UCLA
COMPUTATIONAL AND APPLIED MATHEMATICS

Wavelet Algorithms for High-Resolution Image Reconstruction

Raymond H. Chan
Tony F. Chan
Lixin Shen
Zuowei Shen

May 2002

CAM Report 02-21

Department of Mathematics
University of California, Los Angeles
Los Angeles, CA. 90024-1555

Wavelet Algorithms for High-Resolution Image Reconstruction

Raymond H. Chan*, Tony F. Chan†, Lixin Shen‡, Zuowei Shen‡

April 29, 2002

Abstract

High-resolution image reconstruction refers to reconstructing high-resolution images from multiple low-resolution, shifted, degraded samples of a true image. In this paper, we analyze this problem from the wavelet point of view. By expressing the true image as a function in $L_2(\mathbb{R}^2)$, we derive iterative algorithms which recover the function completely in the L_2 sense from the given low-resolution functions. These algorithms decompose the function obtained from the previous iteration into different frequency components in the wavelet transform domain and add them into the new iterate to improve the approximation. We apply wavelet (packet) thresholding methods to denoise the function obtained in the previous step before adding it into the new iterate. Our numerical results show that the reconstructed images from our wavelet algorithms are better than that from the Tikhonov least squares approach. Extension to super-resolution image reconstruction, where some of the low-resolution images are missing, is also considered.

1 Introduction

Many applications in image processing require deconvolving noisy data, for example the deblurring of astronomical images [11]. The main objective in this paper is to develop algorithms for these applications using wavelet approach. We will concentrate on one such application, namely, the high-resolution image reconstruction problem. High-resolution images are often desired in many situations, but made impossible because of hardware limitations. Increasing the resolution by image processing techniques [1, 3, 9, 14, 15, 22, 23, 24] is of great importance. Here we consider creating high-resolution images of a scene from the low-resolution images of the same scene. When we have a full set of low-resolution images, the problem is referred to as *high-resolution image reconstruction*; and when only some of the low-resolution images are available, the problem is called *super-resolution image reconstruction.*

In both cases, the low-resolution images are obtained from sensor arrays which are shifted from each others with subpixel displacements. The reconstruction of the high-resolution image can be modeled as solving a linear system $L\mathbf{f} = \mathbf{g}$, where L is the convolution operator, $\mathbf{g}$ is a vector formed from the low-resolution images, and $\mathbf{f}$ is the desired high-resolution image, see [1].

In this paper, we look at this problem from the wavelet point of view and analyze the process through multiresolution analysis. The true image can be considered as a function f in $L_2(\mathbb{R}^2)$ and

*Department of Mathematics, the Chinese University of Hong Kong, Shatin, NT, P. R. China. Research supported in part by HKRGC Grant CUHK4212/99P and CUHK Grant DAG 2060183.

†Department of Mathematics, University of California, Los Angeles, Los Angeles, CA 90024-1555, USA. Research supported in part by the ONR under contract N00014-96-1-0277 and by the NSF under grant DMS-9973341

‡Department of Mathematics, National University of Singapore, 2 Science Drive 2, Singapore 117543. Research supported in part by the Centre of Wavelets, Approximation and Information Processing (CWAIP) funded by the National Science and Technology Board, the Ministry of Education under Grant RP960 601/A.

the low-resolution images can be thought of as the low-frequency samples of f obtained by passing f through some lowpass filters. Thus the problem can be posed as reconstructing a function from the given multiple low-frequency samples of f. To recover f, we deconvolve iteratively the high-frequency components of f which are hidden in the low-frequency samples. Our iterative process decomposes the function obtained in the previous iteration into different frequency components in the wavelet transform domain and then adds them to the new iterate to improve the approximation. In this setting, it is easy to apply wavelet methods to denoise the function obtained in the previous step before adding it into the new iterate.

The high-resolution image reconstruction problem is closely related to the deconvolution problem. In the recent works of [12] and [13], an analysis of minimizing the maximum risk over all the signals in a set of signals is given. Then, it was applied to estimate the risk of the wavelet thresholding method used on the deconvoluted signals. Their wavelet thresholding algorithm is proven to be close to the optimal risk bound when a mirror wavelet basis is used. The main difficulty in denoising deconvoluted signals is that when the convolution lowpass filter has zeros at high frequency, the noise variance in the solution has a hyperbolic growth [12]. To overcome this difficulty, a mirror wavelet basis is constructed to define a sparse representation of all the signals in the set of given signals and to nearly diagonalize the covariance operator of the noise in the deconvoluted solution in order to reach the optimal risk bound.

The approach here is different. The highpass filters are added to perturb the zeros of the convolution kernel to prevent the noise variance from blowing up. Our wavelet (packet) thresholding method, which is a wavelet denoising method, is built into each iterative step so as to remove the noise from the original data. It also keeps the features of the original signal while denoising. In this sense, our method is more related to the Tikhonov least squares method where a regularization operator is used to perturb the zeros of the convolution kernel and a penalty parameter is used to damp the high-frequency components for denoising. Since the least squares method penalizes the high-frequency components of the original signal at the same rate as that of the noise, it smoothens the original signal. In contrast, our thresholding method penalizes the high-frequency components of the signal in a rate significantly lower than that of the noise, and hence it will not smoothen the original signal in general. Moreover, there is no need to estimate the regularization parameter in our method. Our numerical tests show that the reconstructed images are of better quality. Also our algorithms can easily be extended to the super-resolution case.

The outline of the paper is as follows. In §2, we give a mathematical model of the high-resolution image reconstruction problem. In §3, we derive our algorithms. Extensions to the super-resolution case are also discussed there. Numerical examples are given in §4 to illustrate the effectiveness of the algorithms. After the concluding remarks, we provide in the Appendix an analysis of our algorithms via the multiresolution analysis.

2 The Mathematical Model

Here we give a brief introduction to the mathematical model of the high-resolution image reconstruction problem. Details can be found in [1]. Suppose the image of a given scene can be obtained from sensors with $N_1 \times N_2$ pixels. Let the actual length and width of each pixel be T_1 and T_2 respectively. We will call these sensors *low-resolution sensors*. The scene we are interested in, i.e. the *region of interest*, can be described as:

$$\mathbf{S} = \left\{(x_1, x_2) \in \mathbb{R}^2 \mid 0 \le x_1 \le T_1 N_1, 0 \le x_2 \le T_2 N_2\right\}.$$

Our aim is to construct a higher resolution image of the same scene by using an array of $K_1 \times K_2$ low-resolution sensors. More precisely, we want to create an image of $\mathbf{S}$ with $M_1 \times M_2$ pixels, where

$M_1 = K_1 N_1$ and $M_2 = K_2 N_2$. Thus the length and width of each of these *high-resolution pixels* will be T_1/K_1 and T_2/K_2 respectively. To maintain the aspect ratio of the reconstructed image, we consider only $K_1 = K_2 = K$.

Let $f(x_1, x_2)$ be the intensity of the scene at any point (x_1, x_2) in $\mathbf{S}$. By reconstructing the high-resolution image, we mean to find or approximate the values

$$\frac{K^2}{T_1 T_2} \int_{iT_1/K}^{(i+1)T_1/K} \int_{jT_2/K}^{(j+1)T_2/K} f(x_1, x_2) dx_1 dx_2, \quad 0 \le i < M_1, 0 \le j < M_2,$$

which is the average intensity of all the points inside the (i, j)th high-resolution pixel:

$$\left[i\frac{T_1}{K}, (i+1)\frac{T_1}{K} \right] \times \left[j\frac{T_2}{K}, (j+1)\frac{T_2}{K} \right], \quad 0 \le i < M_1, 0 \le j < M_2. \tag{1}$$

In order to have enough information to resolve the high-resolution image, there are subpixel displacements between the sensors in the sensor arrays. Ideally, the sensors should be shifted from each other by a value proportional to the length and the width of the high-resolution pixels. More precisely, for sensor (k_1, k_2), $0 \le k_1, k_2 < K$, its horizontal and vertical displacements $d^x_{k_1 k_2}$ and $d^y_{k_1 k_2}$ with respect to the point $(0, 0)$ are given by

$$d^x_{k_1 k_2} = \left(k_1 + \frac{1-K}{2} \right) \frac{T_1}{K} \quad \text{and} \quad d^y_{k_1 k_2} = \left(k_2 + \frac{1-K}{2} \right) \frac{T_2}{K}.$$

For this low-resolution sensor, the average intensity registered at its (n_1, n_2)th pixel is modeled by:

$$g_{k_1 k_2}[n_1, n_2] = \frac{1}{T_1 T_2} \int_{T_1 n_1 + d^x_{k_1 k_2}}^{T_1(n_1+1)+d^x_{k_1 k_2}} \int_{T_2 n_2 + d^y_{k_1 k_2}}^{T_2(n_2+1)+d^y_{k_1 k_2}} f(x_1, x_2) dx_1 dx_2 + \eta_{k_1 k_2}[n_1, n_2]. \tag{2}$$

Here $0 \le n_1 < N_1$ and $0 \le n_2 < N_2$ and $\eta_{k_1 k_2}[n_1, n_2]$ is the noise, see [1]. We remark that the integration is over an area the same size of a low-resolution pixel.

Notice that using the mid-point quadrature rule and neglecting the noise $\eta_{k_1 k_2}[n_1, n_2]$ for the moment,

$$\begin{aligned} g_{k_1 k_2}[n_1, n_2] &\approx f\left(T_1(n_1 + \frac{1}{2}) + d^x_{k_1 k_2}, T_2(n_2 + \frac{1}{2}) + d^y_{k_1 k_2} \right) \\ &= f\left(\frac{T_1}{K}(Kn_1 + k_1 + \frac{1}{2}), \frac{T_2}{K}(Kn_2 + k_2 + \frac{1}{2}) \right). \end{aligned}$$

Thus if we intersperse all the low-resolution images $g_{k_1 k_2}$ to form an $M_1 \times M_2$ image g by assigning

$$g[Kn_1 + k_1, Kn_2 + k_2] = g_{k_1 k_2}[n_1, n_2], \tag{3}$$

then

$$g[i, j] \approx f\left(\frac{T_1}{K}(i + \frac{1}{2}), \frac{T_2}{K}(j + \frac{1}{2}) \right), \quad 0 \le i < M_1, 0 \le j < M_2$$

which is the value of f at the mid-point of the (i, j)th high-resolution pixel in (1). Thus g is an approximation of f. Figure 1 shows how to form a 4×4 image g from four 2×2 sensor arrays $\{g_{k_1 k_2}\}_{k_1, k_2 = 0}^{1}$ where all $g_{k_1 k_2}$ have 2×2 pixels. The image g, called the *observed high-resolution image*, is already a better image (i.e. better approximation to f) than any one of the low-resolution samples g_{k_1, k_2} themselves, see Figures 4 (a)–(c) in §4.

