Non-Iterative Sparse Image Reconstruction from a Few 2-D DFT Frequency Values

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

Abstract— **We consider the problem of reconstructing a sparse image from a few of its 2-D DFT frequency values. A sparse image has pixel values that are mostly zero, with a few non-zero values at unknown locations. The number of known 2-D DFT values must exceed four times the number of non-zero pixel values. We unwrap the 2-D problem to a 1-D problem using the Good-Thomas FFT, and apply Prony's method to compute the non-zero pixel value locations. Thus we reformulate the problem as a dual 2-D harmonic retrieval problem. Our solution has three advantages over direct application of 2-D ESPRIT: (1) Instead of solving a huge generalized eigenvalue problem, we compute the roots on the unit circle of a huge polynomial; (2) the locations of the known 2-D DFT values need not form a centrosymmetric region; and (3) there are no matching issues. Our algorithm is also applicable to 2-D beamforming.**

I. INTRODUCTION

A. Problem Discussion

The problem of reconstructing an image from its irregular frequency samples (arbitrary values of its $(N \times N)$ 2-D DFT where N is large) arises in many applications. Some examples and the corresponding locations where the DFT is known:

- SAR: On several arcs of points;
- CAT: On a polar raster of points;
- MRI: On a polar raster of points;
- fMRI: On a square spiral of points;
- Limited-angle tomography: In a bowtie region;
- 2-D FIR filter design: A prescribed response.

If the $(N \times N)$ 2-D DFT of an image is known *everywhere*, then clearly we can recover the image using an inverse 2-D DFT. But if the DFT is known only for *some* of these DFT values, this will not work. While interpolation can be used to resample the frequency values to a rectangular lattice, this necessarily involves some approximation and some computation. And if only a few values of the 2-D DFT are known, then the problem is grossly underdetermined.

Many images arising in medical imaging applications consist of several regions in which the pixel values are constant. A Laplacian or similar edge-enhancing operation transforms the image reconstruction problem into one of reconstructing relatively few non-zero pixel values. This should require a much smaller number of 2-D DFT values (data), and result in a much smaller-sized problem than reconstructing the entire image from many more 2-D DFT values.

Snakes have been used to parametrize image region boundaries, but these are computationally intensive, and they can encounter problems for complicated boundary curves. Linear programming, which computes the minimum ℓ_1 norm, has been used to reconstruct images whose pixel values are restricted to a few values, but there is no guarantee these will work, since the heuristic that ℓ_1 minimization leads to solutions at simplex corners is not guaranteed.

B. Contributions of This Paper

This paper reformulates the sparse image reconstruction problem as a (Fourier) dual 2-D harmonic retrieval problem. Analogous quantities are:

sparse image	**harmonic retrieval**	**beamforming**
nonzero pixels	unknown wavenums	source angles
2-d dft support	2-d data window	array pattern

This suggests use of one of the algorithms that have been developed for 2-D harmonic retrieval. 2-D unitary ESPRIT in particular seems well-suited to this problem. But there are three problems associated with simple application of 2-D harmonic retrieval to the sparse image reconstruction problem:

- 2-D ESPRIT requires solution of a generalized eigenvalue problem. While this is suitable for beamforming for a small number of sources, even a very sparse image contains far too many non-zero pixel values for this to be computationally practical;
- 2-D ESPRIT requires the array pattern to be centrosymmetric. In limited angle tomography, known 2-D DFT values are not centrosymmetric;
- 2-D harmonic retrieval includes the matching problem of correctly associating wavenumber locations along one axis with those along the other axis;
- 2-D ESPRIT avoids this problem by reformulating

the problem using the mapping from centrohermitian matrices to real matrices, but this is often inapplicable (see the above problem).

Instead of using 2-D harmonic retrieval methods, we use the Good-Thomas FFT to unwrap the 2-D problem into a 1-D problem. This allows application of 1-D harmonic retrieval methods, such as least-squares Prony or Pisarenko. This results in the following advantages, which answer the problems above:

- These methods require only the solution of a structured linear system of equations, followed by computing the unit-circle zeros of a polynomial;
- The support of known 2-D DFT values need only be a convex polygonal region. In particular, limited-angle tomography is included;
- Since the 2-D to 1-D mapping is unique (provided the 2-D DFT transform lengths are relatively prime), there is no matching problem.

