Signal and Image Denoising via Wavelet Thresholding: Orthogonal and Biorthogonal, Scalar and Multiple Wavelet Transforms

Vasily Strela and Andrew Walden

1 Introduction

The method of signal denoising via wavelet thresholding was popularised by Donoho and Johnstone (1994, 1995) and is now widely applied in science and engineering. It is based on thresholding of wavelet coefficients arising from the standard scalar orthogonal discrete wavelet transform (DWT). Recently this approach has been extended to incorporate thresholding coefficients arising from the discrete multiple wavelet transform (DMWT). Complications are introduced by the fact that non-orthogonal prefilters are required before the DMWT. Strela *et al.* (1999) applied scalar thresholding to the output from the Geronimo *et al.* (1994) DMWT with prefiltering, while Downie and Silverman (1998) gave a multivariate thresholding approach which takes account of the correlation in the DMWT coefficients induced by the prefiltering.

In this paper we study wavelet thresholding in the context of scalar orthogonal, scalar biorthogonal, multiple orthogonal and multiple biorthogonal wavelet transforms. Two types of multiwavelet thresholding are considered: scalar and vector. Both of them take into account the covariance structure of the transform. The form of the universal threshold is carefully formulated and is the key to the excellent results obtained in the extensive numerical simulations of signal and image denoising reported here.

Sections 2 to 5 are concerned with the deterministic formulation of relevant components of multiwavelet analysis. In Section 2 we give a summary of multiresolution analysis and semi-orthogonal, orthogonal and biorthogonal multiwavelet functions. The rest of the paper concentrates on the most-used practical cases of multiplicity 1 (scalar) and 2 (two scaling functions). Section 3 introduces two classes of orthogonal multiwavelets which may be used in multifilter banks — the Geronimo-Hardin-Massopust (GHM) and Chui-Lian (CL) classes, while Section 4 discusses biorthogonal multifilter banks, denoted BiGHM and BiHermite. For multifilter banks the given scalar input signal must be associated with a sequence of length-2 vectors: oversampling and critical sampling preprocessing is carefully studied in Section 5.

Sections 6 and 7 look at two important aspects of the processing of 1D stochastic signals, the covariance structure of the output, and denoising via thresholding. If stochastic noise is the input to preprocessing followed by discrete multiwavelet transform, very variable covariance structures of the output can result; in particular the average input variance of white noise can be inflated or deflated on output as shown in Section 6 for the GHM and CL orthogonal multiwavelet transforms. For biorthogonal multiwavelet transforms the variance inflation/deflation are seen to be almost independent of sample size, but can be strongly dependent on the level to which the transform is taken. Denoising of signals via universal thresholding of the wavelet coefficients resulting from preprocessing and the discrete multiwavelet transform can be carried out by treating the output in a scalar or vector sense, as in Section 7. For the scalar case the variance inflation/deflation can be used to carefully formulate the appropriate universal threshold level, while for the vector case the covariance structure of the noise can be used likewise. Extensive simulations using four standard 1D test signals are reported. Multiwavelets outperform scalar wavelets for three out of four noisy test signals, and the Chui-Lian scaling functions and wavelets combined with repeated row preprocessing appears to be a good general method. Vector thresholding does not always outperform scalar thresholding.

Attention is turned to the denoising of 2D signals in Section 8. Firstly the algorithm for the 2D case is carefully set out. In particular the formulation of the first step of the multiwavelet transform in the horizontal and vertical directions is explicitly illustrated, and it is shown how the resulting coefficients arise via sums of Kronecker products of the matrices of filter coefficients. For thresholding in 2D images, the variance inflation/deflation factor can be simply formulated, and is used to derive the appropriate universal threshold for scalar thresholding. Extensive simulations using four standard 2D test images are discussed. Multiwavelets generally outperform scalar wavelets for image denoising for all four noisy test images, and the results are visually very impressive. Only for 'Lenna' and 'fingerprints' with signal to noise ratios of 2 do scalar wavelets perform best. As for 1D signal processing, Chui-Lian scaling functions and wavelets combined with repeated row preprocessing appears to be a good general method.

