $\{(o_n,x_n),n=1\ldots M\}$. $o_n$ is proportional to the atomic number of the $n^{th}$ atom.

More precisely $o(x)$ is sparse (mostly zero-valued) and its nonzero values specify the electron density, which is clustered around atomic nuclei. It thus may be collections of small regions rather than impulses.

These properties of $o(x)$ imply the following properties of its Fourier transform $O(k)$:

• $O(k)$ is *discrete* in wavenumber $k$ (see below);
• The atomicity of $o(x)$ lead to more complicated properties of $O(k)$ (see below).

More precisely $O(k)$ can be written as

$$\sum_{i,j,k}O_{i,j,k}\delta(k_x-i\frac{2\pi}{L_x})\delta(k_y-j\frac{2\pi}{L_y})\delta(k_z-k\frac{2\pi}{L_z}). \quad (5)$$

More properly, the periodic $o(x)$ can be expanded in a 3-D Fourier series with Fourier coefficients $O_{i,j,k}$.

However, there are also consequences to the use of X-ray wavelengths, which have $\lambda\approx$1 Angstrom:

• Only the $o(x)$ Fourier magnitude $|O(k)|$ can be measured, since there is no lens for imaging X-rays;
• Only a low-wavenumber-filtered version of $o(x)$ can be recovered, with resolution about 1 Angstrom.

We thus have the two problems of phase retrieval (recovering $\angle O(k)$) and superresolution or bandwidth extrapolation (recovering $O(k)$ for large $|k|$). There are well-known iterative algorithms for both of these problems that require a *support constraint*: $o(x)=0$ for $|x|>R$ for some $R$. However, these algorithms cannot be applied here, since the periodicity of $o(x)$ implies there is no support constraint. It could be assumed that in each unit cell there is a bounding region in which the crystal is known to be empty of atoms, but this is seldom true in practice.

The problem is as follows: How to take advantage of sparsity and periodicity to recover the phase and high-wavenumber information. Note sparsity cannot be used as a support constraint: Although $o(x)$ is mostly zero, there are no regions in which $o(x)$ is *known* to be zero, so there is no *fixed* constraint.

## *C. Glossary*

We repeat the following glossary from an earlier paper of ours linking terms in X-ray crystallography to corresponding concepts in signal processing. This glossary should be helpful to readers in both fields.

| X-RAY CRYSTAL. | SIGNAL PROC. |
|---|---|
| Electron density | image or object |
| Crystal structure | space-periodic |
| Unit cell lengths | spatial periods |
| Atomicity | object sparsity |
| P$\bar{1}$ group | even symmetry |
| Reciprocal space | Fourier domain |
| Structure factors | $\mathcal{F}\{$object$\}$ |
| Structure amplitude | $|\mathcal{F}\{$object$\}|$ |
| Patterson map | autocorrelation |
| Karle-Hauptman determinants$\geq$0 | Circulant matrix pos.semi-definite |
| Sayre equation | $\mathcal{F}\{o(x)(o(x)-1)\}$ |
| Isomorphous replacement | Inserting atoms into the object |
| Anomalous dispersion | Vary $\lambda$ excite heavy atoms |
| Crystallographic symmetry | Rotation invariant within the lattice |
| Noncrystallographic symmetry | Rotation invariant extending lattice |
| Born approximation | Linearization |

## *D. Superresolution*

The problem here is to reconstruct the $K^2$–K-sparse autocorrelation $r_n$ from knowledge of the low-wavenumber values of the DFT magnitude $\{|X_k|,|k|\leq K^2\}$. Repeating (**??**), we have

$$r_n=\sum_{i=0}^{M-1}x_ix_{i-n};\quad |X_k|^2=\sum_{n=0}^{N-1}r_ne^{-j2\pi nk/N} \quad (6)$$

Since $x_n$ is K-sparse, $r_n$ is ($K^2$–K)-sparse.

Let $s_n$ be the indicator function for nonzero $r_n$:

$$\begin{cases}s_n=0 & \text{if } r_n\neq 0\\ s_n\neq 0 & \text{if } r_n=0\end{cases}\quad \begin{cases}S_k\neq 0 & 0\leq k\leq K^2\\ S_k=0 & \text{otherwise}\end{cases} \quad (7)$$

where $S_k$ is the N-point DFT of $s_n$. Then we have

$$s_nr_n=0\rightarrow\sum_{i=0}^{N-1}|X_i|^2S_{k-i}=0 \quad (8)$$

Since there are only $K^2$+1 nonzero values of $S_k$, the $2K^2$+1 known values $\{|X_k|^2,|k|\leq K^2\}$ determine $S_k$ to an irrelevant scale factor. The second equation can be written as a Hermitian Toeplitz linear system of equations. All this generalizes directly to multiple dimensions; the only difference is that the Toeplitz matrix becomes a Toeplitz-block-Toeplitz matrix.

The polynomial of degree $2K^2$+1 with coefficients $S_k$ has zeros $\{e^{jn_i}\}$, where $n_i$ are now the locations of the nonzero values of $r_n$. So an inverse DFT of $S_k$ is zero at locations of nonzero $r_n$. Then the linear system (**??**) determines the nonzero values of $r_n$.