C. Problem Statement

The goal is to reconstruct a sparse image

$$x(i,j) = \begin{cases} x(i_k, j_k) & \text{for } (i_k, j_k) \in \Theta; \\ 0 & \text{for } (i,j) \notin \Theta. \end{cases} \quad (1)$$

from some of its $((N-1) \times N)$ 2-D DFT values

$$X(k_1,k_2) = \begin{cases} X(k_1,k_2) & \text{for } (k_1,k_2) \in \Omega; \\ \text{unknown} & \text{for } (k_1,k_2) \notin \Omega. \end{cases} \quad (2)$$

where the $((N-1) \times N)$ 2-D DFT is

$$X(k_1,k_2) = \sum_{i_1=0}^{N-2} \sum_{i_2=0}^{N-1} x(i_1,i_2) e^{-j2\pi(\frac{i_1 k_1}{N-1} + \frac{i_2 k_2}{N})} \quad (3)$$

and where

- Ω=locations of known frequency values;
- Ω=known convex polygonal region;
- Θ=locations of non-zero image pixel values;
- Θ=unknown and entirely arbitrary;
- Size (cardinality) of $\Omega > 4$(size Θ).

In matrices, we will use X_{k_1,k_2} instead of $X(k_1,k_2)$ to save space.

II. DERIVATION OF DETERMINISTIC PRONY'S METHOD AND ESPRIT

A. Derivation of ESPRIT

We now derive a *deterministic* 1-D ESPRIT algorithm for the sparse signal problem. We have:

$$X(k) = \sum_{n \in \Theta} x(n) e^{-j2\pi nk/N} \quad (4)$$

where Θ is again the set of $0 \le n \le (N-1)$ where $x(n) \neq 0$. We then have

$$C_0 = F^H D F \quad \text{and} \quad C_1 = F D E F^H \quad (5)$$

where

- $C_0(i,j) = X((i-j) \bmod N)$=circulant
- $C_1(i,j) = X((i-j+1) \bmod N)$=circulant
- $F(i,k) = e^{-j2\pi ik/N}$=DFT matrix
- D=diag$[x(0) \ldots x(N-1)]$=diagonal
- E=diag$[e^{-j2\pi 0/N} \ldots e^{-j2\pi(N-1)/N}]$=diagonal

Note this is just the modulation theorem for the DFT, or the Fourier dual of the time-delay theorem.

Next, delete all rows and columns of C_0 and C_1 that contain an unknown value of $X(k)$. This leaves

$$T_0 = V^H D V \quad \text{and} \quad T_1 = V^H D E V \quad (6)$$

- V=Vandermonde matrix, i.e., $(V_{ij} = v_i^j)$
- $V = F$ with some rows and columns deleted
- T_0 has Toeplitz blocks
- T_1 has Toeplitz blocks
- D=diag$[x(n), n \in \Theta]$=diagonal
- E=diag$[e^{-j2\pi n/N}, n \in \Theta]$=diagonal

Provided that the number of non-zero $x(n)$ does not exceed the size of T_0 and T_1, we have

$$\lambda T_0 - T_1 = V^H D(\lambda I - E) V \quad (7)$$

is singular if and only if $\lambda = \{e^{-j2\pi n/N}, n \in \Theta\}$. Thus the locations n of the non-zero $x(n)$ can be found from the eigenvalues of the generalized eigenvalue problem for T_0 and T_1.

This procedure generalizes easily to 2-D ; the only difference is that the matrix is circulant-block-circulant instead of circulant. However, there is an ambiguity in recovering (i,j) from a single eigenvalue exponent. This is called the "matching problem."

B. Derivation of Prony

Also, note that if the size of T_0 exceeds the number of non-zero $x(n)$ by one, we have

$$T_0 a = V^H D V a \rightarrow V a = 0 \quad (8)$$

This states that the zeros of the polynomial with coefficients a_n are $\{e^{-j2\pi n/N}, n \in \Theta\}$. Thus we can determine the locations of the non-zero $x(n)$ by computing the null vector a of T_0 and rooting the polynomial having coefficients a_n (Prony's method).

