A Non-Iterative Procedure for Sparse Solutions to Linear Equations with Bandlimited Rows

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

Abstract— **The problem of computing sparse (mostly zero) solutions to underdetermined linear systems of equations has received much attention recently, due to its applications to compressed sensing. Under mild assumptions, the sparsest solution has minimum-L1-norm, and can be computed using linear programming. We present a non-iterative algorithm for this problem that requires only that each row of the system matrix be bandlimited to the aspect ratio of the matrix. The algorithm can be used directly with linear-time-invariant sparsifying operators, and with wavelet transforms if the matrix is more bandlimited. A numerical example and code illustrate the algorithm.**

Keywords— **Sparse reconstruction**
Phone: 734-763-9810. Fax: 734-763-1503.
Email: aey@eecs.umich.edu. EDICS: 2-REST.

I. INTRODUCTION

A. Problem Statement

The goal is to solve underdetermined linear system

$$y = \tilde{H}x = Hz; \quad H = \tilde{H}W^{-1}; z = Wx \qquad (1)$$

where it is assumed that

- $\tilde{H}$ and H=$\tilde{H}W^{-1}$ have rank=M and M×N,N>M;
- W is an N×N circulant matrix that sparsifies the unknown column N-vector x to vector z;
- z has at most K out of N non-zero elements;
- $y, \tilde{H}, W, H, K, M, N$ are known; x, z unknown.
- Each row of $\tilde{H}$ is bandlimited to $\pi\frac{M}{N}$; $\quad K < \frac{M}{2}$.

For the sparsifying transformation W we use a gradient edge detector. The procedure is adapted to wavelet transforms in Section III below. Of course, if the unknown x is itself known to be sparse, as in X-ray crystallography, then we can use $W = I$.

If the problem is only slightly underdetermined (N-M<<N), then a different approach that does not require bandlimited rows of H can be used [1]. We have also used related but different approaches to X-ray crystallography [2] and to the limited-angle tomography formulation of SAR [3].

B. Relevant Previous Approaches

Use of sparseness as side information in signal reconstruction goes back at least as far as 1979 [4]. Most early work considered the deconvolution of sparse 1D spike trains arising in reflections from layered media in seismic exploration, although [5] considered 1D reconstruction from bandlimited data. The first use of sparseness in 2D (image) reconstruction of which we are aware is [6]. All of this early work minimized the ℓ_1 norm (sum of absolute values) of the signal, using linear programming. The idea was that the ℓ_1 norm solution lies on a vertex of the simplex and so is sparse. This idea has recently been put on a firmer theoretical ground in [7] and other recent papers. More recent work using this approach is [8]-[10]. Using the ℓ_1 norm for the signal and ℓ_2 for the error is called LASSO. The problem with this approach is the amount of computation required by linear programming for image reconstruction.

Another way of formulating the linear sparse reconstruction problem is as a matrix "subset selection" problem [11]-[14]. The forward greedy algorithm successively selects the matrix column closest (in the mean-square sense) to the residual error resulting from the previous matrix column selections. The backward greedy algorithm starts with a general solution and successively removes the matrix column that increases the mean-square residual error the least. The latter algorithm has been shown to give the correct answer if the noise level is sufficiently small [13]. However, again the problem is the amount of computation required here by subset selection.

Still another recent approach is to include a thresholding constraint in a Landweber-like iterative algorithm [15]-[18], often arising from statistical image priors that implicitly (but not explicitly) maximize sparsity. This is a straightforward approach, and unlike the above methods requires reasonable computation for 2D problems. However, even in the absence of noise, convergence to an optimally-sparse solution is not in general guaranteed for these algorithms.

C. New Approach of This Paper

All of the above approaches are iterative, requiring many iterations, are computationally intensive, or both. The approach of this paper requires only:

- Discrete Fourier transforms of each row of $\tilde{H}$;
- Solution of an M×M linear system of equations;

- The nullspace of a K×K Toeplitz matrix;
- Solution of an M×K linear system of equations.

If $\tilde{H}$ and W are set in advance, then only two linear systems, both with K unknowns, need to be solved. This is a major advantage if $K << M$, as is often the case. These savings also apply to deconvolution problems, as the first example below shows.

Note that all of these operations are straightforward linear algebra and can be implemented in a non-iterative way. Of course, iterative methods such as conjugate gradient may be used in some steps, but the overall algorithm is non-iterative, and can even be considered to be a closed-form computation.