## III. Recursive Reconstruction of Sparse Signal $x_n$ from Autocorrelation $r_n$

Having reconstructed the ($K^2$–K)-sparse autocorrelation $r_n$, the goal is now to reconstruct the K–sparse signal $x_n$. We assume that each $r_n = x_i x_{i-n}$ for some specific $i$; no $r_n$ except n=0 is the sum of more than one such term. This is realistic since the $x_n$ are resolved into points at random locations.

### *A. Finite Support $x_n$*

Suppose $x_n$=0 outside the range $0 \leq n \leq M-1$ and N≥2M–1, so there is no aliasing. Without loss of generality, due to the *translational ambiguity*, set $x_0$=1. Since each $r_n$ is a single term $x_i x_{i-n}$, we can set all nonzero $x_n$ and $r_n$ to one to determine locations of nonzero $x_n$. Then arranging the actual values of $r_n$ into a matrix, the $x_n$ are determined by a rank-one factorization of this matrix. This determines $x_n$ to an overall *sign ambiguity.*

The algorithm is *initialized* as follows:
1. Let $r_n$=0 outside the range $|n| \leq M$ and $r_{\pm M}$=1. Since $x_0$=1, we have $x_M$=1.
2. Let L be the largest integer L<M such that $r_{\pm M}$=1. Then either $x_L$=1 or $x_{M-L}$=1.
3. There is no way to tell at this point–this is the *reversal ambiguity.*
4. Without loss of generality, let $x_L = 1$.
5. Take $r_L$ and $r_{M-L}$ "off the board" by setting to 0.

The algorithm *recursions* are as follows:
1. For each n decreasing from L to 2;
2. If $r_n$=0 go to the next smaller n;
3. If $r_n \neq 0$ either $x_n$=1 or $x_{M-n}$=1;
4. Check $r_{L-n}$ and $r_{|L-(M-n)|}$;
5. One of these will be one; the corresponding $x$=1;
6. If both are one, both $x$=1;
7. Say $x_n$=1. Set $r_{|i-n|}$=0 for all $\{i : x_i = 1\}$;
8. This prevents false alarms from these $r_n$;
9. Go to the next smaller value of n; go to #2.

### *B. Tiny Example*

As an example, suppose we are given $r_n$=: {1,0,0,1,1,1,1,1,0,4,0,1,1,1,1,1,0,0,1}

*Initialization:*
- $x_0 = x_9 = 1$.
- $r_6$=1→ $x_6$=1 or $x_3$=1:
- Without loss of generality $x_6$=1;
- Take $r_6$ and $r_{9-6}$ "off the board."

*Recursion:*
- $r_5$=1→ $x_5$=1 or $x_4$=1:
- $x_5$=1→ $r_{6-5}$=1. NO.
- $x_4$=1→ $r_{6-4}$=1. YES.
- Take $r_{9-4}, r_{6-4}, r_{|0-4|}\}$ "off the board."
- $r_n$=0 for all remaining $n$.

So the solution is

$$x_n = \{1, 0, 0, 0, 1, 0, 1, 0, 0, 1\} \tag{9}$$

or its reversal, translation, or sign change.

### *C. No Support $x_n$*

With no support information, $r_n$ computed from the squared DFT magnitude is the *cyclic* (aliased) autocorrelation $r_n + r_{N-n}$. Assuming at most one of these two terms is nonzero, the problem is to distinguish whether each nonzero computed $r_n$ is actually $r_n$ or an aliased value $r_{N-n}$.

This can be performed during the *recursion*, since:
- $r_n$=1→ $x_n$=1 or $x_{M-n}$=1;
- $r_{N-n}$=1→ $x_{N-n}$=1 or $x_{M-(N-n)}$=1;
- M<N is the largest index $r_M$=1;
- So we need to check the correlations between all previously computed nonzero $x_i$ and these four candidate nonzero $x_n$, rather than the two before;
- Only one of these four sets will **all** be nonzero;
- This determines a nonzero $x_n$ and dealiases $r_n$.

The *initialization* is more difficult:
- There is now a *cyclic translational* ambiguity.
- Two nonzero $x_n$ are closest together.
- Let them be $x_0$=1 and $x_M$=1, where:
- M is largest and N-M smallest $i : r_i \neq 0$.

Now let $L$ be the next-largest index such that $r_L \neq 0$. Run the algorithm assuming $r_L$ is a true autocorrelation. If the result doesn't work, $r_{N-L}$ was the true autocorrelation. Run the algorithm using the next-largest index. Continue until the algorithm works (generates an $x_n$ satisfying the $r_n$).

### *D. Converting 2-D to 1-D Problems*

A 2-D or 3-D phase retrieval problem with known finite image support can be unwrapped into a 1-D phase retrieval problem by rows or columns. The resulting 1-D problem signal support constraint includes known bands of zeros, ensuring a unique solution to the trivial ambiguities.

A 2-D or 3-D phase retrieval problem without any support information can be unwrapped into a 1-D phase retrieval problem using the Agarwal-Cooley fast convolution algorithm. This regards the 2-D or 3-D problem indices as a residue number system representation of the 1-D problem indices. The DFT sizes $N_x, N_y, N_z$ must be relatively prime integers; that is, the lengths $L_x, L_y, L_z$ must be relatively prime integer multiples of the same unit length.

## IV. Numerical Examples

### A. Example: Known Finite Support

The image is a $100 \times 100$ 16-sparse image in which each nonzero pixel has value one. Its autocorrelation is $199 \times 199$ and is 240-sparse (excluding the $0^th$ lag which has value 16). So

- K=16; $K^2$–K=240; N=2M-1=199.