A better procedure, if additional data are available, is to compute the null vector of the covariance matrix

$$(T_0^H T_0)a = T_0^H = 0 \tag{9}$$

since this tends to collect additive zero-mean white noise in the data as $\sigma^2 I$. Thus, we can compute the minimum eigenvector of $(T_0^H T_0)$, or equivalently the minimum singular vector of T_0, and use that as an estimate of a (the basic idea behind Pisarenko's method).

There are many other methods of 1-D harmonic analysis; we will make no attempt to review them all. However, note that Prony's and Pisarenko's method do not extend directly to 2-D, since there is no 2-D version of the fundamental theorem of algebra, so that root-based methods cannot be used directly on 2-D problems.

ESPRIT avoids this problem, but it introduces the "matching problem" of associating the components of 2-D frequencies along each axis. This can be avoided if the 2-D data windown (2-D array configuration) is centrosymmetric by transforming the complex problem into a real-valued one, but that is not applicable to the present problem unless data $X(k_1, k_2)$ are known in a rectangular region, which is not always the case (e.g., limited-angle tomography).

Next, we show how to unwrap the 2-D problem into a 1-D problem, allowing use of 1-D methods.

III. REFORMULATION OF 2-D AS 1-D

A. Good-Thomas FFT

The Good-Thomas FFT reformulates a $(N_1 \times N_2)$-point 2-D DFT as an $(N_1 N_2)$-point 1-D DFT using a residue number system (RNS) indexing scheme. It requires N_1 and N_2 to be relatively prime. Here we derive it for specifically $N_1 = N - 1$ and $N_2 = N$. The exponent in the 2-D DFT (3) is

$$\begin{aligned}
\frac{i_1 k_1}{N-1} + \frac{i_2 k_2}{N} &= \frac{N i_1 k_1 + (N-1) i_2 k_2}{N(N-1)} \\
&\equiv \pmod 1 \quad \frac{[\, i_1 \quad i_2 \,] \begin{bmatrix} N & 0 \\ 0 & N-1 \end{bmatrix} \begin{bmatrix} k_1 \\ k_2 \end{bmatrix}}{N(N-1)} \\
&\quad + \frac{N(N-1)\,[\, i_1 \quad i_2 \,] \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix} \begin{bmatrix} k_1 \\ k_2 \end{bmatrix}}{N(N-1)} \\
&= \frac{[\, i_1 \quad i_2 \,] \begin{bmatrix} N \\ 1-N \end{bmatrix} [\, N \quad N-1 \,] \begin{bmatrix} k_1 \\ k_2 \end{bmatrix}}{N(N-1)} = ik
\end{aligned} \tag{10}$$

$$\text{for} \quad i = \quad N i_1 - (N-1) i_2$$

$$\text{and} \quad j = \quad N k_1 + (N-1) k_2 \tag{11}$$

To obtain (i_1, i_2) from i, we use the following:

$$\begin{aligned}
i &\equiv i_1 \pmod{N-1} \\
i &\equiv i_2 \pmod{N}
\end{aligned} \tag{12}$$

which we recognize as the residue number system (RNS) mapping.

B. Dual 2-D Harmonic Retrieval

Now define the region ω as having the same shape as Ω but scaled by $1/2$ in each dimension. The area of ω is $1/4$ the area of Ω; this equals the area of Θ. Define $s(i, j)$ using

$$\begin{cases} s(i,j) = 0 & \text{for } (i,j) \in \Theta; \\ S(k_1, k_2) = 0 & \text{for } (k_1, k_2) \notin \omega \end{cases} \tag{13}$$

After unwrapping, we can provide an explicit formula for $s(i, j)$. For now, note that there are just enough linear equations to specify $s(i, j)$ uniquely, except for a scale factor. We now have

$$x(i,j)s(i,j) = 0 \rightarrow X(k_1, k_2) ** S(k_1, k_2) = 0 \tag{14}$$

where $**$ denotes 2-D cyclic convolution. Since:

- $X(k_1, k_2)$=known for $(k_1, k_2) \in \Omega$
- $S(k_1, k_2) \neq 0$ only for $(k_1, k_2) \in \omega$
- ω has the same shape as Ω
- ω is $1/4$ the area of Ω

$X(k_1, k_2) ** S(k_1, k_2) = 0$ is a linear system of equations that determines $S(k_1, k_2)$ to a scale factor.