One of the most encouraging aspects of our methodology is that for both 1D and 2D cases, the reconstructed signals derived from such a good general method demonstrate much reduced noise levels — an average noise reduction gain of 7.5dB is achieved over test examples.

2 Multiresolution analysis

2.1 Multiscaling functions

The concept of multiresolution analysis can be extended from the scalar case to general dimension $r \in \mathbb{N}$, (e.g., Goodman and Lee, 1994). Let $\Phi(t) = [\phi_1(t), ..., \phi_r(t)]^T \in L^2(\mathbb{R})^r$ be a vector of functions. It generates a multiresolution analysis of multiplicity r for $L^2(\mathbb{R})$ if the subspaces

$$V_j \equiv \overline{\text{span}\,\{2^{-j/2}\phi_i(\frac{t}{2^j} - k) : 1 \le i \le r, k \in \mathbb{Z}\}}$$

are nested:

$$\cdots \subset V_3 \subset V_2 \subset V_1 \subset V_0 \subset V_{-1} \subset \cdots. \tag{2.1}$$

and the following conditions hold,

$$\overline{\bigcup_{j\in\mathbb{Z}} V_j} = L^2(\mathbb{R}) \text{and} \bigcap_{j\in\mathbb{Z}} V_j = \{0\}. \tag{2.2}$$

Equations (2.1) and (2.2) ensure that the projections of any finite energy signal onto subspaces V_j will give successive approximations of the signal. If (2.1) and (2.2) hold, together with:

- $x(\cdot)$ is a member of V_0 if and only if $x(t-k)$ is a member of V_0, for $k \in \mathbb{Z}$;
- $x(\cdot)$ is a member of V_0 if and only if $x(\frac{t}{2^j})/2^{j/2}$ is a member of V_j, for $j \in \mathbb{Z}$; and
- $\{\phi_i(t-k) : 1 \le i \le r, k \in \mathbb{Z}\}$ form a Riesz basis for V_0,

then the vector $\Phi(t)$ is a *multiscaling function*.

All scaling functions $\phi_1(t), ..., \phi_r(t)$ are in $V_0 \subset V_{-1}$, the latter having basis $\{\sqrt{2}\phi_i(2t-k) : 1 \le i \le r, k \in \mathbb{Z}\}$. Thus the multiscaling function must satisfy the two-scale dilation equation

$$\Phi(t) = \sqrt{2}\sum_k \mathbf{G}_k \Phi(2t-k), \tag{2.3}$$

where $\mathbf{G}_{k\in\mathbb{Z}} \in \ell^2(\mathbb{Z})^{r\times r}$ is an $r \times r$ matrix of filter coefficients.

The collection of scaling functions $\{\phi_{i,j,k}(t) : 1 \le i \le r, k \in \mathbb{Z}\}$, with $\phi_{i,j,k}(t) = 2^{-j/2}\phi_i(\frac{t}{2^j} - k)$ is a Riesz basis of V_j. We also write

$$[\phi_{1,j,k}(t), ..., \phi_{r,j,k}(t)]^T = \Phi_{j,k}(t).$$

For the special case of orthonormal multiscaling functions, the basis $\{\phi_i(t-k) : 1 \le i \le r, k \in \mathbb{Z}\}$ is not just a Riesz basis, but is orthonormal,

$$< \Phi(t), \Phi(t-k) >= \int \Phi(t)\Phi^T(t-k)dt = \mathbf{I}_r\delta_{k,0},\ k \in \mathbb{Z}, \tag{2.4}$$

where $\mathbf{I}_r$ is the $r \times r$ identity matrix. By substituting (2.3) in the integral the orthonormality condition implies the following for the matrices of filter coefficients:

$$\sum_k \mathbf{G}_k \mathbf{G}_{2l+k}^T = \mathbf{I}_r \delta_{0,l}, \quad l \in \mathbb{Z}. \tag{2.5}$$

In the scalar case this tells us that the sum of squares of the low-pass filter coefficients is unity, and the filter is orthogonal to its even translates.