This paper is organized as follows. Section II derives the algorithm for $W = I$. Section III extends it to circulant W representing linear-time-invariant operators, and to wavelet transforms W; for wavelets the rows of $\tilde{H}$ must now be more bandlimited. Section IV provides an illustrative numerical example.

II. Derivation of New Algorithm for W=I

In this section the problem is to reconstruct a K-sparse signal z from its bandlimited and underdetermined linear transformation y=Hz. This has evident applications in reconstruction of sparse signals.

Define the 1-D DFTs of z and each row of H as

$$\hat{H}_{ik} = \sum_{n=1}^{N} H_{in} e^{-j2\pi(n-1)(k-1)/N}, 1 \le k \le N \quad (2)$$

$$\hat{Z}_k = \sum_{n=1}^{N} z_n e^{-j2\pi(n-1)(k-1)/N}; 1 \le k \le N \quad (3)$$

By Parseval's theorem, we have

$$y_i = \sum_{n=1}^{N} H_{in} z_n = \sum_{k=1}^{N} \hat{H}_{ik} \hat{Z}_k \quad (4)$$

Since each row of H is bandlimited to $\pi\frac{M}{N}$, this becomes the M×M linear system of equations

$$\begin{bmatrix} y_1 \\ \vdots \\ y_M \end{bmatrix} = \begin{bmatrix} \cdots & \hat{H}_{1,\frac{M}{2}} & \hat{H}_{1,N-\frac{M}{2}} & \cdots \\ \cdots & \vdots & \vdots & \cdots \\ \cdots & \hat{H}_{M,\frac{M}{2}} & \hat{H}_{M,N-\frac{M}{2}} & \cdots \end{bmatrix} \begin{bmatrix} \vdots \\ \hat{Z}_{\frac{M}{2}} \\ \hat{Z}_{N-\frac{M}{2}} \\ \vdots \end{bmatrix}$$

Solution yields the lowest $\frac{M}{2}$ frequencies $\hat{Z}_k$ of z.

Next, form the N×N circulant matrix C whose first row is $\{\hat{Z}_k, 1 \le k \le N\}$. The eigenvalues of C are the values of z_i. Since only K of these values are non-zero, C has rank K, and there exists a column vector v of length K+1 such that

$$0 = C \begin{bmatrix} v \\ 0 \end{bmatrix} = \begin{bmatrix} \hat{Z}_1 & \cdots & \hat{Z}_{K+1} \\ \ddots & \ddots & \ddots \\ \hat{Z}^*_{K+1} & \cdots & \hat{Z}_1 \end{bmatrix} v \quad (5)$$

If $K < \frac{M}{2}$, then all of its elements are known.

Next, make the following definitions:

$$\{z_{i_n}, 1 \le n \le K\} = \{z_i : z_i \ne 0, 1 \le i \le N\} \quad (6)$$

$$[F_{ik}] = [e^{-j2\pi(i-1)(k-1)/N}]_{ik}, 1 \le i, k \le N \quad (7)$$

$$[\tilde{F}_{ik}] = [e^{-j2\pi(i-1)(n_k-1)/N}]_{ik}, 1 \le k \le K \quad (8)$$

- $\{z_{i_n}, 1 \le n \le K\}$ are the nonzero values of $\{z_i\}$;
- F is the DFT matrix implementing $\hat{Z} = Fz$;
- $\tilde{F}$ is a tall N×K submatrix of F.

Then we have the (N×K)(K×N) factorization

$$C = F\text{diag}[z_i]F^H = \tilde{F}\text{diag}[z_{i_n}]\tilde{F}^H \quad (9)$$

Extending $\tilde{F}$ by any other column of F gives

$$0 = C \begin{bmatrix} v \\ 0 \end{bmatrix} = \tilde{F}\text{diag}[z_{i_n}\ 0]\tilde{F}^H \begin{bmatrix} v \\ 0 \end{bmatrix} \quad (10)$$

Since $\tilde{F}$ is a tall matrix and $z_{i_n} \ne 0$, we have

$$\tilde{F}^H \begin{bmatrix} v \\ 0 \end{bmatrix} = 0 \rightarrow F^H \begin{bmatrix} v \\ 0 \end{bmatrix} = 0 \text{ for } i \in \{i_n\} \quad (11)$$

This shows that an inverse N-point DFT of a zero-padded v is zero at the locations $\{i_n\}$ of nonzero z_i. The conditioning of the location problem is determined by the condition number of $\tilde{F}$. Random distribution of $\{i_n\}$ yields good conditioning; clumping of $\{i_n\}$ and large gaps yields poor conditioning.