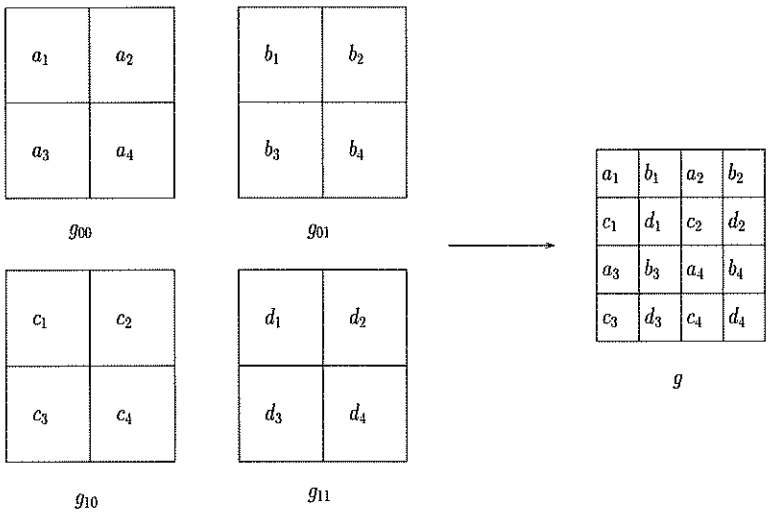

Figure 1: Construction of the observed high-resolution image.

To obtain an even better image than the observed high-resolution image g, one will have to solve (2) for f. According to [1], we solve it by first discretizing it using the rectangular quadrature rule. Or equivalently, we assume that for each (i,j)th high-resolution pixel given in (1), the intensity f is constant and is equal to $f[i,j]$ for every point in that pixel. Then carrying out the integration in (2), and using the re-ordering (3), we obtain a system of linear equations relating the unknown values $f[i,j]$ to the given low-resolution pixel values $g[i,j]$. This linear system, however, is not square. This is because the evaluation of $g_{k_1k_2}[n_1,n_2]$ in (2) involves points outside the region of interest $\mathbf{S}$. For example, $g_{0,0}[0,0]$ requires the values $f(x_1,x_2)$ where $x_1<0$ and $x_2<0$, i.e. it involves $f[-1,-1]$. Thus we have more unknowns than given values, and the system is underdetermined.

To compensate for this, one imposes boundary conditions on f for x_i outside the domain. A standard way is to assume that f is periodic outside:

$$f(x+iT_1N_1, y+jT_2N_2) = f(x,y), \qquad i,j \in \mathbb{Z},$$

see for instance [8, §5.1.3]. Other boundary conditions, such as the *symmetric* (also called Neumann or reflective) boundary condition and the *zero* boundary condition, can also be imposed, see [1, 18]. We emphasize that these boundary conditions will introduce boundary artifacts in the recovered images, see for examples Figures 4 (d)–(f) in §4. For simplicity, we will only develop our algorithms for periodic boundary conditions here. For other boundary conditions, similar algorithms can be derived straightforwardly.

Using the periodic boundary condition and ordering the discretized values of f and g in a row-by-row fashion, we obtain an $M_1M_2 \times M_1M_2$ linear system of the form:

$$L\mathbf{f} = \mathbf{g}. \tag{4}$$

The blurring matrix L can be written as

$$L = L^x \otimes L^y \tag{5}$$

where $\otimes$ is the Kronecker tensor product and L^x is the $M_1 \times M_1$ circulant matrix with the first row given by

$$\frac{1}{K}\left[1,\ldots,1,\frac{1}{2},0,\ldots,0,\frac{1}{2},1,\ldots,1\right].$$

Here the first $K/2$ entries are equal to 1. We note that the last $K/2$ nonzero entries are there because of our periodic assumption on f. The $M_2 \times M_2$ blurring matrix L^y is defined similarly.

We note that the matrix L is a block-circulant-circulant-block (BCCB) matrix. Thus (4) can be solved by using three 2-dimensional Fast Fourier Transforms (FFTs) in $O(M_1M_2\log(M_1M_2))$ operations, see for instance [8, §5.2.2]. As examples, the matrices L^x for the cases of 2×2 and 4×4 sensor arrays are given respectively by:

$$L_2 \equiv \frac{1}{4}\begin{bmatrix} 2 & 1 & & & 1 \\ 1 & 2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & 2 & 1 \\ 1 & & & 1 & 2 \end{bmatrix} \quad \text{and} \quad L_4 \equiv \frac{1}{8}\begin{bmatrix} 2 & 2 & 1 & & & 1 & 2 \\ 2 & 2 & 2 & 1 & & & 1 \\ 1 & 2 & 2 & 2 & 1 & & \\ & \ddots & \ddots & \ddots & \ddots & \ddots & \\ & & 1 & 2 & 2 & 2 & 1 \\ 1 & & & 1 & 2 & 2 & 2 \\ 2 & 1 & & & 1 & 2 & 2 \end{bmatrix}. \tag{6}$$

Because (2) is an averaging process, the system in (4) is ill-conditioned and susceptible to noise. To remedy this, one can use the Tikhonov regularization which solves the system

$$(L^*L + \beta R)\mathbf{f} = L^*\mathbf{g}. \tag{7}$$

Here R is a regularization operator (usually chosen to be the identity operator or some differential operators) and $\beta > 0$ is the regularization parameter, see [8, §5.3]. If the boundary condition of R is chosen to be periodic, then (7) is still a BCCB system and hence can be solved by 3 FFTs in $O(M_1M_2\log(M_1M_2))$ operations. If the symmetric boundary condition is used, then (7) is a block Toeplitz-plus-Hankel system with Toeplitz-plus-Hankel blocks. It can be solved by using three 2-dimensional fast cosine transforms (FCTs) in $O(M_1M_2\log(M_1M_2))$ operations, see [18].

We note that (7) is derived from the least squares approach of solving (4). In the next section, we will derive algorithms for finding $\mathbf{f}$ by using the wavelet approach. They will improve the quality of the images when compared with (7).

3 Reconstruction

In this section, we analyze the model given in §2 using the wavelet approach. Since (2) is an averaging process, the matrices in (6) can be considered as a lowpass filtering acting on the image $\mathbf{f}$ with a tensor product refinement mask, say a. Let ϕ be the tensor product bivariate refinable function with such a refinement mask. Here, we recall that a function ϕ in $L_2(\mathbb{R}^2)$ is *refinable* if it satisfies

$$\phi = 4\sum_{\alpha\in\mathbb{Z}^2} a(\alpha)\phi(2\cdot -\alpha).$$

The sequence a is called a *refinement mask*, or lowpass filter. The *symbol* of the sequence a is defined as

$$\widehat{a}(\omega) := \sum_{\alpha\in\mathbb{Z}^2} a(\alpha)e^{-i\alpha\omega}.$$

The function ϕ is *stable* if its shifts (integer translates) form a *Riesz system*, i.e. there exist constants $0 < c \le C < \infty$, such that for any sequence $q \in \ell_2(\mathbb{Z}^2)$,

$$c\|q\|_2 \le \left\| \sum_{\alpha \in \mathbb{Z}^2} q(\alpha)\phi(\cdot - \alpha) \right\|_2 \le C\|q\|_2. \tag{8}$$

Stable functions ϕ and ϕ^d are called a *dual pair* when they satisfy

$$\langle \phi, \phi^d(\cdot - \alpha) \rangle = \begin{cases} 1, & \alpha = 0; \\ 0, & \alpha \in \mathbb{Z}^2 \setminus \{(0,0)\}. \end{cases}$$

We will denote the refinement mask of ϕ^d by a^d.

For a given compactly supported refinable stable function $\phi \in L_2(\mathbb{R}^2)$, define $S(\phi) \subset L_2(\mathbb{R}^2)$ to be the smallest closed shift invariant subspace generated by ϕ and define

$$S^k(\phi) := \{u(2^k \cdot) : u \in S(\phi)\}, \quad k \in \mathbb{Z}.$$

Then the sequence $S^k(\phi)$, $k \in \mathbb{Z}$, forms a multiresolution of $L_2(\mathbb{R}^2)$. Here we recall that a sequence $S^k(\phi)$ forms a *multiresolution* when the following conditions are satisfied: (i) $S^k(\phi) \subset S^{k+1}(\phi)$; (ii) $\cup_{k\in\mathbb{Z}} S^k(\phi) = L_2(\mathbb{R}^2)$ and $\cap_{k\in\mathbb{Z}} S^k(\phi) = \{0\}$; (iii) ϕ and its shifts form a Riesz basis of $S(\phi)$, see [5]. The sequence $S^k(\phi^d)$, $k \in \mathbb{Z}$, also forms a multiresolution of $L_2(\mathbb{R}^2)$.

The tensor product of univariate refinable functions and wavelets used in this paper will be derived from the following examples.

Example 1 ([5, p.277]). *For 2×2 sensor arrays, using the rectangular rule for (2), we get L_2 in (6). Correspondingly, the refinement mask m is the piecewise linear spline, i.e.*

$$m(-1) = \frac{1}{4},\ m(0) = \frac{1}{2},\ m(1) = \frac{1}{4},$$

and $m(\alpha) = 0$ for all other α. The nonzero terms of the dual mask of m used in this paper are:

$$m^d(-2) = -\frac{1}{8},\ m^d(-2) = \frac{1}{4},\ m^d(0) = \frac{3}{4},\ m^d(1) = \frac{1}{4},\ m^d(2) = -\frac{1}{8}.$$

In general, the tensor product bivariate filters for the dilation $2I$, where I is the identity matrix, are generated as follows. Let ϕ and ϕ^d be a dual pair of the univariate refinable functions with refinement masks m and m^d respectively. Then, the biorthonormal wavelets ψ and ψ^d are defined by

$$\psi := 2\sum_{\alpha\in\mathbb{Z}} r(\alpha)\phi(2\cdot - \alpha), \quad \text{and} \quad \psi^d := 2\sum_{\alpha\in\mathbb{Z}} r^d(\alpha)\phi^d(2\cdot - \alpha),$$

where

$$r(\alpha) := (-1)^\alpha m^d(1-\alpha), \quad \text{and} \quad r^d(\alpha) := (-1)^\alpha m(1-\alpha)$$

are the wavelet masks, see for example [5] for details. The tensor product dual pair of the refinement symbols are given by $\widehat{a}(\omega) = \widehat{m}(\omega_1)\widehat{m}(\omega_2)$, $\widehat{a}^d(\omega) = \widehat{m}^d(\omega_1)\widehat{m}^d(\omega_2)$, and the corresponding wavelet symbols are $\widehat{b}_{(0,1)}(\omega) = \widehat{m}(\omega_1)\widehat{r}(\omega_2)$, $\widehat{b}^d_{(0,1)}(\omega) = \widehat{m}^d(\omega_1)\widehat{r}^d(\omega_2)$, $\widehat{b}_{(1,0)}(\omega) = \widehat{r}(\omega_1)\widehat{m}(\omega_2)$, $\widehat{b}^d_{(1,0)}(\omega) = \widehat{r}^d(\omega_1)\widehat{m}^d(\omega_2)$, $\widehat{b}_{(1,1)}(\omega) = \widehat{r}(\omega_1)\widehat{r}(\omega_2)$, $\widehat{b}^d_{(1,1)}(\omega) = \widehat{r}^d(\omega_1)\widehat{r}^d(\omega_2)$, where $\omega = (\omega_1, \omega_2)$.