2.2 Multiwavelet functions

Now let W_j denote a space complementing V_j in V_{j-1}, so that

$$V_{j-1} = V_j \oplus W_j, \text{ and } V_j \cap W_j = \emptyset$$

where $'\oplus'$ denotes a direct sum. Hence W_j contains the 'detail' required in going from an approximation at level j to one at level $j-1$. In view of (2.1) and (2.2) we have

$$\bigoplus_{j \in \mathbb{Z}} W_j = L^2(\mathbb{R}).$$

Let $\Psi(t) = [\psi_1(t), ..., \psi_r(t)]^T \in L^2(\mathbb{R})^r$. For $j \in Z$ we define

$$W_j \equiv \overline{\text{span}\,\{2^{-j/2}\psi_i(\frac{t}{2^j} - k) : 1 \le i \le r, k \in \mathbb{Z}\}}.$$

If $\{\psi_i(t-k) : 1 \le i \le r, k \in \mathbb{Z}\}$ is a Riesz basis of W_0, then $\Psi(t)$ is a multiwavelet function. The collection of wavelet functions $\{\psi_{i,j,k}(t) : 1 \le i \le r, k \in \mathbb{Z}\}$, with $\psi_{i,j,k}(t) = 2^{-j/2}\psi_i(\frac{t}{2^j} - k)$ is a Riesz basis of W_j and $\{\psi_{i,j,k}(t) : 1 \le i \le r; j, k \in \mathbb{Z}\}$ is a Riesz basis of $L^2(\mathbb{R})$. As for multiscaling functions it is convenient to write $[\psi_{1,j,k}(t), ..., \psi_{r,j,k}(t)]^T = \Psi_{j,k}(t)$.

Now all wavelet functions $\psi_1(t), ..., \psi_r(t)$ are in $W_0 \subset V_{-1}$, the latter having basis $\{\sqrt{2}\phi_i(2t-k) : 1 \le i \le r, k \in \mathbb{Z}\}$. Thus the multiscaling and multiwavelet functions must satisfy the two-scale wavelet equation

$$\Psi(t) = \sqrt{2} \sum_k \mathbf{H}_k \Phi(2t-k), \tag{2.6}$$

where $\mathbf{H}_{k \in \mathbb{Z}} \in \ell^2(\mathbb{Z})^{r \times r}$ is an $r \times r$ matrix of filter coefficients.

An orthogonal multiresolution analysis is one where the wavelet spaces W_j, as defined above, are the *orthogonal* complement of V_j in V_{j-1} :

$$V_{j-1} = V_j \oplus W_j, \text{ and } V_j \perp W_j.$$

For $j = 0$ we thus know

$$< \Phi(t-k), \Psi(t-n) >= \int \Phi(t-k)\Psi^T(t-n)dt = \mathbf{0}_r, \quad k, n \in \mathbb{Z}.$$

Substituting (2.3) and (2.6) in the integral leads to

$$\sum_k \mathbf{G}_k \mathbf{H}_{2l+k}^T = \mathbf{0}_r, \quad l \in \mathbb{Z}. \tag{2.7}$$