The 199×199 2-D DFT magnitude is known only for the 20 lowest wavenumbers along each axis. The goal is to reconstruct the 16-sparse image from this low-wavenumber Fourier magnitude data.

The autocorrelation estimate obtained by taking the inverse 199×199 2-D DFT of this given data, setting the unknown Fourier magnitudes to zero, is shown in Figure 1. The $0th$ lag is at the center. It can be seen that reconstructing $x_n$ from this unresolved data would be difficult.

The true autocorrelation (with the $0th$ lag set to zero for display purposes) is shown in Figure 2.

The logarithms of the singular values of the Toeplitz-block-Toeplitz matrix are plotted in Figure 3. The sharp drop between 241 and 242 shows that this matrix has an effective rank of 241. The threshold used to determine the locations of nonzero $r_{i,j}$ is between the 241 and 242 smallest values. Specifically,

- $\sigma_{241} = 3.4 \times 10^{-5}$;    $\sigma_{242} = 5 \times 10^{-10}$
- $t_{241} = 1.5 \times 10^{-10}$;    $t_{242} = 4 \times 10^{-5}$

Reconstructed autocorrelation is shown in Figure 4. Compare to the actual autocorrelation in Figure 2.

The original and reconstructed sparse images are shown in Figures 5 and 6. Note the translation; this is considered a trivial ambiguity. In fact, the algorithm reconstructed the reversal of the original image; its reversal is shown to facilitate comparison.

Matlab code used to generate this example:

```
clear;rand('seed',0);XX=ceil(rand(100,100)-.9988);
YY=conv2(XX,fliplr(flipud(XX)));
FYY=fftshift(fft2(YY));
FYY=FYY(100-20:100+20,100-20:100+20);FY1=FYY(:);
%Autocorrelation reconstructed from low freqs only
FYY(199,199)=0;YL=abs(ifft2(FYY)); %modulated
YL(1:3,1:3)=zeros(3,3);YL(197:199,197:199)=zeros(3,3);
YL(197:199,1:3)=zeros(3,3);YL(1:3,197:199)=zeros(3,3);
figure,imagesc(YL),colormap(gray)
FY1=FY1((41^2+1)/2:41^2);TT=toeplitz(FY1,FY1');
II=[];for I=0:20;II=[II [1:21]+I*41];end
T=TT(II,II);[U S V]=svd(T); %Toeplitz-block-Toeplitz
P=reshape(V(:,441),21,21);FP=abs(fft2(P,199,199));
ZZ=YY;ZZ(100,100)=1; %Set 0th lag to 1 for display
figure,imagesc(ZZ),colormap(gray)
figure,plot(log(diag(S))),title('SINGULAR VALUES')
FPP=FP(:);[W1,J]=sort(FPP);W(J(1:241))=1;W(199^2)=0;
figure,imagesc(reshape(W,199,199)),colormap(gray)
XX(199,199)=0;%W1(241)=1.5X10^-10;W1(242)=4X10^-5
figure,imagesc(XX),colormap(gray)
X1=(XX(:))';X=X1(min(find(X1>0)):max(find(X1>0)));
Y1=(YY(:))';Y=Y1(min(find(Y1>0)):max(find(Y1>0)));
M=length(Y);Y=Y((M+1)/2:M);
[W2,N]=find(Y>0);L=length(N);
Z(N(1))=1;Z(N(L))=1;Z(N(L-1))=1;
Y(N(L-1))=0;Y(N(L)+1-N(L-1))=0;
for I=L-2:-1:2;if Y(N(I))==1;
if Y(N(L-1)-N(I)+1)==1;Z(N(I))=1;
Y(N(I+1)+find(Z(N(I+1):N(L))==1)-N(I))=0;
Y(N(I)-find(Z(N(1):N(I-1))==1)+1)=0;
else NI=N(L)+1-N(I);
Z(NI)=1;NI1=N(L)+1-N(I-1);
Y(NI1+find(Z(NI1:N(L))==1)-NI)=0;
Y(NI-find(Z(N(1):N(L)+1-N(I+1))>0)+1)=0;
end;else;end;end;Z=fliplr(Z);Z(199^2)=0;
ZZ=reshape(Z',199,199); %Get reversal
figure,imagesc(ZZ),colormap(gray)
```

### B. Example: Sparse Image; No Support

This example is similar to the first example, with similar results. The 16-sparse image is now 100×99, where 100 and 99 are relatively prime integers. This facilitate unwrapping from 2-D to 1-D using the Agarwal-Cooley fast convolution. The image is actually generated as a 1-D signal whose ends are chosen to avoid having to rerun the algorithm, and then mapped to 2-D, then back to 1-D for the algorithm.

Figures correspond to those from the first example, and are quite similar. Singular values and thresholds are also similar numbers, and are not given.