Although we only give here the details of refinable functions and their corresponding wavelets with dilation $2I$, the whole theory can be carried over to the general isotropic integer dilation matrices. The details can be found in [10] and the references therein. In the next example, we give the refinable and wavelet masks with dilation $4I$ that are used to generate the matrices for 4×4 sensor arrays.

Example 2. *For 4×4 sensor arrays, using the rectangular rule for (2), we get L_4 in (6). The corresponding mask is*

$$m(\alpha) = \frac{1}{8}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{8}, \quad \alpha = -2, \ldots, 2,$$

with $m(\alpha) = 0$ for all other α. It is the mask of a stable refinable function ϕ with dilation 4 (see e.g. [16] for a proof). The nonzero terms of a dual mask of m is

$$m^d(\alpha) = -\frac{1}{16}, \frac{1}{8}, \frac{5}{16}, \frac{1}{4}, \frac{5}{16}, \frac{1}{8}, -\frac{1}{16}, \quad \alpha = -3, \ldots, 3.$$

The nonzero terms of the corresponding wavelet masks (see [20]) are

$$r_1(\alpha) = -\frac{1}{8}, -\frac{1}{4}, 0, \frac{1}{4}, \frac{1}{8}, \quad \alpha = -2, \ldots, 2,$$

$$r_2(\alpha) = -\frac{1}{16}, -\frac{1}{8}, \frac{5}{16}, -\frac{1}{4}, \frac{5}{16}, -\frac{1}{8}, -\frac{1}{16}, \quad \alpha = -2, \ldots, 4,$$

$$r_3(\alpha) = \frac{1}{16}, \frac{1}{8}, -\frac{7}{16}, 0, \frac{7}{16}, -\frac{1}{8}, -\frac{1}{16} \quad \alpha = -2, \ldots, 4.$$

The dual wavelet masks are

$$r_1^d(\alpha) = (-1)^{1-\alpha} r_3(1-\alpha), \; r_2^d(\alpha) = (-1)^{1-\alpha} m(1-\alpha), \; r_3^d(\alpha) = (-1)^{1-\alpha} r_1(1-\alpha),$$

for appropriate α.

In the next two subsections, we will use the wavelet approach to design algorithms for recovering the high-resolution image from the low-resolution images of a true image. In §3.1, we first consider the true image as a representation of a function in certain subspace of $L_2(\mathbb{R}^2)$ and using wavelet means we recover this function from the given set of low-resolution images. In §3.2, we translate the wavelet algorithms into matrix terminologies.

3.1 Function Reconstruction

Since $S^k(\phi^d)$, $k \in \mathbb{Z}$, forms a multiresolution of $L_2(\mathbb{R}^2)$, we can assume without loss of generality that the pixel values of the original image are the coefficients of a function f in $S^k(\phi^d)$ for some k. The pixel values of the low-resolution images can be considered as the coefficients of a function g in $S^{k-1}(\phi^d)$ and its $1/2^k$ translates, i.e. g is represented by $\phi^d(2^{k-1}(\cdot - \alpha/2^k))$ with $\alpha \in \mathbb{Z}^2$. The low-resolution images keep most of the low-frequency information of f and the high-frequency information in f is folded by the lowpass filter a. Hence, to recover f, the high-frequency information of f hidden in the low-resolution images will be unfolded and combined with the low-frequency information to restore f. We will unfold the high-frequency content iteratively using the wavelet decomposition and reconstruction algorithms.

For 2×2 sensor arrays, the multiresolution analysis $S^k(\phi^d)$ used is from a refinable function with dilation matrix $2I$. In general, $K \times K$ sensor arrays can be analyzed by the multiresolution analysis generated by a refinable function with dilation matrix $K \cdot I$. For simplicity, we give the analysis for the case that the dilation matrix is $2I$ and $k = 1$. A similar analysis can be carried out for more general cases.

Let $f \in S^1(\phi^d)$. Then,

$$f = \sum_{\alpha \in \mathbb{Z}^2} \langle f, 2\phi(2 \cdot - \alpha) \rangle 2\phi^d(2 \cdot - \alpha) := 2 \sum_{\alpha \in \mathbb{Z}^2} v(\alpha) \phi^d(2 \cdot - \alpha). \tag{9}$$

The numbers $v(\alpha)$, $\alpha \in \mathbb{Z}^2$, are the pixel values of the high-resolution image we are seeking, and they form the discrete representation of f under the basis $2\phi^d(2\cdot - \alpha)$, $\alpha \in \mathbb{Z}^2$. The given data set $a * v(\alpha)$ is the observed high-resolution image. By using the refinability of ϕ^d, one finds that $a * v$ is the coefficient sequence of the function g represented by $\phi^d(\cdot - \alpha/2)$, $\alpha \in \mathbb{Z}^2$, in $S^1(\phi^d)$. We call this g the *observed function* and it is given by

$$g := \sum_{\alpha \in \mathbb{Z}^2} (a * v)(\alpha)\phi^d(\cdot - \alpha/2). \tag{10}$$

The observed function can be obtained precisely once $a * v$ is given.

When only $a * v$ is given, to recover f, one first finds v from $a * v$; then, derives f using the basis $2\phi^d(2\cdot - \alpha)$, $\alpha \in \mathbb{Z}^2$, as in (9). Here we provide an iterative algorithm to recover v. At step $(n+1)$ of the algorithm, it improves the high-frequency components of f by updating the high-frequency components of the previous step. The algorithm is presented in the Fourier domain where the problem becomes: for a given $\widehat{a * v} = \widehat{a}\widehat{v}$, one needs to find $\widehat{v}$ in order to restore f.

Our algorithm will make use of the following fact from the biorthogonal wavelet theory: for a given tensor product bivariate refinement mask a corresponding to a stable refinable function ϕ, one can find a tensor product dual mask a^d and the corresponding wavelet masks b_ν and b^d_ν, $\nu \in \mathbb{Z}^2_2$, such that the symbols of the refinement masks and wavelet masks satisfy the following equation:

$$\overline{\widehat{a}^d}\widehat{a} + \sum_{\nu \in \mathbb{Z}^2_2 \setminus \{(0,0)\}} \overline{\widehat{b}^d_\nu}\widehat{b}_\nu = 1. \tag{11}$$

We note that when the refinable function ϕ (of the convolution kernel a) and its shifts form an orthonormal system (e.g. in the Haar case), then (11) holds if we choose $\widehat{a}^d = \widehat{a}$ and $\widehat{b}^d_\nu = \widehat{b}_\nu$. This leads to orthonormal wavelets. When ϕ and its shifts form only a Riesz system, one has to use biorthogonal wavelets.

By (11), we have $\widehat{v} = \overline{\widehat{a}^d}\widehat{a * v} + (\sum_\nu \overline{\widehat{b}^d_\nu}\widehat{b}_\nu)\widehat{v}$. Hence we have the following algorithm.

Algorithm 1.

(i) Choose $\widehat{v}_0 \in L_2[-\pi, \pi]^2$;

(ii) Iterate until convergence:

$$\widehat{v}_{n+1} = \overline{\widehat{a}^d}\widehat{a * v} + \left(\sum_{\nu \in \mathbb{Z}^2_2 \setminus \{(0,0)\}} \overline{\widehat{b}^d_\nu}\widehat{b}_\nu \right) \widehat{v}_n. \tag{12}$$

We remark that the first term $\overline{\widehat{a}^d}\widehat{a * v} = \overline{\widehat{a}^d}\widehat{a}\widehat{v}$ in the right hand side of (12) represents the approximation of the low-frequency components of f whereas the second term improves the high frequency approximation.

Given $\widehat{v}_n$, f_n is defined via its Fourier transform as:

$$\widehat{f}_n(\cdot) := \widehat{v}_n(\cdot/2)\widehat{\phi}^d(\cdot/2) \in S^1(\phi^d). \tag{13}$$

We now show that the functions f_n converge to the function f in (9).

Proposition 1. *Let ϕ and ϕ^d be a pair of dual refinable functions with refinement masks a and a^d and let b_ν and b^d_ν, $\nu \in \mathbb{Z}^2_2 \setminus \{(0,0)\}$, be the wavelet masks of the corresponding biorthogonal wavelets. Suppose that $0 \le \overline{\widehat{a}^d}\widehat{a} \le 1$ and its zero set has measure zero. Then, the sequence $\widehat{v}_n$ defined in (12) converges to $\widehat{v}$ in the L_2-norm for any arbitrary $\widehat{v}_0 \in L_2[-\pi, \pi]^2$. In particular, f_n in (13) converges to f in (9) in the L_2-norm.*

Proof. For an arbitrary $\widehat{v}_0 \in L_2[-\pi,\pi]^2$, applying (12), we have

$$\widehat{v} - \widehat{v}_n = \left(\sum_{\nu \in \mathbb{Z}_2^2 \setminus \{(0,0)\}} \overline{\widehat{b}_\nu^d} \widehat{b}_\nu \right)^n (\widehat{v} - \widehat{v}_0).$$

It follows from $0 \le \overline{\widehat{a}^d}\widehat{a} \le 1$ and (11) that

$$\left| \sum_{\nu \in \mathbb{Z}_2^2 \setminus \{(0,0)\}} \overline{\widehat{b}_\nu^d} \widehat{b}_\nu \right| \le 1.$$

Since $\overline{\widehat{a}^d}\widehat{a} \ge 0$ and its zero set has measure zero, the inequality

$$\left| \sum_{\nu \in \mathbb{Z}_2^2 \setminus \{(0,0)\}} \overline{\widehat{b}_\nu^d} \widehat{b}_\nu \right| < 1$$

holds almost everywhere. Hence,

$$\left(\sum_{\nu \in \mathbb{Z}_2^2 \setminus \{(0,0)\}} \overline{\widehat{b}_\nu^d} \widehat{b}_\nu \right)^n (\widehat{v} - \widehat{v}_0) \to 0, \quad \text{as } n \to \infty \text{ a.e..}$$

By the Dominated Convergence Theorem (see e.g. [19]), $\widehat{v}_n$ converges to $\widehat{v}$ in the L_2-norm.