This is the dual 2-D harmonic retrieval problem. The problem is that while $S(k_1, k_2)$ encodes the locations (i_k, j_k) of the non-zero image pixels, it is not evident how to extract that information, since there is no 2-D version of the fundamental theorem of algebra. This is why 2-D harmonic retrieval is harder than 1-D harmonic retrieval, which can be solved by simply computing the zeros of a polynomial.

C. Unwrapping 2-D to 1-D

In present context the solution is evident: unwrap the 2-D problem to a 1-D problem. Repeating the above 2-D exposition in 1-D, we define $s(n)$ using

$$\begin{cases} s(n) = 0 & \text{for } n \in \Theta'; \\ S(k) = 0 & \text{for } k \notin \omega' \end{cases} \tag{15}$$

where Θ', Ω' and ω' are the unwrapped versions of Θ, Ω and ω, respectively. We now have

$$x(n)s(n) = 0 \rightarrow X(k) * S(k) = 0 \tag{16}$$

where $*$ denotes 1-D cyclic convolution. This is a linear system of equations $Xs = 0$ with Toeplitz blocks for the nonzero values of $S(k)$. Solving this system determines $S(k)$ to a scale factor.

The difference from the 2-D case is that $S(k)$ directly determines the locations (i_k, j_k) of the non-zero image pixels–we need only compute the zeros of the polynomial having $S(k)$ as coefficients. While rooting a polynomial is far from trivial, we are searching specifically only for zeros on the unit circle.

Alternatively, we can avoid polynomial rooting in 1-D problems by noting that the Toeplitz matrix X is singular, since it has a non-zero null vector s. The Toeplitz matrix can be extended by noting that every $(M \times M)$ minor of a matrix of rank M is zero. This can be used to extrapolate $S(k)$ until all its values are known. An inverse DFT of $S(k)$ yields $s(n)$.

Another alternative, if more data are available, is to use Pisarenko's method instead of Prony's method:

$$(X^H X)s = X^H 0 = 0 \tag{17}$$

This tends to collect any additive white noise in an additive term $\sigma^2 I$, so that computing the eigenvector associated with the minimum eigenvalue of $(X^H X)$ (equivalently, the minimum singular vector of X) yields a noise-resistant estimate of s. We will not attempt to discuss here all of the methods for 1-D harmonic retrieval. Also note the following:

- Unwrapped $X(k)$ has strings of unknown values
- Unwrapped $S(k)$ has strings of zeros
- Unwrapped $X(k) * S(k)$ is known=zero in strings
- Hence the linear system has Toeplitz blocks:
  - Many matrix columns are multiplied by zero;
  - Many matrix rows are discarded, since many rows lead to an unknown right side of the equation.
- The 2-D cyclic convolution can be unwrapped to a 1-D cyclic convolution directly using Agarwal-Cooley convolution, which uses the same RNS mapping as the Good-Thomas FFT.

IV. NUMERICAL EXAMPLES

A. Example #1: Small 1-D Example

We use a small 1-D example to illustrate how a piecewise-constant signal can be reconstructed from a few DFT values. We are given the following 32-point 1-D DFT values of a piecewise-constant signal known to be zero at its endpoints:

$$\begin{array}{llll} X_0 = & 106.00 & X_1 = & 7.62 + j37.04 \\ X_2 = & -28.60 + j24.14 & X_3 = & -9.89 - j10.55 \\ X_4 = & -26.73 - j11.41 & X_5 = & -7.04 - j8.74 \\ X_6 = & 1.39 - j4.60 & X_7 = & -7.56 - j1.62 \end{array} \tag{18}$$