Matlab code used to generate this example:

```
clear;rand('seed',1);X=ceil(rand(1,9900)-.999);
X(9900)=1;X(9899)=1;X(9897)=1; %To avoid having to
N=length(X);Y1=conv(X,fliplr(X)); %rerunning algorithm
Y2=Y1(N+1:2*N-1)+Y1(1:N-1);Y2=[Y1(N) Y2]; %Cyclic
for I=0:100*99-1;YY(mod(I,100)+1,mod(I,99)+1)=Y2(I+1);end
%Autocorrelation reconstructed from low freqs only
FYY=fftshift(fft2(YY));FYY=FYY(51-20:51+20,50-20:50+20);
FY1=FYY(:);FYY(100,99)=0;YL=abs(ifft2(FYY)); %mod.
YL(1:3,1:3)=zeros(3,3);YL(98:100,97:99)=zeros(3,3);
YL(98:100,1:3)=zeros(3,3);YL(1:3,97:99)=zeros(3,3);
figure,imagesc(YL),colormap(gray)
FY1=FY1((41^2+1)/2:41^2);TT=toeplitz(FY1,FY1');
II=[];for I=0:20;II=[II [1:21]+I*41];end
T=TT(II,II);[U S V]=svd(T); %Toeplitz-block-Toeplitz
figure,plot(log(diag(S)))
P=reshape(V(:,441),21,21);FP=abs(fft2(P,100,99));
FPP=FP(:);[W1,J]=sort(FPP);W(J(1:241))=1;W(9900)=0;
ZY=reshape(W,100,99);YY(1,1)=1; %0th lag→1
figure,imagesc(YY),colormap(gray)
figure,imagesc(ZY),colormap(gray)
%Agarwal-Cooley unwrapping:
for I=0:100*99-1;Y(I+1)=ZY(mod(I,100)+1,mod(I,99)+1);end
Y=[Y(2:N-1) 1 1]; %cyclic shift 0th lag to N
[W,K]=find(Y>0.1);L=length(K);M=K(L-1);
Z(K(L))=1;Z(K(L-1))=1;Z(K(L-3))=1; %initialize
Y(K(L))=0;Y(K(L-1))=0;Y(K(L-3))=0; %from initialized
Y(K(L-2))=0;
for I=L-4:-1:(L+1)/2;if Y(K(I))>0.1;
if Y(K(L)-K(I))*Y(K(L-1)-K(I))*...
Y(K(L-3)-K(I))>0.1;Z(K(I))=1;
for J=1:L;if Z(K(J))>0.1;if K(I)-K(J)~=0;
Y(abs(K(I)-K(J)))=0;end;end;end;Y(N-K(I))=0;
elseif Y(K(L)-(M-K(I)))*Y(K(L-1)-(M-K(I)))...
*Y(K(L-3)-(M-K(I)))>0.1;Z(M-K(I))=1;
for J=1:L;if Z(K(J))>0.1;if M-K(I)-K(J)~0;
Y(abs(M-K(I)-K(J)))=0;end;end;end;Y(N-(M-K(I)))=0;
end;else;end;end
for I=0:100*99-1;XX(mod(I,100)+1,mod(I,99)+1)=X(I+1);end
for I=0:100*99-1;ZZ(mod(I,100)+1,mod(I,99)+1)=Z(I+1);end
figure,imagesc(XX),colormap(gray)
figure,imagesc(ZZ),colormap(gray)
```

### *C. Example: Sparsifiable Image; No Support*

In this example we reconstruct a non-sparse but sparsifiable image from its cyclic autocorrelation. The problem size is much smaller in order to illustrate several important points that are lost in larger problems. We also skip the superresolution part.

The image is a block letter 'E' in which each segment has a different size. This can be sparsified by a 2-D difference operator (corner detector). Each lag of the $30\times29$ cyclic autocorrelation of the sparsified image is a single product. For larger problems, this will almost certainly hold, unless the block letters have precisely identical segments.

The cyclic translational ambiguity of the unwrapped 1-D cyclic phase retrieval problem is addressed by simply translating the solution. Without this, the reconstructed block letter 'E' is circularly shifted in 2-D, which may make recognition of the letter difficult. Note this is an inevitable feature of the problem that requires additional information.

In Example #2, the sparse image had a pixel added to it so that the algorithm would work the first time. In the present example, the algorithm must be rerun several times since the first few nonzero cyclic autocorrelation lags are in fact aliased small lags. The L–3, L–4 and L–5 lags were all aliased; the L–6 lag was genuine, and the algorithm then worked.

To illustrate this, we have included a plot of both the cyclic autocorrelation and the doubled (to distinguish it) linear autocorrelation. Examination of this plot (which is of course unknown from the data) shows that the largest few lags are in fact aliased small lags. These are the lags noted above.

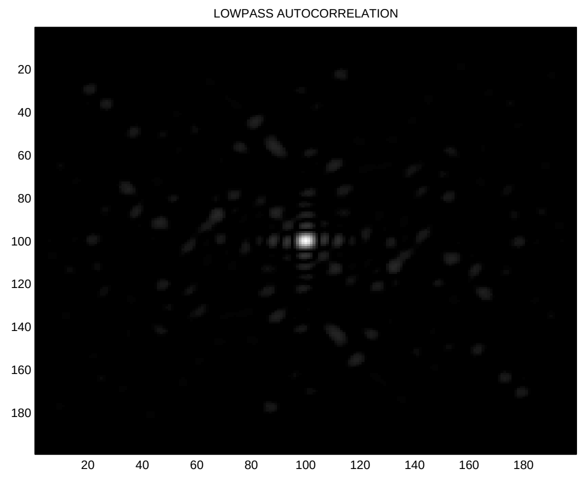
The Matlab code provided below only checks for nonzero lags between a prospective nonzero image location and the three known nonzero image locations used to initialize the algorithm. This is simpler than checking lags between the prospective location and all previously determined locations, but it does allow a single false value to creep in.