Since $\phi^d(\cdot - \alpha)$ is a Riesz basis of $S^0(\phi^d)$, from (8), $2\phi^d(2\cdot - \alpha)$ is a Riesz basis of $S^1(\phi^d)$. Hence by (8) again,

$$\begin{aligned} \|f_n - f\|_2 &= \| \sum_{\alpha \in \mathbb{Z}^2} (v_n(\alpha) - v(\alpha)) 2\phi(2\cdot - \alpha)\|_2 \\ &\le C \| \sum_{\alpha \in \mathbb{Z}^2} (v_n(\alpha) - v(\alpha))\|_2 = \frac{C}{2\pi} \|\widehat{v}_n - \widehat{v}\|_2 \longrightarrow 0. \end{aligned}$$

□

Remark 1. *The symbols of the refinement masks and the corresponding dual masks used in this paper are tensor products of univariate ones that satisfy the assumptions of this proposition.*

Remark 2. *The above convergence result is also applicable to the super-resolution case. Assume for simplicity that $a*v$ is downsampled to four sub-samples $a*v(\nu - 2\alpha)$, $\nu \in \mathbb{Z}_2^2$. For the super-resolution case, one only has some of the sub-samples, $a*v(\nu - 2\alpha)$, $\nu \in A \subset \mathbb{Z}_2^2$. In this case, one first applies an interpolatory scheme (e.g. [10]) to obtain the full set of the sample w approximately. Let the ℓ_2 solution of the equation $a * z = w$ be u. Then, $a * u(\nu - 2\alpha) = a * v(\nu - 2\alpha)$ for $\nu \in A$. Applying Algorithm 1 to $a * u$, the above proposition asserts that it converges to u.*

When there are noise in the given data $a * v$, one may subtract some high-frequency components from $\widehat{v}_n$ at each iteration to reduce the noise, since noise is in the high-frequency components. Then we get the following modified algorithm:

Algorithm 2.

(i) Choose $\widehat{v}_0 \in L_2[-\pi,\pi]^2$;

(ii) Iterate until convergence:

$$\widehat{v}_{n+1} = \overline{\widehat{a}^d}\widehat{a * v} + (1-\beta)\left(\sum_{\nu\in\mathbb{Z}_2^2\setminus\{(0,0)\}} \overline{\widehat{b_\nu^d}}\widehat{b}_\nu\right)\widehat{v}_n, \quad 0<\beta<1. \tag{14}$$

In this denoising procedure, the high-frequency components are penalized uniformly by the factor $(1-\beta)$. This smoothens the original signals while denoising.

To remedy this, we now introduce a wavelet thresholding denoising method. It is based on the observation that $\widehat{b_\nu v_n} = \widehat{b_\nu * v_n}$, $\nu \in \mathbb{Z}_2^2\setminus\{(0,0)\}$, is the exact wavelet coefficients of the wavelet decomposition (without downsampling) of the function f_n, the nth approximation of f. A further decomposition of $b_\nu * v_n$, by using the lowpass filter a and the highpass filters b_ν several times, will give the wavelet coefficients of the decomposition of f_n by the translation invariant wavelet packets defined by the biorthogonal filters a, a^d and b_ν and b_ν^d (see e.g. [17] and [25]). More precisely, by using (11) we have

$$\widehat{b}_\nu\widehat{v}_n = \left(\overline{\widehat{a}^d}\right)^J (\widehat{a})^J \widehat{b}_\nu\widehat{v}_n + \sum_{j=0}^{J-1}\left(\overline{\widehat{a}^d}\right)^j \sum_{\gamma\in\mathbb{Z}_2^2\setminus\{(0,0)\}} \overline{\widehat{b_\gamma^d}}\widehat{b}_\gamma\,(\widehat{a})^j\,\widehat{b}_\nu\widehat{v}_n,$$

where J is the number of levels used to decompose $\widehat{b}_\nu\widehat{v}_n$. For each j and $\gamma \in \mathbb{Z}_2^2\setminus\{(0,0)\}$, $\widehat{b}_\gamma\,(\widehat{a})^j\,\widehat{b}_\nu\widehat{v}_n$ is the coefficients of the wavelet packet (see e.g. [25]) down to the jth level. A wavelet thresholding denoising procedure is then applied to $\widehat{b}_\gamma\,(\widehat{a})^j\,\widehat{b}_\nu\widehat{v}_n$, the coefficients of the wavelet packet decomposition of f_n, before $b_\nu * v_n$ is reconstructed back by the dual masks. This denoises the function f_n. Our method keeps the features of the original signal. Moreover, since we do not downsample (by a factor of 2) in the decomposition procedure, we are essentially using a translation invariant wavelet packet system [17], which is a highly redundant system. As was pointed out in [4] and [17], a redundant system is desirable in denoising, since it reduces the Gibbs oscillations.

Another potential problem with Algorithm 2 is that at each iteration, $\widehat{v}_{n+1}$ inherits the noise from the observed data $\widehat{a * v}$ present in the first term on the right hand side of (14). If the algorithm converges at the n_0-th step, then $\widehat{v}_{n_0}$ still carries the noise from $\widehat{a * v}$. One can eliminate part of these noise by passing the final iterate $\widehat{v}_{n_0}$ through the wavelet thresholding scheme we mentioned above (see Step (iii) below). We summarize our thresholding method in the following algorithm.

Algorithm 3.

(i) Choose $\widehat{v}_0 \in L_2[-\pi,\pi]^2$;

(ii) Iterate until convergence:

$$\widehat{v}_{n+1} = \overline{\widehat{a}^d}\widehat{a * v} + \sum_{\nu\in\mathbb{Z}_2^2\setminus\{(0,0)\}} \overline{\widehat{b_\nu^d}}\,\mathcal{T}\left(\widehat{b}_\nu\widehat{v}_n\right),$$

where

$$\mathcal{T}(\widehat{b}_\nu\widehat{v}_n) = \left(\overline{\widehat{a}^d}\right)^J (\widehat{a})^J \widehat{b}_\nu\widehat{v}_n + \sum_{j=0}^{J-1}\left(\overline{\widehat{a}^d}\right)^j \sum_{\gamma\in\mathbb{Z}_2^2\setminus\{(0,0)\}} \overline{\widehat{b_\gamma^d}}\,\mathcal{D}\left(\widehat{b}_\gamma\,(\widehat{a})^j\,\widehat{b}_\nu\widehat{v}_n\right),$$

and $\mathcal{D}$ *is a thresholding operator (see for instant (19) below).*

(iii) Let $\widehat{v}_{n_0}$ be the final iterate from Step (ii). The final solution of our Algorithm is:

$$\widehat{v}_c = \left(\widehat{\overline{a^d}}\right)^J (\widehat{a})^J \widehat{v}_{n_0} + \sum_{j=0}^{J-1} \left(\widehat{\overline{a^d}}\right)^j \sum_{\gamma \in \mathbb{Z}_2^2 \setminus \{(0,0)\}} \widehat{\overline{b_\gamma^d}} \mathcal{D}\left(\widehat{b}_\gamma (\widehat{a})^j \widehat{v}_{n_0}\right),$$

where $\mathcal{D}$ is the same thresholding operator used in Step (ii).

In both the regularization method (Algorithm 2) and the thresholding method (Algorithm 3), the nth approximation f_n is denoised before it is added to the iterate to improve the approximation. The major difference between Algorithms 2 and 3 is that Step (ii) in Algorithm 2 is replaced by Step (ii) of Algorithm 3 where the thresholding denoising procedure is built in.

3.2 Image Reconstruction

Let us now translate the results above from wavelets notations into matrix terminologies. Denote by L, L^d, H_ν^d and H_ν the matrices generated by the symbols of the refinement and wavelet masks $\widehat{a}$, $\widehat{\overline{a^d}}$, $\widehat{b_\nu}$ and $\widehat{\overline{b_\nu^d}}$ respectively. For the periodic boundary conditions, the matrix L is already given in (5) and the matrices L^d, H_ν^d and H_ν will be given in detail in §4. Using these matrices, (11) can be written as:

$$L^d L + \sum_{\nu \in \mathbb{Z}_2^2 \setminus \{(0,0)\}} H_\nu^d H_\nu = I. \tag{15}$$

Also (12) can be written as:

$$\mathbf{f}_{n+1} = L^d \mathbf{g} + \left(\sum_{\nu \in \mathbb{Z}_2^2 \setminus \{(0,0)\}} H_\nu^d H_\nu \right) \mathbf{f}_n \tag{16}$$

where $\mathbf{g}$ ($\sim a * v$) is the observed high-resolution image given in (4) and $\mathbf{f}_n$ are the approximations of $\mathbf{f}$ at the nth iteration.

Rewriting (16) as

$$\mathbf{f}_{n+1} - \left(\sum_{\nu \in \mathbb{Z}_2^2 \setminus \{(0,0)\}} H_\nu^d H_\nu \right) \mathbf{f}_n = L^d \mathbf{g},$$

one sees that it is a stationary iteration for solving the matrix equation

$$\left[I - \left(\sum_{\nu \in \mathbb{Z}_2^2 \setminus \{(0,0)\}} H_\nu^d H_\nu \right) \right] \mathbf{f} = L^d \mathbf{g}.$$

Therefore by (15), we get the matrix form of Algorithm 1.

Algorithm 1 in Matrix Form:

$$L^d L \mathbf{f} = L^d \mathbf{g}. \tag{17}$$

We note that there is no need to iterate on (16) to get $\mathbf{f}$.

In a similar vein, one can show that Algorithm 2 is actually a stationary iteration for the matrix equation

$$\left(L^d L + \beta \sum_{\nu \in \mathbb{Z}_2^2 \setminus \{(0,0)\}} H_\nu^d H_\nu \right) \mathbf{f} = L^d \mathbf{g}.$$

By (15), this reduces to the following algorithm.

Algorithm 2 in Matrix Form:

$$\left(L^d L + \frac{\beta}{1-\beta} I \right) \mathbf{f} = \frac{1}{1-\beta} L^d \mathbf{g}. \tag{18}$$

Again there is no need to iterate on (14) to get $\mathbf{f}$. For periodic boundary conditions, both the matrices L and L^d in (18) are BCCB matrices of size $M_1M_2 \times M_1M_2$ and hence (18) can be solved efficiently by using three 2-dimensional FFTs in $O(M_1M_2 \log(M_1M_2))$ operations, see [8, §5.2.2]. For symmetric boundary conditions, both matrices L and L^d are block Toeplitz-plus-Hankel matrices with Toeplitz-plus-Hankel blocks. Thus (18) can be solved by using three 2-dimensional FCTs in $O(M_1M_2 \log(M_1M_2))$ operations (see [18]). It is interesting to note that (18) is similar to the Tikhonov least squares method (7), except that instead of L^*, we use L^d.