The DFTs of the differences $x(n) - x(n-1)$ are computed from these given DFT values by multiplying X_k by $(1 - e^{-j2\pi k/32})$. Using these new values, we set up the following Hermitian Toeplitz matrix:

$$\begin{bmatrix} X_0 & X_1 & X_2 & X_3 & X_4 & X_5 & X_6 & X_7 \\ X_{-1} & X_0 & X_1 & X_2 & X_3 & X_4 & X_5 & X_6 \\ X_{-2} & X_{-1} & X_0 & X_1 & X_2 & X_3 & X_4 & X_5 \\ X_{-3} & X_{-2} & X_{-1} & X_0 & X_1 & X_2 & X_3 & X_4 \\ X_{-4} & X_{-3} & X_{-2} & X_{-1} & X_0 & X_1 & X_2 & X_3 \\ X_{-5} & X_{-4} & X_{-3} & X_{-2} & X_{-1} & X_0 & X_1 & X_2 \\ X_{-6} & X_{-5} & X_{-4} & X_{-3} & X_{-2} & X_{-1} & X_0 & X_1 \\ X_{-7} & X_{-6} & X_{-5} & X_{-4} & X_{-3} & X_{-2} & X_{-1} & X_0 \end{bmatrix} \tag{19}$$

and compute its null vector

$$[\, A_7 \quad A_6 \quad A_5 \quad A_4 \quad A_3 \quad A_2 \quad A_1 \quad A_0 \,]' \tag{20}$$

This (8×8) matrix is found to have rank=7, so there are only seven jumps in $x(n)$. We obtain

$$\begin{array}{llll} A_0 = & -0.2085 + 0.5034j & A_1 = & -0.1047 - 0.3882j \\ A_2 = & 0.0972 + 0.0496j & A_3 = & -0.1174 - 0.1255j \\ A_4 = & 0.0710 + 0.1565j & A_5 = & -0.0086 - 0.1088j \\ A_6 = & 0.3186 + 0.2453j & A_7 = & -0.5449 + j0.0000 \end{array} \tag{21}$$

The polynomial equation

$$A_0 z^7 + A_1 z^6 + \ldots + A_6 z + A_7 = 0 \tag{22}$$

has the following roots:

$$\{e^{j2.36}, e^{-j2.75}, e^{j1.37}, e^{-j1.96}, e^{j.40}, e^{-j.98}, e^{-j.39}\} \tag{23}$$

and $-\frac{32}{2\pi}$ times the phases of these roots is

$$\{-12, 14, -7, 10, -2, 5, 2\} \equiv \{2, 5, 10, 14, 20, 25, 30\} \tag{24}$$

These specify the locations of the jumps in $x(n)$.

To compute the actual values of $x(n)$, we solve the following (7×7) linear system of equations

$$\begin{bmatrix} e^{-j\frac{2*pi}{32}(1)(2)} & e^{-j\frac{2*pi}{32}(1)(5)} & e^{-j\frac{2*pi}{32}(1)(10)} \\ e^{-j\frac{2*pi}{32}(2)(2)} & e^{-j\frac{2*pi}{32}(2)(5)} & e^{-j\frac{2*pi}{32}(2)(10)} \\ e^{-j\frac{2*pi}{32}(3)(2)} & e^{-j\frac{2*pi}{32}(3)(5)} & e^{-j\frac{2*pi}{32}(3)(10)} \\ e^{-j\frac{2*pi}{32}(4)(2)} & e^{-j\frac{2*pi}{32}(4)(5)} & e^{-j\frac{2*pi}{32}(4)(10)} \\ e^{-j\frac{2*pi}{32}(5)(2)} & e^{-j\frac{2*pi}{32}(5)(5)} & e^{-j\frac{2*pi}{32}(5)(10)} \\ e^{-j\frac{2*pi}{32}(6)(2)} & e^{-j\frac{2*pi}{32}(6)(5)} & e^{-j\frac{2*pi}{32}(6)(10)} \\ e^{-j\frac{2*pi}{32}(7)(2)} & e^{-j\frac{2*pi}{32}(7)(5)} & e^{-j\frac{2*pi}{32}(7)(10)} \end{bmatrix} \ldots$$