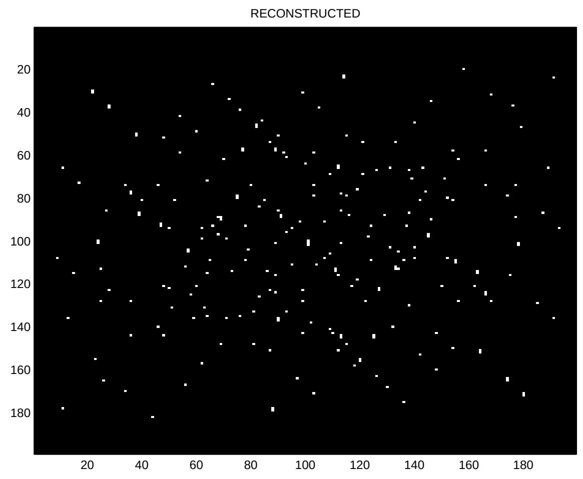
We have chosen to keep the program simple, to make the point that even for a problem as dense as this one, only a single false value crept in. Complete checking may be unnecessary for most problems.

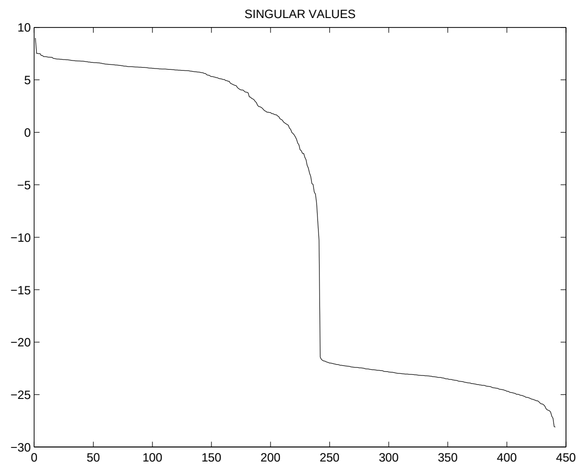
Since the sparsified image is used to create a non-sparse image, its true values must be computed by a rank-1 matrix factorization. This step is now included in the program below.

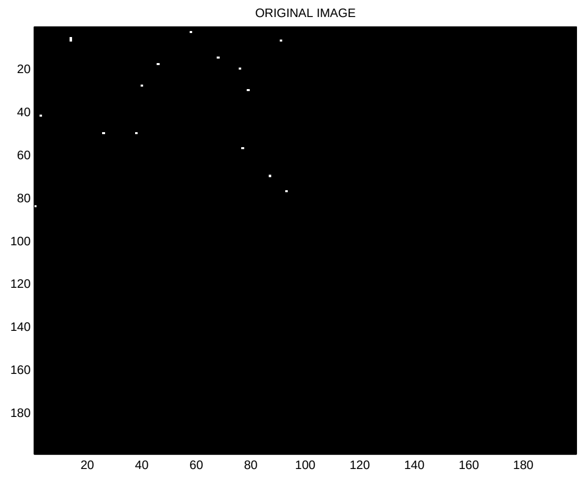
Matlab code used to generate this example:

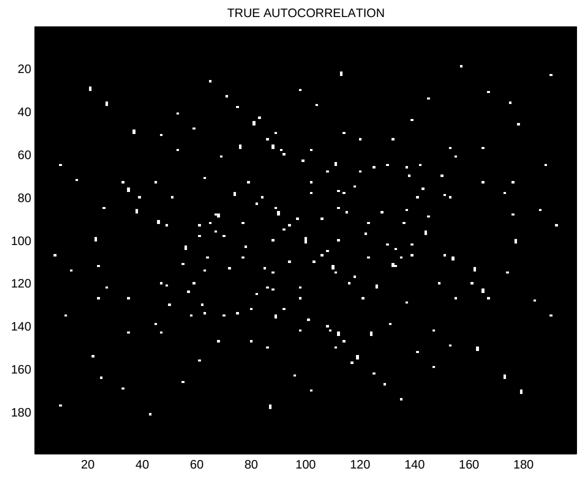
```
%Create block letter E with unequal parts:
clear;X(27,27)=0;X(14:16,5)=ones(3,1);
X(3:22,3:4)=ones(20,2);X(3:6,5:21)=ones(4,17);
X(9:13,5:20)=ones(5,16);X(17:22,5:25)=ones(6,21);
FY=abs(fft2(X,30,29)).^2; %Given Fourier data.
figure,imagesc(log(fftshift(FY))),colormap(gray)
title('FOURIER MAGNITUDE DATA; ORIGIN AT')
FDY=FY.*abs(fft2([1 -1;-1 1],30,29)).^2;
DY=(real(ifft2(FDY)));for I=0:30*29-1;
Y1(I+1)=DY(mod(I,30)+1,mod(I,29)+1);end
figure,imagesc(DY),colormap(gray)
title('SPARSIFIED CYCLIC AUTOCORRELATION')
figure,plot(Y1), %GOAL: Reconstruct X from DY.
title('UNWRAPPED CYCLIC AUTOCORRELATION')
%abs(Y1): To find nonzero locations of DX
Y=abs(Y1);Y=[Y(2:870) 1];N=length(Y);
[W,K]=find(Y>0.1);L=length(K);M=K(L-1);
%initialize: L-3,L-4,L-5 don't work.
Z(K(L))=1;Z(K(L-1))=1;Z(K(L-6))=1;
for I=L-7:-1:7(L+1)/2;if Y(K(I))>0.1;
if Y(K(L)-K(I))*Y(K(L-1)-K(I))*...
Y(K(L-6)-K(I))>0.1;Z(K(I))=1;
for J=1:L;if Z(K(J))>0.1;if K(I)-K(J)~=0;
Y(abs(K(I)-K(J)))=0;end;end;end;Y(N-K(I))=0;
elseif Y(K(L)-(M-K(I)))*Y(K(L-1)-(M-K(I)))...
*Y(K(L-6)-(M-K(I)))>0.1;Z(M-K(I))=1;
for J=1:L;if Z(K(J))>0.1;if M-K(I)-K(J)~=0;
Y(abs(M-K(I)-K(J)))=0;end;end;end;
Y(N-(M-K(I)))=0;end;else;end;end
%Translational ambiguity: cyclic shift Z.
%Artifact: set Z(853)=0. See paper text.
Z=Z([549:870 1:548]);Z(853)=0;Y=[Y1(2:870) 1];
%Now find actual values of DY from locations.
[W,K]=find(Z>0.1);L=length(K);%Reuse variables.
for I=1:L;for J=1:L;if K(I)-K(J)~=0;YY(I,I)=1;
YY(I,J)=Y(abs(K(I)-K(J)));end;end;end;
%Compute rank=1 factorization of matrix YY:
[U S V]=svd(YY);Z(K)=-Z(K).*V(:,1)'/V(1,1);
for I=0:30*29-1;DZ(mod(I,30)+1,mod(I,29)+1)=Z(I+1);end
figure,imagesc(DZ),title('RECONSTRUCTED LOCATIONS')
ZZ=fliplr(flipud(cumsum(fliplr(flipud(cumsum(DZ)))')'));
figure,imagesc(ZZ),title('RECONSTRUCTED IMAGE')
```