In (16), it is easy to further decompose $H_\nu \mathbf{f}_n$, $\nu \in \mathbb{Z}_2^2 \setminus \{(0,0)\}$, by applying the matrices L and H_γ, $\gamma \in \mathbb{Z}_2^2 \setminus \{(0,0)\}$, to obtain the wavelet packet decomposition of $\mathbf{f}_n$. Then we can apply the threshold denoising method we have discussed in §3.1. To present the matrix form of Algorithm 3, we define Donoho's thresholding operator. For a given λ, let

$$\mathcal{D}_\lambda((x_1, \ldots, x_l, \ldots)^T) = (t_\lambda(x_1), \ldots, t_\lambda(x_l), \ldots)^T, \tag{19}$$

where the thresholding function t_λ is either (i) $t_\lambda(x) = x\chi_{|x|>\lambda}$, referred to as the *hard threshold*, or (ii) $t_\lambda(x) = \operatorname{sgn}(x)\max(|x| - \lambda, 0)$, the *soft threshold.*

Algorithm 3 in Matrix Form:

(i) Choose an initial approximation $\mathbf{f}_0$ *(e.g.* $\mathbf{f}_0 = L^d\mathbf{g}$*);*

(ii) Iterate until convergence:

$$\mathbf{f}_{n+1} = L^d \mathbf{g} + \sum_{\nu \in \mathbb{Z}_2^2 \setminus \{(0,0)\}} H_\nu^d \mathcal{T}\left(H_\nu \mathbf{f}_n\right).$$

Here

$$\mathcal{T}\left(H_\nu \mathbf{f}_n\right) = (L^d)^J (L)^J (H_\nu \mathbf{f}_n) + \sum_{j=0}^{J-1} (L^d)^j \sum_{\gamma \in \mathbb{Z}_2^2 \setminus \{(0,0)\}} H_\gamma^d \mathcal{D}_{\lambda_{n,\nu}} \left(H_\gamma (L)^j H_\nu \mathbf{f}_n\right) \tag{20}$$

with $\mathcal{D}_{\lambda_{n,\nu}}$ *given in (19) and* $\lambda_{n,\nu} = \sigma_{n,\nu}\sqrt{2\log(M_1M_2)}$ *where* $\sigma_{n,\nu}$ *is the variance of* $H_\nu \mathbf{f}_n$ *estimated numerically by the method provided in [7].*

(iii) Let $\mathbf{f}_{n_0}$ *be the final iterate from Step (ii). The final solution of our Algorithm is*

$$\mathbf{f}_c = (L^d)^J (L)^J \mathbf{f}_{n_0} + \sum_{j=0}^{J-1} (L^d)^j \sum_{\gamma \in \mathbb{Z}_2^2 \setminus \{(0,0)\}} H_\gamma^d \mathcal{D}_{\lambda_{n_0}} \left(H_\gamma (L)^j \mathbf{f}_{n_0}\right),$$

where $\mathcal{D}_{\lambda_{n_0}}$ *is the thresholding operator used in Step (ii).*

According to [7], the choice of $\lambda_{n,\nu}$ in Step (ii) is a good thresholding level for orthonormal wavelets as well as biorthonormal wavelets.

The computational complexity of Algorithm 3 depends on the number of iterations required for convergence. In each iteration, we essentially go through a J-level wavelet decomposition and reconstruction procedure once, therefore it needs $O(M_1M_2)$ operations. As for the value of J, the larger it is, the finer the wavelet packet decomposition of $\mathbf{f}_n$ will be before it is denoised. This leads to a better denoising scheme. However, a larger J will cost slightly more computational time. From our numerical tests, we find that it is already good enough to choose J to be either 1 or 2. The variance $\sigma_{n,\nu}$ is estimated by the method given in [7] which uses the median of the absolute value of the entries in the vector $H_\nu \mathbf{f}_n$. Hence the cost of computing $\sigma_{n,\nu}$ is $O(M_1M_2 \log(M_1M_2))$, see for instance [21]. Finally, the cost of Step (iii) is less than one additional iteration of Step (ii). One nice feature of Algorithm 3 is that it is parameter-free — we do not have to choose the regularization parameter β as in the Tikhonov method (7) or Algorithm 2.

For the super-resolution case, where only some of the sub-samples $a * v(\nu - 2\alpha)$, $\nu \in A \subset \mathbb{Z}_2^2$, are available, we can first apply an interpolatory scheme (e.g. interpolatory subdivision scheme in [6] and [10]) to obtain the full set of the sample $\mathbf{w}$ approximately. Then, with $\mathbf{w}$ as the observed image, we find an approximation solution $\mathbf{u}$ from either Algorithm 2 or Algorithm 3. To tune up the result, we can compute $L\mathbf{u}$ to obtain $a * u(\nu - 2\alpha)$, $\nu \in \mathbb{Z}_2$, and replace the component $a * u(\nu - 2\alpha)$, $\nu \in A$, by the sample data, i.e. $a * v(\nu - 2\alpha)$, $\nu \in A$. With this new observed high-resolution image, we again use either Algorithm 2 or Algorithm 3 to get the final high-resolution image $\mathbf{f}$.

4 Numerical Experiments

In this section, we implement the wavelet algorithms developed in the last section to 1D and 2D examples and compare them with the Tikhonov least squares method. We evaluate the methods using the relative error (RE) and the peak signal-to-noise ratio (PSNR) which compare the reconstructed signal (image) $\mathbf{f}_c$ with the original signal (image) $\mathbf{f}$. They are defined by

$$\text{RE} = \frac{\|\mathbf{f} - \mathbf{f}_c\|_2}{\|\mathbf{f}\|_2}$$

and

$$\text{PSNR} = 10 \log_{10} \frac{\|\mathbf{f}\|_2^2}{\|\mathbf{f} - \mathbf{f}_c\|_2^2},$$

for 1D signals and

$$\text{PSNR} = 10 \log_{10} \frac{255^2 NM}{\|\mathbf{f} - \mathbf{f}_c\|_2^2},$$

for 2D images respectively, where the size of the signals (images) is $N \times M$.

In our tests, $N = 1$ for 1D signals while $N = M$ for 2D images. For the Tikhonov method (7), we will use the identity matrix I as the regularization operator R. For both (7) and Algorithm 2, the optimal regularization parameters β^* are chosen by trial and error so that they give the best PSNR values for the resulting equations. For Algorithm 3, we use the hard thresholding for $\mathcal{D}_\lambda$ in (19) and $J = 1$ in (20), and we stop the iteration as soon as the values of PSNR peaked.

4.1 Numerical Simulation for 1D Signals

To emphasize that our algorithms work for general deblurring problems, we first apply our algorithms to two 1D blurred and noisy signals. The blurring are done by the filter given in Example 2. The

matrices L^d, L, H_ν^d and H_ν, $\nu = 1,2,3$, are generated by the corresponding univariate symbols of the refinement and wavelet masks $\widehat{m^d}$, $\widehat{m}$, $\widehat{r_\nu^d}$ and $\widehat{r}_\nu$ respectively (with either periodic or symmetric boundary conditions). For example, the matrix L for the periodic boundary condition is given by the matrix L_4 in (6) (cf. §4.2.1 for how to generate the other matrices).

The Tikhonov method (7) and Algorithm 2 reduce to solving the linear systems

$$(L^*L + \beta I)\mathbf{f} = L^*\mathbf{g} \quad \text{and} \quad \left(L^dL + \frac{\beta}{1-\beta}I\right)\mathbf{f} = \frac{1}{1-\beta}L^d\mathbf{g}$$

respectively. For Algorithm 3, Steps (ii) and (iii) become

$$\mathbf{f}_{n+1} = L^d\mathbf{g} + \sum_{\nu=1}^{3} H_\nu^d \left(L^dLH_\nu\mathbf{f}_n + \sum_{\gamma=1}^{3} H_\gamma^d \mathcal{D}_{\lambda_{n,\nu}} \left(H_\gamma H_\nu \mathbf{f}_n\right)\right),$$

$$\mathbf{f}_c = L^dL\mathbf{f}_{n_0} + \sum_{\nu=1}^{3} H_\nu^d \mathcal{D}_{\lambda_{n,\nu}} \left(H_\nu \mathbf{f}_{n_0}\right).$$

The 1D signals in our test are taken from the WaveLab Toolbox at "`http://www-stat.stanford.edu/~wavelab/`" developed by Donoho's research group. Figure 2(a) shows the first original signal $\mathbf{f}$. Figure 2(b) depicts the blurred and contaminated signal with white noises at SNR = 25. The results of deblurring by (7), Algorithm 2 and Algorithm 3 with periodic boundary conditions are shown in Figures 2(c)–(e) respectively.

It is clear from Figure 2 that Algorithm 3 outperforms the other two methods. When the symmetric boundary condition is used, the numerical results for each algorithm are almost the same as that of the corresponding algorithms with the periodic boundary condition. Hence, we omit the figures here. The similarity of the performance for the two different boundary conditions is due to the fact that for the given filter, the extensions of the signal by both boundary conditions are very close (note that the original signal has almost the same values at the two end points).

In the second example (Figure 3), the different boundary conditions lead to different extensions of the signal. It is therefore not surprising to see in Figure 3 that the symmetric boundary condition gives better PSNR values and visual quality than those of the periodic one. We omit the figures generated by Algorithm 2 for this example. The numerical results from both tests show clearly that Algorithm 3 (the thresholding method) outperforms the regularization methods (the least squares method and Algorithm 2).

4.2 High-Resolution Image Reconstruction

This section illustrates the effectiveness of the high-resolution image reconstruction algorithm derived from the wavelet analysis. We use the "Boat" image of size 263 × 263 shown in Figure 4(a) as the original image in our numerical tests. To simulate the real world situations, the pixel values of the low-resolution images near the boundary are obtained from the discrete equation of (2) by using the actual pixel values of the "Boat" image instead of imposing any boundary conditions on these pixels.

4.2.1 2×2 Sensor Array

For 2×2 sensor arrays, (2) is equivalent to blurring the true image with a 2-dimensional lowpass filter a which is the tensor product of the lowpass filter given in Example 1. Gaussian white noises are added to the resulting blurred image, and it is then chopped to size 256 × 256 to form our observed

high-resolution image $\mathbf{g}$. We note that the four 128×128 low-resolution frames can be obtained by downsampling $\mathbf{g}$ by a factor of 2 in both the horizontal and the vertical directions.

The vector $\mathbf{g}$ is then used in the Tikhonov method (7), Algorithm 2 and Algorithm 3 to recover the high-resolution image vector $\mathbf{f}$. We recall from §2 that the matrix system relating $\mathbf{g}$ and $\mathbf{f}$ is not a square system; and in order to recover $\mathbf{f}$, we impose boundary assumptions on $\mathbf{f}$ to make the matrix system a square system, see (4). We have tested both periodic and symmetric boundary conditions for all three methods. For simplicity, we will only present the details for the periodic case. The case for the symmetric boundary condition can be derived analogously, see [18, 2].