In the scalar case this tells us that the low-pass filter is orthogonal to the even translates of the high-pass filter. A multiwavelet function $\Psi(t)$ which gives an orthogonal multiresolution analysis is called a *semi-orthogonal* multiwavelet (Cotronei *et al.* 1998). If in addition, functions in the same space are orthonormal, then the corresponding multiwavelets are *orthonormal*. For $j = 0$,

$$< \Psi(t), \Psi(t-k) >= \int \Psi(t)\Psi^T(t-k)dt = \mathbf{I}_r \delta_{k,0} \; k \in \mathbb{Z},$$

and $\{\psi_i(t-k) : 1 \le i \le r, k \in \mathbb{Z}\}$ is an orthonormal basis of W_0. $\{\psi_{i,j,k}(t) : 1 \le i \le r; j, k \in \mathbb{Z}\}$ is an orthonormal basis of $L^2(\mathbb{R})$. Substituting (2.6) into the integral, the orthonormality condition means that the matrices of filter coefficients must satisfy,

$$\sum_k \mathbf{H}_k \mathbf{H}_{2l+k}^T = \mathbf{I}_r \delta_{0,l}, \quad l \in \mathbb{Z}. \tag{2.8}$$

In the scalar case this tells us that the sum of squares of the high-pass filter coefficients is unity, and the filter is orthogonal to its even translates.

2.3 Orthogonal Discrete Multiple Wavelet Transform

Now let $x(t) \in V_0$ with $V_1 \subset V_0$. Then define

$$\mathbf{v}_{0,k}^T =< x(t), \Phi_{0,k}(t) > \quad \text{and} \quad \mathbf{v}_{1,k}^T =< x(t), \Phi_{1,k}(t) >,$$

where the integrals defined by the inner products are component-wise. Note that $\mathbf{v}_{0,k}$ and $\mathbf{v}_{1,k}$ are $r \times 1$ column vectors. From (2.3) we have

$$\Phi_{1,k}(t) = \sum_m \mathbf{G}_m \Phi_{0,2k+m}(t),$$

so that

$$\begin{aligned} \mathbf{v}_{1,k}^T &= \int x(t) \sum_m \Phi_{0,2k+m}^T(t) \mathbf{G}_m^T dt \\ &= \sum_m \left[\int x(t) \Phi_{0,2k+m}^T(t) dt \right] \mathbf{G}_m^T = \sum_m \mathbf{v}_{0,2k+m}^T \mathbf{G}_m^T \\ \Longrightarrow \mathbf{v}_{1,k} &= \sum_m \mathbf{G}_{m-2k} \mathbf{v}_{0,m}. \end{aligned}$$

In general we obtain,

$$\mathbf{v}_{j,k} = \sum_m \mathbf{G}_{m-2k} \mathbf{v}_{j-1,m} =< x(t), \Phi_{j,k}(t) >^T . \tag{2.9}$$

Now let

$$\mathbf{w}_{1,k}^T =< x(t), \Psi_{1,k}(t) >,$$

where $\mathbf{w}_{1,k}$ is an $r \times 1$ column vector. From (2.6) we have

$$\Psi_{1,k}(t) = \sum_m \mathbf{H}_m \Phi_{0,2k+m}(t),$$

so that

$$\begin{aligned} \mathbf{w}_{1,k}^T &= \int x(t) \sum_m \Phi_{0,2k+m}^T(t) \mathbf{H}_m^T dt \\ &= \sum_m \left[\int x(t) \Phi_{0,2k+m}^T(t) dt \right] \mathbf{H}_m^T = \sum_m \mathbf{v}_{0,2k+m}^T \mathbf{H}_m^T \\ \Longrightarrow \mathbf{w}_{1,k} &= \sum_m \mathbf{H}_{m-2k} \mathbf{v}_{0,m}. \end{aligned}$$

In general we obtain,

$$\mathbf{w}_{j,k} = \sum_m \mathbf{H}_{m-2k} \mathbf{v}_{j-1,m}. \tag{2.10}$$

Equations (2.9) and (2.10) are the *analysis* equations.