$$\cdots \begin{bmatrix} e^{-j\frac{2*pi}{32}(1)(20)} & e^{-j\frac{2*pi}{32}(1)(25)} & e^{-j\frac{2*pi}{32}(1)(30)} \\ e^{-j\frac{2*pi}{32}(2)(20)} & e^{-j\frac{2*pi}{32}(2)(25)} & e^{-j\frac{2*pi}{32}(2)(30)} \\ e^{-j\frac{2*pi}{32}(3)(20)} & e^{-j\frac{2*pi}{32}(3)(25)} & e^{-j\frac{2*pi}{32}(3)(30)} \\ e^{-j\frac{2*pi}{32}(4)(20)} & e^{-j\frac{2*pi}{32}(4)(25)} & e^{-j\frac{2*pi}{32}(4)(30)} \\ e^{-j\frac{2*pi}{32}(5)(20)} & e^{-j\frac{2*pi}{32}(5)(25)} & e^{-j\frac{2*pi}{32}(5)(30)} \\ e^{-j\frac{2*pi}{32}(6)(20)} & e^{-j\frac{2*pi}{32}(6)(25)} & e^{-j\frac{2*pi}{32}(6)(30)} \\ e^{-j\frac{2*pi}{32}(7)(20)} & e^{-j\frac{2*pi}{32}(7)(25)} & e^{-j\frac{2*pi}{32}(7)(30)} \end{bmatrix}$$

$$\times \begin{bmatrix} x(2)-x(1) \\ x(5)-x(4) \\ x(10)-x(9) \\ x(14)-x(13) \\ x(20)-x(19) \\ x(25)-x(24) \\ x(30)-x(29) \end{bmatrix} = \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \\ X_5 \\ X_6 \\ X_7 \end{bmatrix} ; \begin{bmatrix} x(2)-x(1) \\ x(5)-x(4) \\ x(10)-x(9) \\ x(14)-x(13) \\ x(20)-x(19) \\ x(25)-x(24) \\ x(30)-x(29) \end{bmatrix} = \begin{bmatrix} 3 \\ -2 \\ 3 \\ -3 \\ 4 \\ 4 \\ -9 \end{bmatrix} \tag{25}$$

Since we are given that $x(0) = x(31) = 0$ we have

$$x(n) = \{00 \underbrace{3}_{3} \underbrace{1}_{5} \underbrace{4}_{4} \underbrace{1}_{6} \underbrace{5}_{5} \underbrace{9}_{5} 00\} \tag{26}$$

Matlab code for implementing this example:

```
X=[0 0 3 3 3 1 1 1 1 1 4 4 4 4 1 1 1 1 1 1];
X=[X 5 5 5 5 5 9 9 9 9 9 0 0];
FX=fft(X);F=FX.*(1-exp(-j*2*pi*[0:31]/32));
T=toeplitz(F(1:8)',F(1:8));A=flipud(null(T));
R=roots(A);N=-32*angle(R)/2/pi;
M=exp(-j*2*pi*[0:7]'*N/32);XX=M\F([1:8]).';
```

B. Example #2: Small 2-D Example

We use a small limited-angle tomography problem to illustrate the 2-D procedure. We are given these values of the (10×9) 2-D DFT of a sparse image:

$$\begin{bmatrix} X_{00} & X_{11} & X_{22} \\ X_{01} & X_{12} & X_{23} \\ X_{02} & X_{13} & X_{24} \\ X_{03} & X_{14} & X_{33} \\ X_{04} & X_{44} & X_{34} \end{bmatrix} =$$

$$\begin{bmatrix} 14+j21 & -11.11+j17.33 & -4.65+j13.66 \\ -4.8-j1.27 & -11.78+j5.52 & 2.39-j3.04 \\ -16.59-j7.7 & -9.8-j1.52 & -3.46-j14.27 \\ -1.23-j6.87 & 4.13+j4.75 & -19.23+j.339 \\ -.716-j7.13 & 5.09-j4.81 & -7.39+j11 \end{bmatrix} \tag{27}$$

Note these values form a wedge shape in the Fourier plane, as in a limited-angle tomography problem. The complex conjugate values are unknown, since the image is complex valued.

All but five of the image pixel values are zero. The goal is to compute the locations and values of those five non-zero pixels.

For this example, we have:

- Θ=locations of non-zero values of $x(i,j)$;
- Θ has 5 elements but is otherwise unknown.
- Ω=locations of known values of $X(k_1,k_2)$;
- Ω has 15 elements and is half of a (5×5) square.
- ω=locations of non-zero values of $S(k_1,k_2)$;
- ω has 6 elements and is half of a (3×3) square.