In what follows, all images are viewed as column vectors by reordering the entries of the images in a row-wise order. For the periodic boundary conditions, both the Tikhonov method and Algorithm 2 are BCCB systems and hence can be solved efficiently by three 2-dimensional FFTs, see [8, §5.2.2]. For symmetric boundary conditions, the systems can be solved by three 2-dimensional FCTs, see [18]. For Algorithm 3, we have $L = L_2 \otimes L_2$, $H_{(0,1)} = L_2 \otimes H_2$, $H_{(1,0)} = H_2 \otimes L_2$, $H_{(1,1)} = H_2 \otimes H_2$, $L^d = L_2^d \otimes L_2^d$, $H_{(0,1)}^d = L_2^d \otimes H_2^d$, $H_{(1,0)}^d = H_2^d \otimes L_2^d$, and $H_{(1,1)}^d = H_2^d \otimes H_2^d$. Here L_2 is given in (6) and

$$L_2^d \equiv \frac{1}{8}\begin{bmatrix} 6 & 2 & -1 & & & & -1 & 2 \\ 2 & 6 & 2 & -1 & & & & -1 \\ -1 & 2 & 6 & 2 & -1 & & & \\ & -1 & 2 & 6 & 2 & -1 & & \\ & & \ddots & \ddots & \ddots & \ddots & \ddots & \\ & & & -1 & 2 & 6 & 2 & -1 \\ -1 & & & & -1 & 2 & 6 & 2 \\ 2 & -1 & & & & -1 & 2 & 6 \end{bmatrix}, \tag{21}$$

$$H_2 \equiv \frac{1}{8}\begin{bmatrix} 2 & -6 & 2 & 1 & & & & 1 \\ 1 & 2 & -6 & 2 & 1 & & & \\ & 1 & 2 & -6 & 2 & 1 & & \\ & & \ddots & \ddots & \ddots & \ddots & \ddots & \\ & & & 1 & 2 & -6 & 2 & 1 \\ 1 & & & & 1 & 2 & -6 & 2 \\ 2 & 1 & & & & 1 & 2 & -6 \\ -6 & 2 & 1 & & & & 1 & 2 \end{bmatrix}, \quad H_2^d \equiv \frac{1}{4}\begin{bmatrix} 1 & & & & 1 & -2 \\ -2 & 1 & & & & 1 \\ 1 & -2 & 1 & & & \\ & \ddots & \ddots & \ddots & & \\ & & & 1 & -2 & 1 \end{bmatrix}. \tag{22}$$

Tables 1 and 2 give the PSNR and RE values of the reconstructed images for different Gaussian noise levels, the optimal regularization parameter β^* for the Tikhonov method and Algorithm 2 and also the number of iterations required for Step (ii) in Algorithm 3. For the periodic boundary condition, Algorithm 2 and Algorithm 3 are comparable and both are better than the Tikhonov method. For the symmetric boundary condition, Algorithm 3 performs better than both the Tikhonov method and Algorithm 2. In general, symmetric boundary conditions perform better than the periodic ones.

4.2.2 4×4 Sensor Array

We have done similar numerical tests for the 4×4 sensor arrays. The bivariate filters are the tensor product of the filters in Example 2. The observed high-resolution image is generated by applying the bivariate lowpass filter on the true "Boat" image. Again, true pixel values are used and no boundary conditions are assumed. After adding the noise and chopping to size 256×256, we obtain the observed

	Least Squares Model			Algorithm 2			Algorithm 3		
SNR(dB)	PSNR	RE	β^*	PSNR	RE	β^*	PSNR	RE	Iterations
30	30.00	0.0585	0.0291	32.31	0.0449	0.2596	32.34	0.0447	2
40	30.39	0.0560	0.0275	32.67	0.0430	0.2293	32.53	0.0438	2

Table 1: The results for the 2×2 sensor array with periodic boundary condition.

	Least Squares Model			Algorithm 2			Algorithm 3		
SNR(dB)	PSNR	RE	β^*	PSNR	RE	β^*	PSNR	RE	Iterations
30	32.55	0.0437	0.0173	33.82	0.0377	0.0364	34.48	0.0350	9
40	33.88	0.0375	0.0132	34.80	0.0337	0.0239	35.23	0.0321	12

Table 2: The results for the 2×2 sensor array with symmetric boundary condition.

high-resolution image $\mathbf{g}$. The image $\mathbf{g}$ can be downsampled by a factor of 4 in both the horizontal and the vertical directions to generate the sixteen 64×64 low-resolution frames. We emphasize that the process is the same as obtaining the low-resolution frames via (2). The vector $\mathbf{g}$ is then used in the Tikhonov method, Algorithm 2 and Algorithm 3 to recover $\mathbf{f}$. Again, all three methods here make use of either the periodic or the symmetric boundary conditions to form the coefficient matrix L. The matrices L_4, L_4^d, H_ν and H_ν^d, $\nu = 1, 2, 3$, can be generated by the corresponding filters in Example 2 like what we did in §4.2.1.

	Least Squares Model			Algorithm 2			Algorithm 3		
SNR(dB)	PSNR	RE	β^*	PSNR	RE	β^*	PSNR	RE	Iterations
30	25.09	0.1031	0.0448	26.45	0.0882	0.2354	26.58	0.0868	3
40	25.13	0.1026	0.0444	26.47	0.0880	0.2313	26.59	0.0867	3

Table 3: The results for the 4×4 sensor array with periodic boundary condition.

From Tables 3 and 4, we see that the performance of Algorithm 3 is again better than that of the least squares method and Algorithm 2 in all the cases. Figure 4 depicts the reconstructed high-resolution image with noise at SNR = 30dB. As is shown in the figures, the periodic boundary condition introduces boundary artifacts in the recovered $\mathbf{f}$, while the symmetric one has less boundary artifacts. A careful comparison between Figures 4(g)–(i) reveals that Algorithm 3 gives better denoising performance than the other two methods.

4.3 Super-Resolution Image Reconstruction

In this test, we tried a partial set of the low-resolution images indexed by $A \subset \mathbb{Z}_4^2$. The following procedure is used to approximate the original high-resolution image $\mathbf{f}$:

Step 1: From the given partial set of low-resolution images, we apply an interpolatory subdivision scheme such as those in [6, 10] to obtain an approximate observed high-resolution image $\mathbf{w}$.

Step 2: Using $\mathbf{w}$ as the observed high-resolution image, we solve for the high-resolution image $\mathbf{u}$ by using the least squares model, Algorithm 2 or Algorithm 3.

	Least Squares Model			Algorithm 2			Algorithm 3		
SNR(dB)	PSNR	RE	β^*	PSNR	RE	β^*	PSNR	RE	Iterations
30	29.49	0.0621	0.0125	29.70	0.0601	0.0158	30.11	0.0579	30
40	30.17	0.0573	0.0089	30.30	0.0566	0.0101	30.56	0.0549	45

Table 4: The results for the 4×4 sensor array with symmetric boundary condition.

Step 3: After obtaining $\mathbf{u}$, we re-formulate a set of low-resolution frames from $\mathbf{u}$ by passing it through the lowpass filter and then replacing those in the set A by the given ones. Then we have a new observed high-resolution image $\mathbf{g}$;

Step 4: With this new observed high-resolution image $\mathbf{g}$, we solve for the final high-resolution image $\mathbf{f}$ by using the least squares model, Algorithm 2 or Algorithm 3.

In our test, the interpolatory filter from [6] is used in Step 1 and the subset A is chosen to be $\{(0,0),(0,2),(1,1),(1,3),(2,0),(2,2),(3,1),(3,3)\}$, see Figure 5. As in §4.2.2, the tensor product of the lowpass filter m in Example 2 is used to generate the low-resolution images, and white noises at SNR = 40dB are added. Table 5 shows the results of the least squares model, and Algorithms 2 and 3 with symmetric boundary conditions. The optimal β^* in Step 2 and Step 4 are 0.0121 and 0.0105 for the least squares method and 0.0170 and 0.0161 for Algorithm 2 respectively. The total number of iterations for Algorithm 3 in Step 2 and Step 4 is 35. Figure 6(a) is the approximation of the observed low-resolution image after the interpolatory subdivision scheme (i.e. it is the vector $\mathbf{w}$ in Step 1) and Figure 6(b) is the resulting picture from our super-resolution algorithm with Algorithm 3 (i.e. the vector $\mathbf{f}$ in Step 4).

Least Squares Model		Algorithm 2		Algorithm 3	
PSNR	RE	PSNR	RE	PSNR	RE
27.44	0.0787	27.82	0.0753	28.03	0.0734

Table 5: The results of the super-resolution image reconstruction.

5 Concluding Remarks

Using examples in high-resolution image reconstruction, we have shown that our new wavelet thresholding algorithm is better than the traditional Tikhonov least squares algorithm. We emphasize that the main issue here is essentially deconvolving noisy data by wavelet approach. Our new algorithm works not only for high-resolution image reconstruction, but also for more general deblurring problems, as the 1-D examples in §4.1 have shown.

Appendix. Analysis via Residuals

In this appendix, we explain through the residual analysis, why (17) is the right equation to solve for the image reconstruction problem. For this, we first derive (17) by analyzing the observed function g given in (10). Since ϕ^d is refinable and since $S^1(\phi^d)$ is a half integer shift invariant space, $\phi^d(\cdot - \alpha/2)$, $\alpha \in \mathbb{Z}^2$, are in $S^1(\phi)$. Hence, g is in $S^1(\phi^d)$ and can be written as

$$g = 2 \sum_{\alpha \in \mathbb{Z}^2} h(\alpha) \phi^d(2 \cdot - \alpha).$$

By substituting $\phi^d(\cdot)$ and its half integer translates in (10) by

$$\phi^d = 2\sum_{\alpha\in\mathbb{Z}^2} a^d(\alpha)\phi^d(2\cdot-\alpha)$$

and its (1/2)-integer shifts, we see that $h = a^d(-\cdot) * a * v$. Therefore, the solutions of

$$a^d(-\cdot) * a * z = a^d(-\cdot) * a * v, \tag{23}$$

are possible approximations of v since the given observed function g can be generated from this solution. In fact, (17) is the corresponding matrix equation for (23).

On the other hand, recovering the original image from the observed high-resolution image $a * v$ is to deconvolve the equation

$$a * z = a * v. \tag{24}$$

In fact, (4) is the matrix representation of this equation.