III. Circulant and Wavelet Sparsification

A. Sparsification by Circulant Matrix

We now consider the problem

$$y = \tilde{H}x = Hz; \quad H = \tilde{H}W^{-1}; \quad z = Wx \quad (12)$$

- Unknown x is *not* sparse but is sparsifiable to z;
- The rows of $\tilde{H}$ are bandlimited to $\pi\frac{M}{N}$;
- W is an N×N circulant matrix that:
  - Implements a linear-time-invariant sparsifier;
  - A 2-D difference operator is used in the example;
  - The first row of W is the impulse response h_n;
  - The first row of W^{-1} is the impulse response g_n;

– $h_n * g_n = \delta_n$ where $*$ denotes cyclic convolution.

Then $H{=}\tilde{H}W^{-1}$ implements a cyclic convolution of g_n with each row of $\tilde{H}$. Since each row of $\tilde{H}$ is bandlimited, each row of H is also bandlimited, and the problem reduces to the case $W = I$ treated above.

Often the DFT of h_n will be zero at some low frequencies. For example, the DFT of a difference operator is zero at DC (k=0). g_n is then computed assuming its DFT is also zero there, and the reconstructed x_n is filtered to zero at those frequencies. The correct x_n can be computed using side information that a few values of x_n are known to be zero.

Otherwise, the only addition is to perform a cyclic convolution of each row of $\tilde{H}$ with g_n; this is easily implented using N^{th}-order DFTs. See the example.

B. Sparsification by Wavelet Transform

This procedure can still be applied if the wavelet transform is used for sparsification of x. However, each row of $\tilde{H}$ must now be bandlimited to $\frac{\pi}{2^L}\frac{M}{N}$; L is the number of scales used for wavelet dilation.

The discrete-time wavelet transform consists of:

- The low-pass filter *scaling* function g_n;
- The high-pass filter *wavelet* function h_n;
- $g_n = (-1)^n h_n$; both functions have finite support.

The function x_n to be sparsified is:

- Filtered separately by both g_n and h_n;
- Both outputs are downsampled;
- The downsampled output of g_n is then filtered in the same way x_n was filtered;
- This is continued through a total of L dilations.

The wavelet transform can then be implemented by:

1. Matrix-vector multiplication $\vec{Y} = C\vec{X}$
2. where $(2^L N \times 2^L N)$ matrix C is circulant;
3. 2^{Lth} row of C is the *dilated* wavelet h_n;
4. $\vec{X}'{=}[\vec{X_0}, \vec{X_1}, \vec{X_2} \ldots \vec{X_L}]$ where
5. $\vec{X_0}{=}[x_1, x_2 \ldots x_{N/2}, -x_{N/2+1} \ldots - x_N]$;
6. $\vec{X_1}{=}[x_1, x_2, x_3 \ldots x_N]$;
7. $\vec{X_2}{=}[x_1, 0, x_2, 0, x_3, 0, x_4 \ldots x_N]$;
8. $\vec{X_3}{=}[x_1, 0, 0, 0, x_2, 0, 0, 0, x_3 \ldots x_N]$;
9. $\vec{X_i}$ has length $2^{i-1}N, i \neq 0$, so that;
10. $\vec{X}$ has length N+N+2N+4N+8N+$\ldots 2^{L-1}N{=}2^L N$;
11. Wavelet transform is every $(2^L)^t h$ sample of $\vec{Y}$.

Write $W = UCV'$ where U and V have forms like

$$U = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{13}$$

$$V = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{14}$$

For example, the Haar wavelet transform of a signal of length=8 and 3 scales is implemented by multiplication by the 8×8 matrix

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & -1 & -1 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix}$$

Multiplication by this matrix can also be implemented using a circulant matrix, as follows:

1. 8^{th} row is $[1, 1, 1, 1, -1, -1, -1, -1]'$;
2. Size $2^L N = 2^3 8 = 64$. Multiply by $\vec{X}$=:
3. $[x_1 \ldots x_4, -x_5 \ldots - x_8, x_1 \ldots x_8]$ augmented to
4. $[x_1, 0, x_2, 0, x_3 \ldots x_8, x_1, 0, 0, 0, x_2, 0, 0, 0 \ldots]$
5. Note that $\vec{X}$ has length 8+8+16+32=64
6. Taking elements #{8,16,24,32,40,48,56,64} of $\vec{Y}$.
7. Add scale factors 1/2 and $1/\sqrt{2}$ at the end.