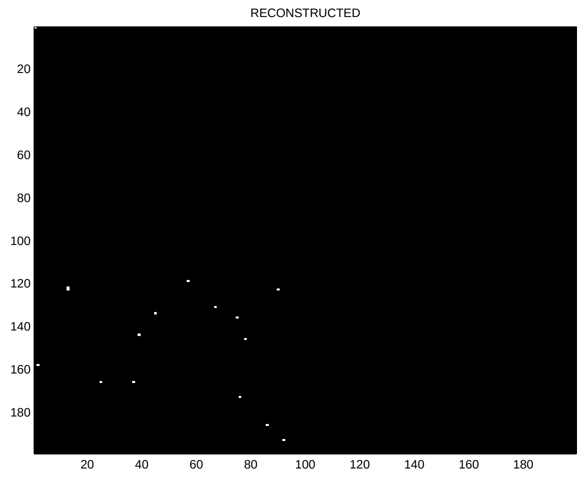

LOWPASS AUTOCORRELATION

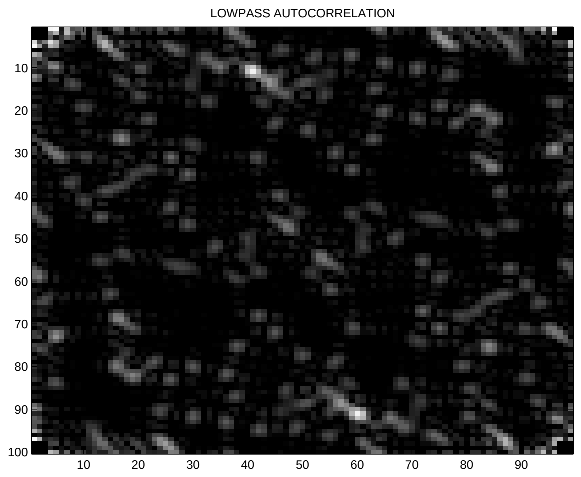

RECONSTRUCTED

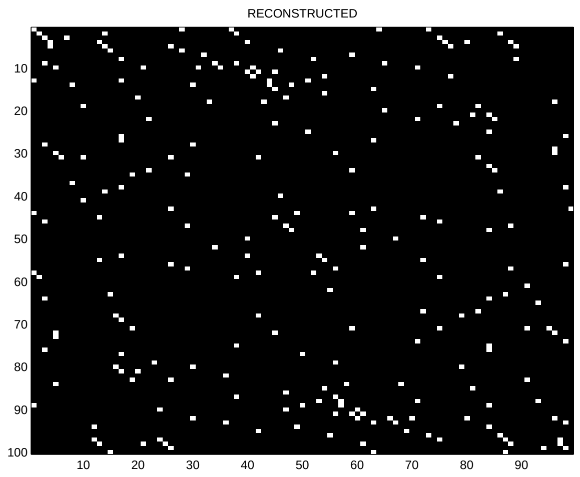

SINGULAR VALUES

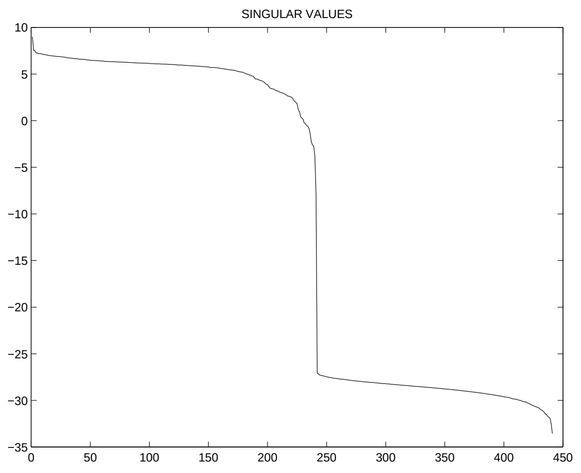

ORIGINAL IMAGE

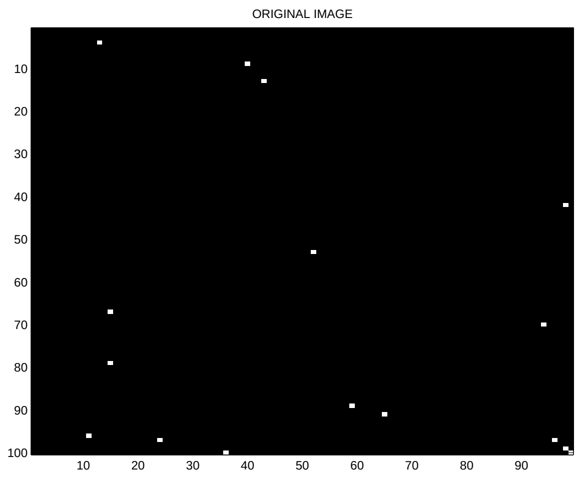

TRUE AUTOCORRELATION

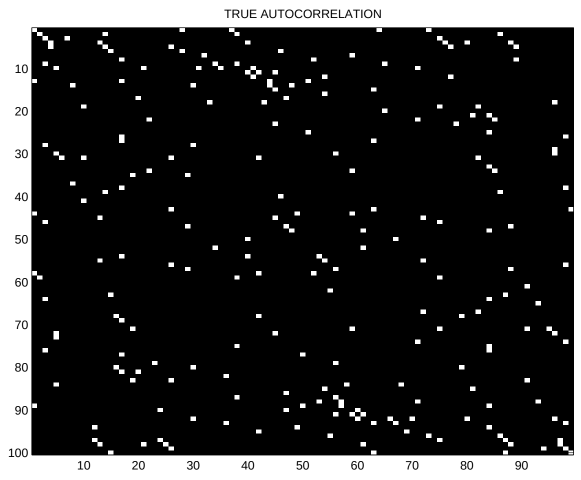

RECONSTRUCTED

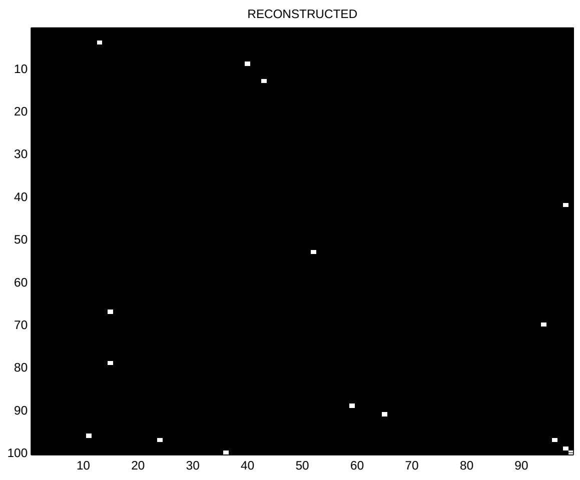

LOWPASS AUTOCORRELATION
10
20
30
40
50
60
70
80
90
100

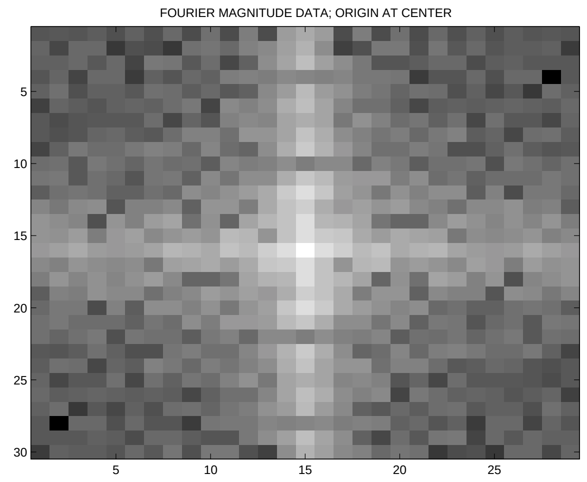

RECONSTRUCTED
10
20
30
40
50
60
70
80
90
100

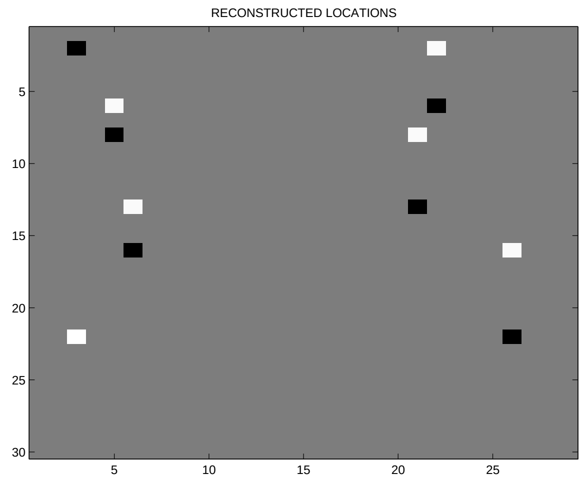

SINGULAR VALUES
10
5
0
−5
−10
−15
−20
−25
−30
−35
0
50
100
150
200
250
300
350
400
450

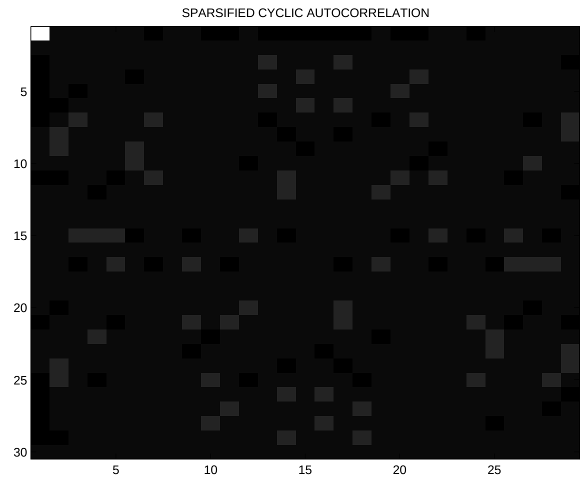