Here, we give some residual analysis on the difference between using (24) and (23), when only numerical approximation solutions can be obtained. Let v_1 be a numerical solution of (24) and v_2 be a numerical solution of (23). Define

$$f_1 := 2\sum_{\alpha\in\mathbb{Z}^2} v_1(\alpha)\phi^d(2\cdot-\alpha),$$

and

$$f_2 := 2\sum_{\alpha\in\mathbb{Z}^2} v_2(\alpha)\phi^d(2\cdot-\alpha).$$

Then f_1 and f_2 are the approximations of f corresponding to v_1 and v_2. Then,

$$g_1 = \sum_{\alpha\in\mathbb{Z}^2} (a * v_1)(\alpha)\phi^d(\cdot-\alpha/2)$$

and

$$g_2 = 2\sum_{\alpha\in\mathbb{Z}^2} (a^d(-\cdot) * a * v_2)(\alpha)\phi^d(2\cdot-\alpha)$$

are the observed functions of f_1 and f_2 respectively, which are the approximations of g, the observed function of f. Since only g is available, we may compare the difference between g and g_1 and also the difference between g and g_2 in terms of the residual errors of v_1 and v_2 respectively.

Since the system $\phi^d(2\cdot-\alpha)$, $\alpha\in\mathbb{Z}^2$, is a Riesz basis of $S^1(\phi^d)$, we will have

$$c\|a^d(-\cdot)a * v - a^d(-\cdot) * a * v_2\|_2 \le \|g - g_2\|_2 \le C\|a^d(-\cdot)a * v - a^d(-\cdot) * a * v_2\|_2. \tag{25}$$

The upper bound of (25) indicates that the corresponding g_2 is a good approximation of g as long as v_2 has a small residual error. The lower bound of (25) asserts that any good approximation of g must come from the solutions of (23) which have small residual errors. More precisely, let

$$f_3 := 2\sum_{\alpha\in\mathbb{Z}^2} v_3(\alpha)\phi^d(2\cdot-\alpha),$$

be an arbitrary approximate solution of f, and g_3 be the corresponding observed function. If $\|g - g_3\|_2$ is small, the lower bound estimate of (25) asserts that $\|a^d(-\cdot) * a * v - a^d(-\cdot) * a * v_3\|_2$ must be small.

On the other hand, since the system $\phi^d(\cdot - \alpha/2)$, $\alpha \in \mathbb{Z}^2$, normally is not a Riesz system (only the upper Riesz bound in (8) holds for this system), we only have

$$\|g - g_1\|_2 \leq C\|a * v - a * v_1\|_2. \tag{26}$$

This indicates that those solutions of (24) with small residual errors will have their observed function close to g. However, the lack of lower bound estimate for (26) indicates that not all good approximations of g necessarily come from the solutions of (24) with small residual errors.

References

[1] N. Bose and K. Boo, *High-Resolution Image Reconstruction with Multisensors*, International Journal of Imaging Systems and Technology, 9 (1998), 294–304.

[2] C. Brislawn, *Classification of Nonexpansive Symmetric Extension Transforms for Multirate Filter Banks*, Appl. Comput. Harmon. Anal., 3 (1996), 337–357.

[3] D. Capel and A. Zisserman, *Super-Resolution Enhancement of Text Image Sequences*, in Proc. International Conference on Pattern Recognition, 2000.

[4] R. Coifman and D. Donoho, *Wavelet and Statistics, Springer Lecture Notes in Statistics 103.* New York: Springer-Verlag, 1994, 125–150.

[5] I. Daubechies, *Ten Lectures on Wavelets*, CBMS Conference Series in Applied Mathematics, Vol. 61, SIAM, Philadelphia, 1992.

[6] G. Deslauriers and S. Dubuc, *Symmetric Iterative Interpolation Processes*, Constr. Approx., 5 (1989), 49–68.

[7] D. Donoho, *De-Noising by Soft-Thresholding*, IEEE Trans. Inform. Theory, 41 (1995), 613–627.

[8] R. Gonzalez and R. Woods, *Digital Image Processing*, Addison-Wesley, 1993.

[9] J. Goyette, G. Lapin, M. Kang and A. Katsaggelos, *Resolution Improvement of Autoradiographs Using Image Restoration Techniques,* IEEE Engineering in Medicine and Biology Magazine, 13 (1994), 571–574.

[10] H. Ji, S. Riemenschneider and Z. Shen, *Multivariate Compactly Supported Fundamental Refinable Functions, Dual and Biorthogonal Wavelets*, Studies in Appl. Math., 102 (1999), 173–204.

[11] S. Jefferies and J. Christou, *Restoration of Astronomical Images by Iterative Blind Deconvolution*, Astrophysics J., 415 (1993), 862–874.

[12] J. Kalifa, S. Mallat, *Thresholding Estimators for Inverse Problems and Deconvolutions*, submitted to Annals of Statistics.

[13] J. Kalifa, S. Mallat, B. Roug, *Minimax Deconvolution in Mirror Wavelet Bases*, submitted to IEEE Trans. Image Process..

[14] E. Kaltenbacher and R. Hardie, *High Resolution Infrared Image Reconstruction using Multiple, Low Resolution, Aliased Frames*, Proc. of IEEE 1996 National Aerospace and Electronic Conf. NAECON, 2 (1996), 702–709.

[15] S. Kim, N. Bose, and H. Valenzuela, *Recursive Reconstruction of High Resolution image from Noisy Undersampled Multiframes*, IEEE Trans. Acoust., Speech, and Signal Process., 38 (1990), 1013–1027.

[16] W. Lawton, S. Lee and Z. Shen, *Stability and Orthonormality of Multivariate Refinable Functions*, SIAM J. Matrix Anal. Appls., 28 (1997), 999–1014.

[17] S. Mallat, *A Wavelet Tour of Signal Processing*, Academic Press, 2nd edition, 1999.

[18] M. Ng, R. Chan, and W. Tang, *A Fast Algorithm for Deblurring Models with Neumann Boundary Conditions*, SIAM J. Sci. Comput., 21 (2000), 851–866.

[19] W. Rudin, *Real and Complex Analysis*, New York: McGraw-Hill, 1987.

[20] Z. Shen, *Extension of Matrices with Laurent Polynomial Entries*, Proceedings of the 15th IMACS World Congress on Scientific Computation Modeling and Applied Mathematics, Ashim Syclow eds., 1 (1997), 57–61.

[21] S. Skiena, *The Algorithm Design Manual*, Telos/Springer-Verlag, 1997.

[22] R. Schultz and R. Stevenson, *Extraction of High-Resolution Frames from Video Sequences*, IEEE Trans. Image Proces., 5 (1996), 996–1011.

[23] A. Tekalp, M. Ozkan, and M. Sezan, *High-Resolution Image Reconstruction from Lower-Resolution Image Sequences and Space-Varying Image Restoration*, In Proc. IEEE Int. Conf. Acoust., Speech, and Signal Process., III, pp. 169–172, San Francisco, CA, March 1992.

[24] R. Tsai and T. Huang, *Multiframe Image Restoration and Registration*, Advances in Computer Vision and Image Processing, 1 (1984), 317–339.

[25] M. V. Wickerhauser, *Adapted Wavelet Analysis from Theory to Software*, Massachusetts: A. K. Peters, 1994.

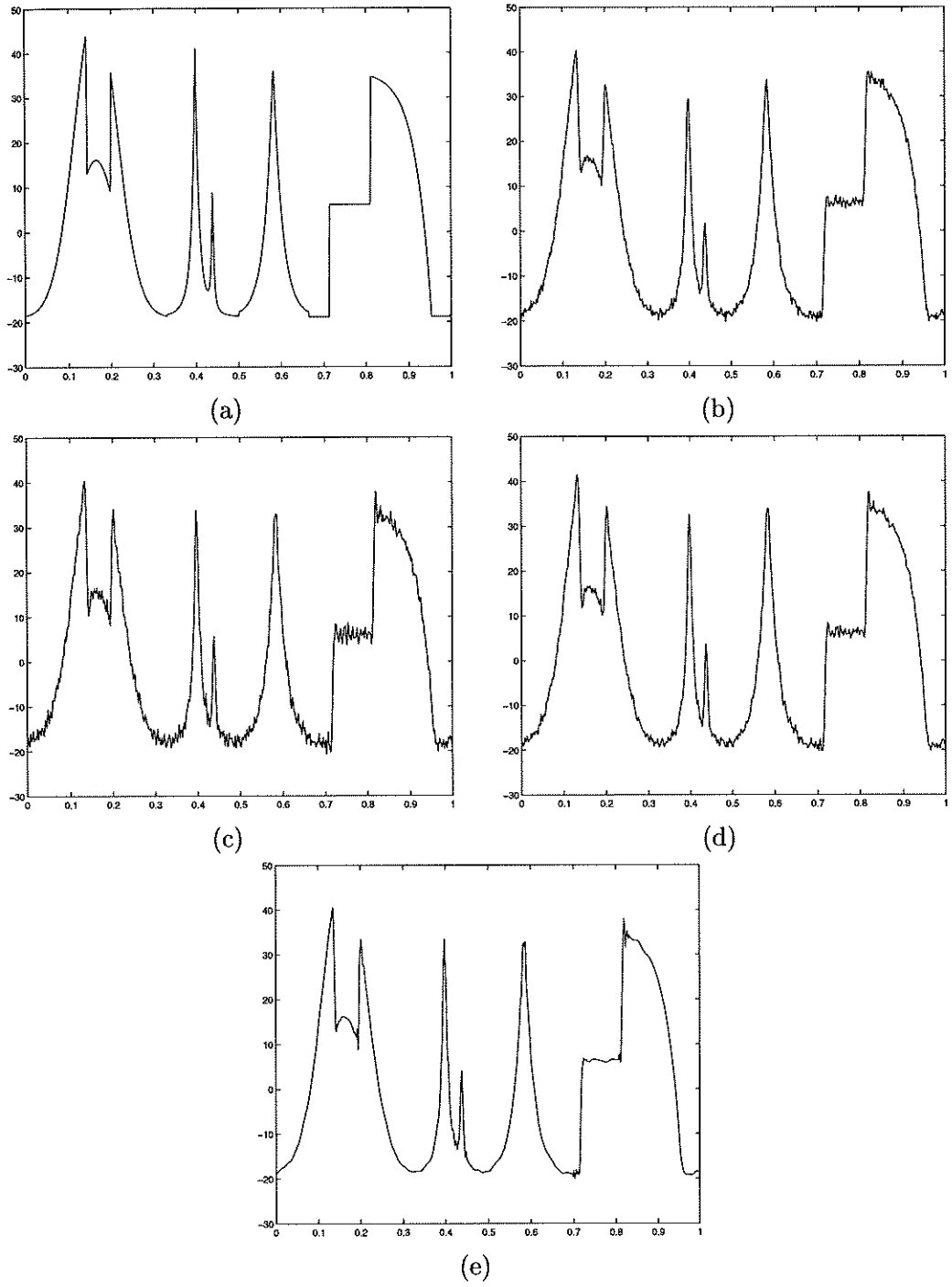

Figure 2: (a) Original signal; (b) Observed signal, blurred by the filter a in Example 2 and contaminated by white noise at SNR = 25; (c) Reconstructed signal from the least squares method with the identity regularization (PSNR = 43.19dB, RE = 0.0987, $\beta^* = 0.0402$); (d) Reconstructed signal from Algorithm 2 (PSNR = 44.44dB, RE = 0.0855, $\beta^* = 0.2205$); (e) Reconstructed signal from Algorithm 3 after 12 iterations (PSNR = 45.75dB, RE = 0.0735).