Matlab code implementing this example is given in the Appendix. The various matrices can be examined.

Since W is orthogonal, $W^{-1} = W'$ and

$$H = \tilde{H}W^{-1} = \tilde{H}W' = \tilde{H}VC'U' \tag{15}$$

Now rewrite the original problem as

$$y = \tilde{H}x = Hz = (\tilde{H}V)C'(U'z) \tag{16}$$

and note that

- $U'z$ is K-sparse since z is K-sparse;
- $\tilde{H}V$ has the form $[\tilde{H}_1 \ldots \tilde{H}_N | \tilde{H}_1 0 \tilde{H}_2 0 \ldots]$
- Each row of $\tilde{H}V$ has the form of $\vec{X}'$ above.

This shows that we can apply the procedure, provided each row of $\tilde{H}$ is bandlimited to $\frac{\pi}{2^L}\frac{M}{N}$. Note that the zero-stuffing of sections of each row of $\tilde{H}$ introduce copies of its spectrum, but this is matched by the similar zero-stuffing of z, so the lowest M frequencies of z can still be found.

IV. Numerical Example

The computational savings are illustrated by deconvolution of a 72×72 block image x_{ij} from its undersampled cyclic convolution with a 2-D Gaussian point-spread-function. We have the following:

- 72^2=5184 unknown image pixels x_{ij};
- 36^2=1296 known undersampled data y_{ij};
- The data are blurred and low-pass filtered;
- The image is known to be sparsifiable to z_{ij} by
- convolution with 2-D filter [1 -1;-1 1]

The lowest frequencies of the sparsified image z_{ij} in a 19×19 square centered at DC were computed from the downsampled data. This enabled reconstruction of the locations of nonzero values of z_{ij}, provided there were no more than 10×10–1=99 of them.

The 100×100 Hermitian Toeplitz-block-Toeplitz matrix with 10×10 blocks of the lowest frequencies of z_{ij} was set up. An image of the element magnitudes is shown in Figure #1. Its singular values were

$$\sigma_{98} = 0.0085, \sigma_{99} = 1.1 \times 10^{-9}, \sigma_{100} = 2.2 \times 10^{-10}$$

which cleary shows that the matrix rank, and hence the number of nonzero values of z_{ij}, is 98.

The 72×72 2-D DFT of the null vector rearranged into a 10×10 array was computed. The logs q_n of its magnitudes were sorted, and there was a clear threshold, confirming the presence of 98 nonzero values of z:

$$q_{98} = 11.47; \quad q_{99} = 17.06; \quad q_{100} = 17.1 \qquad (17)$$

The locations of these 98 smallest values are shown in Figure #2 (all other locations display as zero).

The 2×2 sparsifying filter was then deconvolved from z_{ij} to obtain x_{ij}. Its frequency response is zero along the 2-D frequency axes, so side information x_{i1}=x_{1j}=0 was used to fill in those values. Figure #3 shows the reconstruction of the original image x_{ij}. The complete Matlab code used for this example is listed in the Appendix.

V. Conclusion

We have presented a non-iterative algorithm for reconstructing sparse and sparsifiable solutions to underdetermined linear systems of equations with bandlimited rows. A numerical example demonstrated the algorithm on a sparsifiable image. An important point of this example is that although there are 5184 unknowns and 1296 data points, the only matrix computation was for a 100×100 Toeplitz-block-Toeplitz matrix. The algorithm requires only a second or so. For slightly overdetermined linear systems that do not have bandlimited rows, the algorithm of [1] can be used.

References

[1] A.E. Yagle, "A Non-Iterative Procedure for Computing Sparse and Sparsifiable Solutions to Slightly Underdetermined Linear Systems of Equations," 2008.

[2] A.E. Yagle, "New Atomicity-Exploiting Algorithms for Super-Resolution X-Ray Crystallography," 2008.

[3] A.E. Yagle, "Limited Angle Tomography of Sparse Images from Noisy Data using TLS MUSIC," 2008.