ORIGINAL IMAGE
10
20
30
40
50
60
70
80
90
100

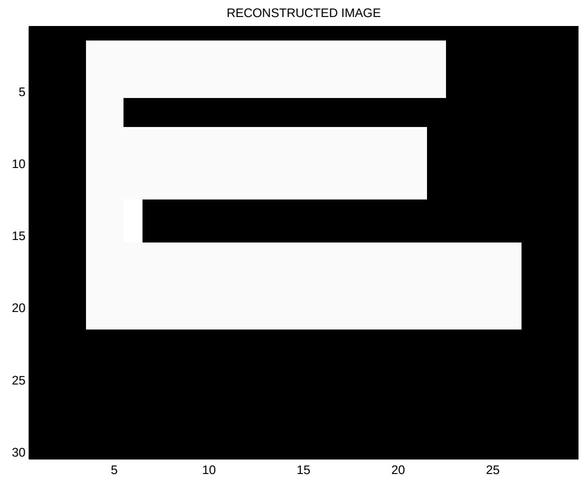

TRUE AUTOCORRELATION
10
20
30
40
50
60
70
80
90
100

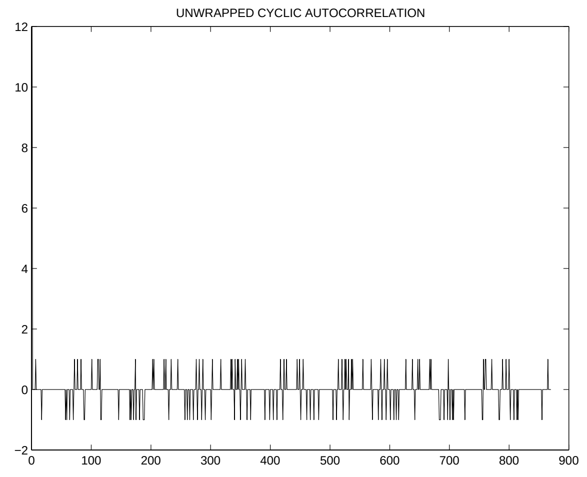

RECONSTRUCTED
10
20
30
40
50
60
70
80
90
100

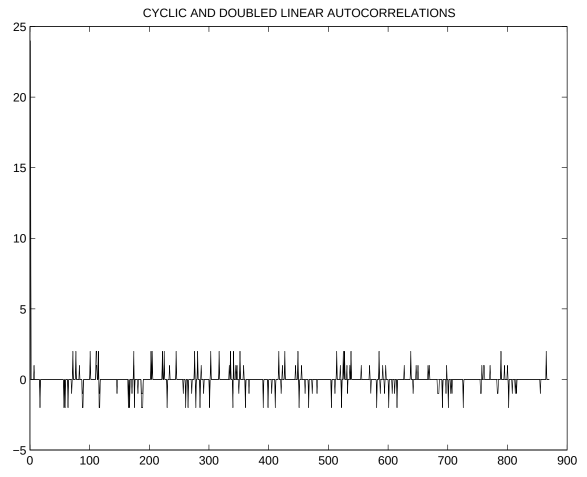

FOURIER MAGNITUDE DATA; ORIGIN AT CENTER
5
10
15
20
25
30
5
10
15
20
25


RECONSTRUCTED LOCATIONS
5
10
15
20
25
30
5
10
15
20
25


SPARSIFIED CYCLIC AUTOCORRELATION
5
10
15
20
25
30
5
10
15
20
25


RECONSTRUCTED IMAGE
5
10
15
20
25
30
5
10
15
20
25


UNWRAPPED CYCLIC AUTOCORRELATION
12
10
8
6
4
2
0
−2
0
100
200
300
400
500
600
700
800
900


CYCLIC AND DOUBLED LINEAR AUTOCORRELATIONS
25
20
15
10
5
0
−5
0
100
200
300
400
500
600
700
800
900