The equation $X(k_1,k_2) * *S(k_1,k_2) = 0$ becomes the linear system of equations

$$\begin{bmatrix} X_{00} & X_{11} & X_{22} & X_{01} & X_{12} & X_{02} \\ X_{11} & X_{22} & X_{33} & X_{12} & X_{23} & X_{13} \\ X_{22} & X_{33} & X_{44} & X_{23} & X_{34} & X_{24} \\ X_{01} & X_{12} & X_{23} & X_{02} & X_{13} & X_{03} \\ X_{12} & X_{23} & X_{34} & X_{13} & X_{24} & X_{14} \\ X_{02} & X_{13} & X_{24} & X_{03} & X_{14} & X_{04} \end{bmatrix} \begin{bmatrix} S_{00} \\ S_{11} \\ S_{22} \\ S_{01} \\ S_{12} \\ S_{02} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{28}$$

Here we have pre-reversed (k_1,k_2) in k_1 and k_2 to make the 2-D convolution easier to envision. Hence the matrix have Hankel blocks, rather than Toeplitz blocks. The reader should unwrap the 2-D convolution into a 1-D convolution to envision how the convolution works in 1-D.

The unwrapped $S(k)$ consists of the just-computed values of $S(k_1,k_2)$ with bands of zeros inserted:

$$S(k) = \{S_{00}, S_{11}, S_{22} \underbrace{,0,}_{7} S_{01}, S_{12} \underbrace{,0,}_{8} S_{02}\} \tag{29}$$

Next, we compute the zeros of the polynomial

$$S_{00}z^{20} + S_{11}z^{19} + S_{22}z^{18} + S_{01}z^{10} + S_{12}z^{9} + S_{02} = 0 \tag{30}$$

Note that since the coefficients $S(k_1,k_2)$ are complex, polynomial zeros do not occur in complex conjugate pairs. 5 of the 20 zeros of this polynomial lie on the unit circle. These zeros are:

$$\{e^{-j1.047}, e^{-j0.768}, e^{-j3.002}, e^{-j3.072}, e^{-j0.070}\} \tag{31}$$

Multiplying the arguments by $\frac{90}{2\pi}$ yields the integers

$$\{-15, -11, -43, -44, -1\} \tag{32}$$

These rewrap to $x(i,j)$ by computing residues mod(10) and mod(9), respectively, and reversing i (due to the combination of reversing $S(k_1,k_2)$ and exchanging the spatial and wavenumber domains).

The locations, and their associated values (whose computation is not shown here), turn out to be

$$\begin{aligned} x(3,2) = 3+j2; x(1,7) = 1+j7; x(4,1) = 4+j1 \\ x(1,8) = 1+j8; \qquad x(5,3) = 5+j3 \end{aligned} \tag{33}$$

Matlab code for implementing this example:

```
X(4,3)=3+2j;X(2,8)=1+7j;X(5,2)=4+j;X(2,9)=1+8j;
X(6,4)=5+3j;X(10,9)=0; F=fft2(X);
T1=[F(1,1) F(2,2) F(3,3) F(1,2) F(2,3) F(1,3)];
```

```
T2=[F(2,2) F(3,3) F(4,4) F(2,3) F(3,4) F(2,4)];
T3=[F(3,3) F(4,4) F(5,5) F(3,4) F(4,5) F(3,5)];
T4=[F(1,2) F(2,3) F(3,4) F(1,3) F(2,4) F(1,4)];
T5=[F(2,3) F(3,4) F(4,5) F(2,4) F(3,5) F(2,5)];
T6=[F(1,3) F(2,4) F(3,5) F(1,4) F(2,5) F(1,5)];
T=[T1;T2;T3;T4;T5;T6];N=null(T);N=N.';
A=[N(1:3) zeros(1,7) N(4:5) zeros(1,8) N(6)];
R=roots(A);I=45/pi*angle(R(abs(abs(R)-1)<.001));
I1=11-mod(I,10);I2=mod(I,9)+1;[I1 I2]
```

C. Example #3: Generalized Eigenvalue Problem

We now redo the above 2-D example using 2-D unitary ESPRIT. Since the region of known values of $X(k_1, k_2)$ is not centrosymmetric, the matching problem arises. We present a new procedure that shows how to solve the matching problem.