[4] H. Taylor, S. Banks, F. McCoy, "Deconvolution with the L_1 norm," *Geophysics* 44, 39-52, 1979.

[5] F. Santosa and W.W. Symes, "Linear inversion of bandlimited reflection seismograms," *SIAM J. Sci. Stat. Comp.* 7, 1307-1330, 1986.

[6] D.C. Dobson and F. Santosa, "Recovery of blocky images from noisy and blurred data," *SIAM J. Appl. Math* 56, 1181-1198, 1996.

[7] D. Donoho and M. Elad, "Optimally sparse representation in general (nonorthogonal) dictionaries via ℓ^1 minimization," *Proc. Nat. Acad. Sci.* 100(5), 2197-2202, 2003.

[8] K. Tsuda and G. Ratsch, "Image reconstruction by linear programming," *IEEE Trans. Image Proc.* 14, 737-744, 2005.

[9] E. Candes and J. Romberg, *Inverse Problems* June 2007.

[10] J. Haupt and R. Nowak, "Signal reconstruction from noisy random projections," *IEEE Trans. Info. Th.* 52, 4036-4048, 2006.

[11] G. Harikumar and Y. Bresler, "A new algorithm for computing sparse solutions to linear inverse problems," Proc. ICASSP 1996, 1331-1334.

[12] G. Harikumar, C. Couvreur, Y. Bresler, "Fast optimal and suboptimal algorithms for sparse solutions to linear inverse problems," Proc. ICASSP 1998, 1877-1880.

[13] C. Couvreur and Y. Bresler, "On the optimality of the backward greedy algorithm for the subset selection problem," *SIAM J. Matrix Anal. and Appl.* 21, 797-808, 2000.

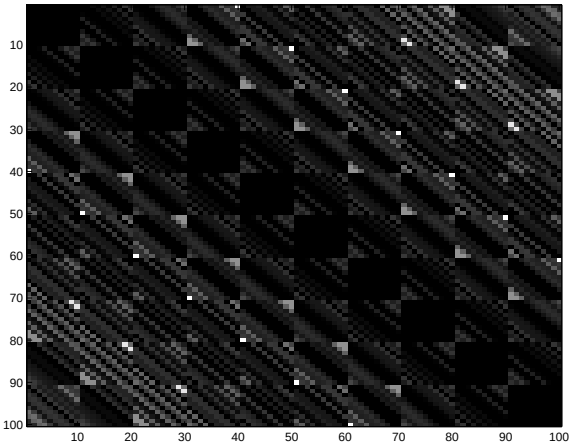
[14] F.R. de Hoog and R.M.M. Mattheij, "Subset selection for matrices," *Linear Algebra and its Applications* 422, 349-359, 2007.

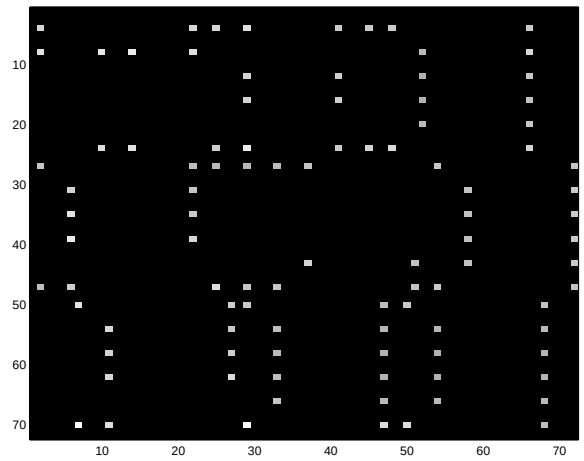
[15] B.K. Natarajan, "Sparse approximate solutions to linear systems," *SIAM J. Computers* 24, 227-234, 1995.

[16] M.A. Figueiredo and R.D. Nowak, "An EM algorithm for wavelet based image restoration," *IEEE Trans. Image Proc.* 12, 906-916, 2003.

[17] I. Daubechies, M. Defrise, C. de Mol, "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint," *Comm. Pure Appl. Math* 57, 1413-1457, 2004.

[18] K.K. Herrity, A.C. Gilbert, J.A. Tropp, "Sparse approximation via iterative thresholding," Proc. ICASSP 2006.