For orthonormal multiscaling and multiwavelet functions, $\{2^{-1/2}\phi_i(\frac{t}{2} - k) : 1 \le i \le r, k \in \mathbb{Z}\}$ is an orthonormal basis for V_1 and $\{2^{-1/2}\psi_i(\frac{t}{2} - k) : 1 \le i \le r, k \in \mathbb{Z}\}$ is an orthonormal basis for W_1 and hence $x(t) \in V_0 = V_1 \oplus W_1$, can be written,

$$x(t) = \sum_m \mathbf{v}_{1,m}^T \Phi_{1,m}(t) + \sum_m \mathbf{w}_{1,m}^T \Psi_{1,m}(t). \tag{2.11}$$

But $\mathbf{v}_{0,k}^T =< x(t), \Phi_{0,k}(t) >$ so that

$$\begin{aligned} \mathbf{v}_{0,k}^T &= < \sum_m \mathbf{v}_{1,m}^T \Phi_{1,m}(t) + \sum_m \mathbf{w}_{1,m}^T \Psi_{1,m}(t), \Phi_{0,k}(t) > \\ &= \sum_m \mathbf{v}_{1,m}^T < \Phi_{1,m}(t), \Phi_{0,k}(t) > + \sum_m \mathbf{w}_{1,m}^T < \Psi_{1,m}(t), \Phi_{0,k}(t) > \\ &= \sum_m \mathbf{v}_{1,m}^T \mathbf{G}_{k-2m} + \sum_m \mathbf{w}_{1,m}^T \mathbf{H}_{k-2m} \\ \Longrightarrow \mathbf{v}_{0,k} &= \sum_m \mathbf{G}_{k-2m}^T \mathbf{v}_{1,m} + \sum_m \mathbf{H}_{k-2m}^T \mathbf{w}_{1,m}. \end{aligned}$$

In general the reconstruction formula for orthonormal multiscaling and multiwavelet functions takes the form,

$$\mathbf{v}_{j-1,k} = \sum_m \mathbf{G}_{k-2m}^T \mathbf{v}_{j,m} + \sum_m \mathbf{H}_{k-2m}^T \mathbf{w}_{j,m}, \tag{2.12}$$

which is the *synthesis* equation.

2.4 Biorthogonal Discrete Multiple Wavelet Transform

In the scalar case ($r = 1$) there are no compactly supported, symmetric and orthogonal wavelets with approximation order higher than one. For multiwavelets, the several scaling functions give more freedom, but attempts to make translates $\Phi(t-k)$ orthogonal generally lead to scaling functions with infinite support. It is helpful then to substitute one orthogonal basis by two biorthogonal ones: one for decomposition ('analysis'), and one for reconstruction ('synthesis'). We introduce another MRA of $L^2(\mathbb{R})$. For $j \in \mathbb{Z}$ we define

$$\widetilde{V}_j \equiv \overline{\text{span}\,\{2^{-j/2}\tilde{\phi}_i(\frac{t}{2^j} - k) : 1 \le i \le r, k \in \mathbb{Z}\}}$$

with nested subspaces

$$\cdots \subset \widetilde{V}_3 \subset \widetilde{V}_2 \subset \widetilde{V}_1 \subset \widetilde{V}_0 \subset \widetilde{V}_{-1} \subset \cdots,$$

and

$$\widetilde{W}_j \equiv \overline{\text{span}\,\{2^{-j/2}\tilde{\psi}_i(\frac{t}{2^j} - k) : 1 \le i \le r, k \in \mathbb{Z}\}}$$

such that

$$\widetilde{V}_{j-1} = \widetilde{V}_j \oplus \widetilde{W}_j, \text{ and } \widetilde{V}_j \cap \widetilde{W}_j = \emptyset.$$