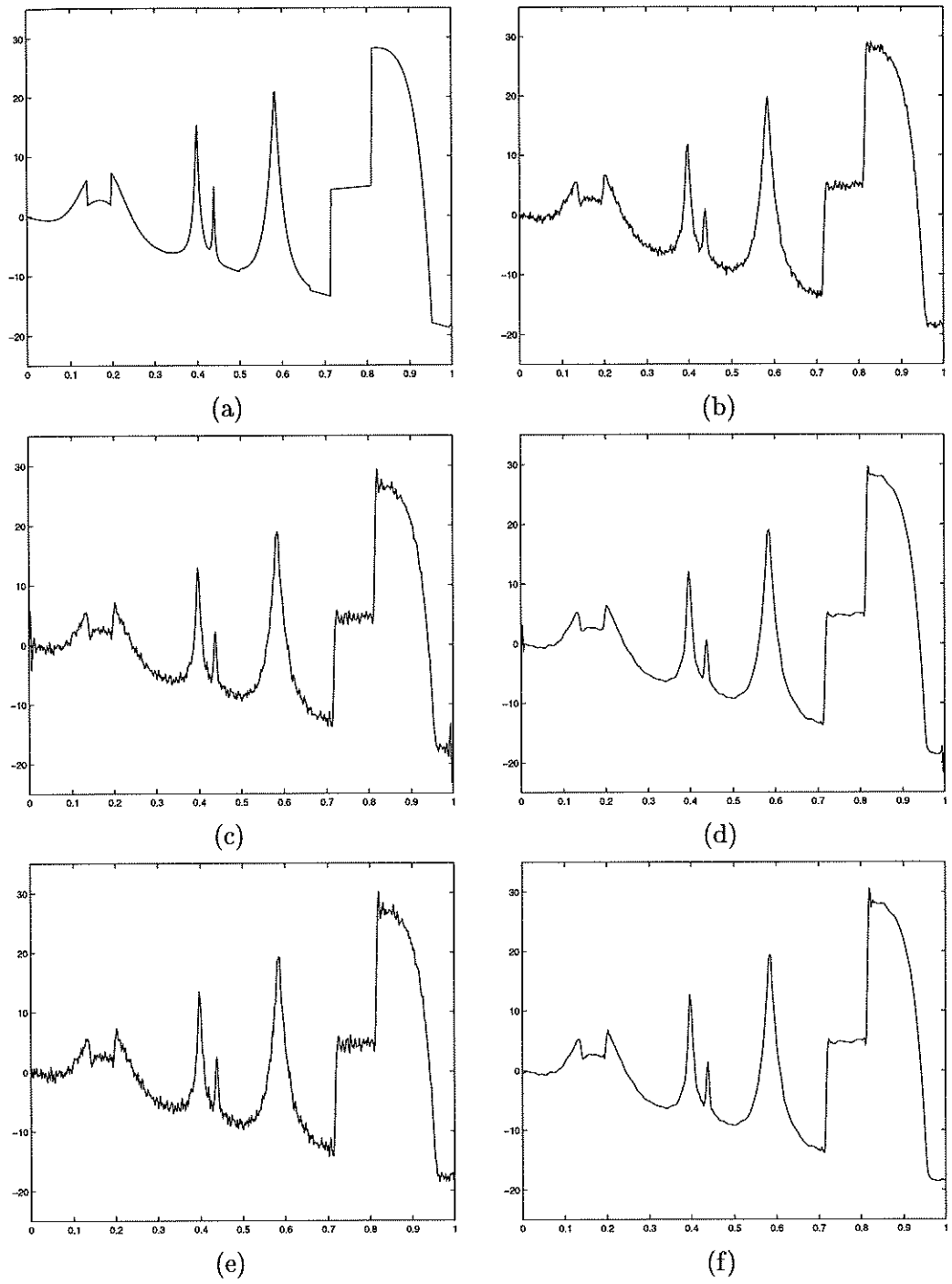

Figure 3: (a) Original signal; (b) Observed signal, blurred by the filter a in Example 2 and contaminated by white noise at SNR = 25; (c) Reconstructed signal from the least squares method with the identity regularization and periodic boundary conditions (PSNR = 45.80dB, RE = 0.1183, $\beta^* = 0.0619$); (d) Reconstructed signal from Algorithm 3 with periodic boundary condition and after 2 iterations (PSNR = 48.90dB, RE = 0.0828); (e) Reconstructed signal from the least squares method with the identity regularization and symmetric boundary condition (PSNR = 47.86dB, RE = 0.0933, $\beta^* = 0.0421$); (f) Reconstructed signal from Algorithm 3 with symmetric boundary condition and after 5 iterations (PSNR = 50.92dB, RE = 0.0657).

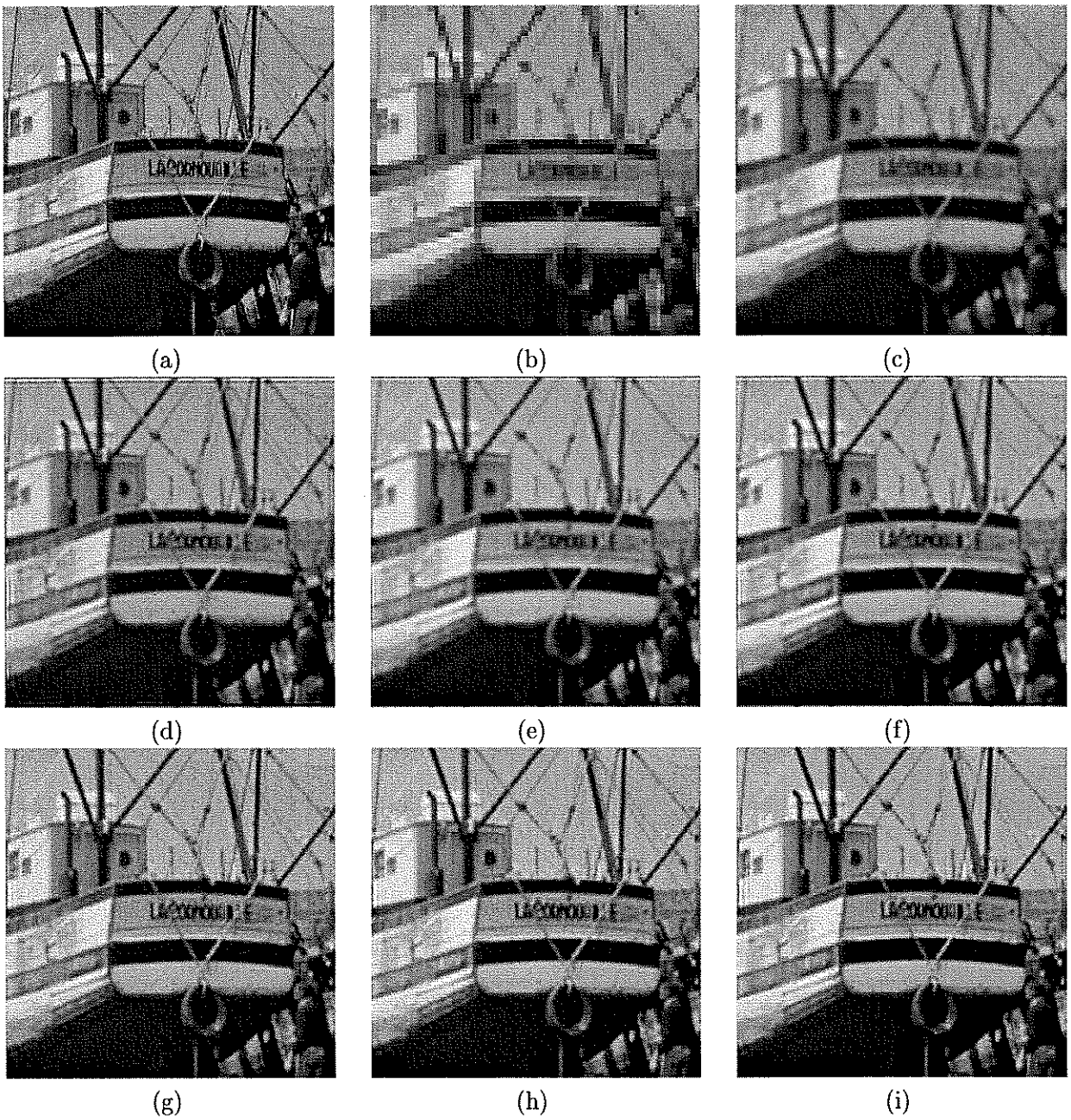

Figure 4: (a) The original "Boat" image; (b) Low-resolution 64×64 image from the $(0,0)$th sensor; (c) Observed high-resolution 256×256 image (with white noise at SNR=30dB added); (d) Reconstructed 256×256 image from the least squares method with periodic boundary condition; (e) Reconstructed 256×256 image from Algorithm 2 with periodic boundary condition; (f) Reconstructed 256×256 image from Algorithm 3 with periodic boundary condition; (g) Reconstructed 256×256 image from the least squares method with symmetric boundary condition; (h) Reconstructed 256×256 image from Algorithm 2 with symmetric boundary condition; (i) Reconstructed 256×256 image from Algorithm 3 with symmetric boundary condition.

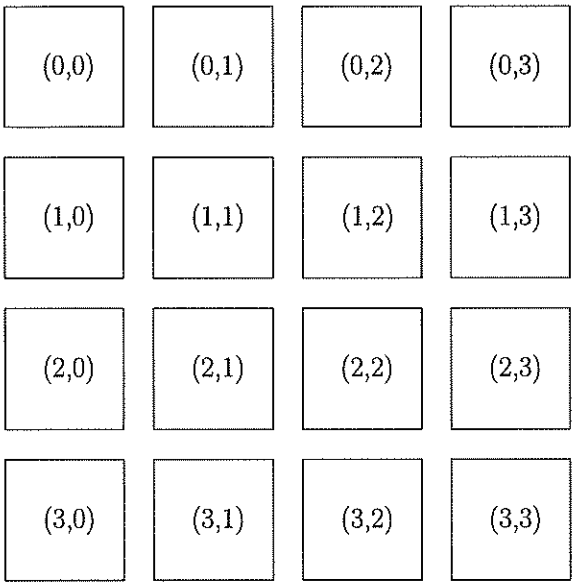

Figure 5: The 4×4 sensor array.

(a) (b)

Figure 6: (a) Approximation of the observed low-resolution image; (b) The reconstructed high-resolution image using Algorithm 3.