VI. Appendix

A. Sparsification by Circulant Matrix

```
clear;X(1:4,1:20)=ones(4,20);X(1:20,9:12)=ones(20,4);%T
X(1:20,24:27)=ones(20,4);X(9:12,24:43)=ones(4,20);X(1:20,40:43)=ones(20,4);%H
X(1:20,47:50)=ones(20,4);X(1:4,47:64)=ones(4,18);X(9:12,47:64)=ones(4,18);X(17:20,47:64)=ones(4,18);%E
X(24:43,1:4)=ones(20,4);X(24:27,1:20)=ones(4,20);X(32:35,1:20)=ones(4,20);%F
X(24:43,24:27)=ones(20,4);X(24:43,32:35)=ones(20,4);X(40:43,32:49)=ones(4,18);%IL
X(24:43,53:56)=ones(20,4);X(24:27,53:70)=ones(4,18);X(32:35,53:70)=ones(4,18);X(40:43,53:70)=ones(4,18);%E
X(47:66,6:9)=ones(20,4);X(47:50,6:25)=ones(4,20);X(55:58,6:25)=ones(4,20);%F
X(47:66,28:31)=ones(20,4);X(47:50,28:45)=ones(4,18);X(55:58,28:45)=ones(4,18);X(63:66,28:45)=ones(4,18);%E
X(47:66,49:52)=ones(20,4);X(47:50,49:66)=ones(4,18);X(55:58,49:66)=ones(4,18);X(63:66,49:66)=ones(4,18);%E
X=[zeros(3,71);[X zeros(66,1)];zeros(3,71)];X=[zeros(72,1),X];%72X72 block letters with borders of zeros.
H=0.98^([-35:35]^2);H=H'*H;Y=real(ifft2(fft2(H,72,72).*fft2(X)));Y=Y(1:2:72,1:2:72);
FX=fftshift(fft2(Y)./fft2(H(1:2:length(H),1:2:length(H)),36,36));%Deconvolve H from low frequencies of X.
FH=fftshift(fft2([1 -1;-1 1],72,72));FZ=FH(28:46,28:46).*FX(10:28,10:28);%Z=sparsified X so reconstruct Z.
F1=FZ(:);F1=conj(F1(181:361));F1(1)=real(F1(1));TT=toeplitz(F1,F1');%Set up the Hermitian Toeplitz matrix.
II=[];for I=0:9;II=[II [1:10]+19*I];end;T=TT(II,II);[U S V]=svd(T);plot(log(diag(S)))%Clearly has rank=98.
W=-log(abs(fft2(reshape(V(:,100),10,10),72,72)));Q=sort(W(:));W(W¡14)=0;Q(5184-100:5184-96)%threshold.
figure,imagesc(W),colormap(gray)%vs.:imagesc(real(ifft2(fft2(X).*fft2([1 -1;-1 1],72,72)))),colormap(gray)
[I J]=find(W);A=exp(-j*2*pi/72*[I-1 J-1]*[rem([0:170],19)-9;1+floor([0:170]/19)]);Z1=A'\F1(11:181);
for K=1:98;Z(I(K),J(K))=real(Z1(K));end;FW=fft2(Z,72,72)./fft2([1 -1;-1 1],72,72);FW(1,:)=-sum(FW(2:72,:));
FW=FW.';FW(1,2:72)=-sum(FW(2:72,2:72));FW=FW.';FW(1,1)=sum(sum(X));W=real(ifft2(FW));imagesc(W)
```

B. Wavelet Transform by Circulant Matrix

```
W=[1 1 1 1 1 1 1 1;1 1 1 1 -1 -1 -1 -1;1 1 -1 -1 0 0 0 0;0 0 0 0 1 1 -1 -1];
W=[W;1 -1 0 0 0 0 0 0;0 0 1 -1 0 0 0 0;0 0 0 0 1 -1 0 0;0 0 0 0 0 0 1 -1];
H=[1 1 1 1 -1 -1 -1 -1]; %Can also construct W from a circulant matrix:
AH=[-1 zeros(1,56) H(1:7)];AV=[flipud(H');zeros(56,1)];A=toeplitz(AV,AH);
U(8,64)=0;U(:,8:8:64)=eye(8);V(64,8)=0;V(1:8,:)=diag([1 1 1 1 -1 -1 -1 -1]);
V(9:16,:)=eye(8);V(17:2:32,:)=eye(8);V(33:4:64,:)=eye(8);W2=U*A*V*X;%W=W2
```