First, the data given above is not sufficient to solve the problem. We need the additional data points:

$$\begin{array}{ll} X_{05} = -7.55 + j1.01 & X_{15} = 4.53 - j4.41 \\ X_{25} = -11.8 + j2.76 & X_{35} = 6.21 + j5.40 \\ X_{45} = -.825 - j.819 & X_{55} = -.65 - j.151 \end{array} \quad (34)$$

This is still a limited-angle problem, but the maximum wavenumber of the data is larger by one. This is another reason for favoring our previous method: We now require 21/90 DFT values, instead of 15/90.

With this additional data, we can now form both

$$T_0 = \begin{bmatrix} X_{00} & X_{11} & X_{22} & X_{01} & X_{12} & X_{02} \\ X_{11} & X_{22} & X_{33} & X_{12} & X_{23} & X_{13} \\ X_{22} & X_{33} & X_{44} & X_{23} & X_{34} & X_{24} \\ X_{01} & X_{12} & X_{23} & X_{02} & X_{13} & X_{03} \\ X_{12} & X_{23} & X_{34} & X_{13} & X_{24} & X_{14} \\ X_{02} & X_{13} & X_{24} & X_{03} & X_{14} & X_{04} \end{bmatrix} \quad (35)$$

$$T_1 = \begin{bmatrix} X_{11} & X_{22} & X_{33} & X_{12} & X_{23} & X_{13} \\ X_{22} & X_{33} & X_{44} & X_{23} & X_{34} & X_{24} \\ X_{33} & X_{44} & X_{55} & X_{34} & X_{45} & X_{35} \\ X_{12} & X_{23} & X_{34} & X_{13} & X_{24} & X_{14} \\ X_{23} & X_{34} & X_{45} & X_{24} & X_{35} & X_{25} \\ X_{13} & X_{24} & X_{35} & X_{14} & X_{25} & X_{15} \end{bmatrix} \quad (36)$$

Solving the generalized eigenvalue problem for these two matrices yields the eigenvalues

$$\{e^{-j1.047}, e^{-j0.768}, e^{-j3.002}, e^{-j3.072}, e^{-j0.070}, 0\} \quad (37)$$

Multiplying the arguments of the non-zero eigenvalues by $\frac{90}{2\pi}$ yields the integers

$$\{-15, -11, -43, -44, -1\} \quad (38)$$

and from this point we proceed as above.

The zero eigenvalue means that we could have recovered a sixth non-zero $x(i, j)$. However, with more data $\{X_{k6}, 0 \le k \le 6\}$ (28/90 data points), we may recover 1+2+3+4-1=9 non-zero $x(i, j)$ using the 1-D method, which requires only rooting a polynomial, instead of a generalized eigenvalue problem.

Matlab code for implementing this example:

```
X(4,3)=3+2j;X(2,8)=1+7j;X(5,2)=4+j;X(2,9)=1+8j;
X(6,4)=5+3j;X(10,9)=0; F=fft2(X);
T1=[F(1,1) F(2,2) F(3,3) F(1,2) F(2,3) F(1,3)];
T2=[F(2,2) F(3,3) F(4,4) F(2,3) F(3,4) F(2,4)];
T3=[F(3,3) F(4,4) F(5,5) F(3,4) F(4,5) F(3,5)];
T4=[F(1,2) F(2,3) F(3,4) F(1,3) F(2,4) F(1,4)];
T5=[F(2,3) F(3,4) F(4,5) F(2,4) F(3,5) F(2,5)];
T6=[F(1,3) F(2,4) F(3,5) F(1,4) F(2,5) F(1,5)];
T=[T1;T2;T3;T4;T5;T6];
U1=[F(2,2) F(3,3) F(4,4) F(2,3) F(3,4) F(2,4)];
U2=[F(3,3) F(4,4) F(5,5) F(3,4) F(4,5) F(3,5)];
U3=[F(4,4) F(5,5) F(6,6) F(4,5) F(5,6) F(4,6)];
U4=[F(2,3) F(3,4) F(4,5) F(2,4) F(3,5) F(2,5)];
U5=[F(3,4) F(4,5) F(5,6) F(3,5) F(4,6) F(3,6)];
U6=[F(2,4) F(3,5) F(4,6) F(2,5) F(3,6) F(2,6)];
U=[U1;U2;U3;U4;U5;U6];45/pi*angle(eig(T,U))
```