We require that

$$\widetilde{W}_j \perp V_j \iff < \tilde{\Psi}(\frac{t}{2^j} - k), \Phi(\frac{t}{2^j} - n) >= \mathbf{0}_r, \ j,k,n \in \mathbb{Z},$$

and

$$W_j \perp \widetilde{V}_j \iff < \Psi(\frac{t}{2^j} - k), \tilde{\Phi}(\frac{t}{2^j} - n) >= \mathbf{0}_r, \ j,k,n \in \mathbb{Z},$$

and also that

$$< \tilde{\Phi}(\frac{t}{2^j} - k), \Phi(\frac{t}{2^j} - n) >= \mathbf{I}_r \delta_{k,n}, \ j,k,n \in \mathbb{Z},$$

and

$$< \tilde{\Psi}(\frac{t}{2^j} - k), \Psi(\frac{t}{2^m} - n) >= \mathbf{I}_r \delta_{j,m} \delta_{k,n}, \ j,k,m,n \in \mathbb{Z},$$

this last equation defining $\widetilde{\Psi}(t)$ to be the dual of $\Psi(t)$. As they define a multiresolution analysis, the dual functions satisfy

$$\tilde{\Phi}(t) = \sqrt{2}\sum_k \tilde{\mathbf{G}}_k \tilde{\Phi}(2t-k) \text{ and } \tilde{\Psi}(t) = \sqrt{2}\sum_k \tilde{\mathbf{H}}_k \tilde{\Phi}(2t-k).$$

Since we no longer are considering orthonormal multiscaling and multiwavelet functions, $y(t) \in \widetilde{V}_0 = \widetilde{V}_1 \oplus \widetilde{W}_1$, can be written,

$$y(t) = \sum_m \mathbf{r}_{1,m}^T \tilde{\Phi}_{1,m}(t) + \sum_m \mathbf{s}_{1,m}^T \tilde{\Psi}_{1,m}(t),$$

where the coefficient vectors may be found as follows:

$$\begin{aligned} < y(t), \Phi_{1,k}(t) > &= \sum_m \mathbf{r}_{1,m}^T < \tilde{\Phi}_{1,m}(t), \Phi_{1,k}(t) > \\ &+ \sum_m \mathbf{s}_{1,m}^T < \tilde{\Psi}_{1,m}(t), \Phi_{1,k}(t) > \\ &= \sum_m \mathbf{r}_{1,m}^T \mathbf{I}_r \delta_{m,k} = \mathbf{r}_{1,k}^T. \end{aligned}$$

Hence, using the same approach as leading to (2.9) , we obtain

$$\mathbf{r}_{j,k} = \sum_m \mathbf{G}_{m-2k} \mathbf{r}_{j-1,m}.$$

Similarly, we get

$$\mathbf{s}_{j,k} = \sum_m \mathbf{H}_{m-2k} \mathbf{r}_{j-1,m}.$$

Finally, we note that the synthesis equation takes the form

$$\begin{aligned} \mathbf{r}_{0,k}^T =< x(t), \Phi_{0,k}(t) > &= \sum_m \mathbf{r}_{1,m}^T < \tilde{\Phi}_{1,m}(t), \Phi_{0,k}(t) > \\ &+ \sum_m \mathbf{s}_{1,m}^T < \tilde{\Psi}_{1,m}(t), \Phi_{0,k}(t) > \\ &= \sum_m \mathbf{r}_{1,m}^T \tilde{\mathbf{G}}_{k-2m} + \sum_m \mathbf{s}_{1,m}^T \tilde{\mathbf{H}}_{k-2m} \\ \Longrightarrow \mathbf{r}_{0,k} &= \sum_m \tilde{\mathbf{G}}_{k-2m}^T \mathbf{r}_{1,m} + \sum_m \tilde{\mathbf{H}}_{k-2m}^T \mathbf{s}_{1,m}, \end{aligned}$$

and in general,

$$\mathbf{r}_{j-1,k} = \sum_m \tilde{\mathbf{G}}_{k-2m}^T \mathbf{r}_{j,m} + \sum_m \tilde{\mathbf{H}}_{k-2m}^T \mathbf{s}_{j,m}.$$

3 Orthonormal multifilter banks

3.1 The scalar case

In the scalar case $r = 1$ and $\mathbf{G}_k$ and $\mathbf{H}_k$ are just single elements, g_k and h_k say, and the filters satisfy

$$\sum_k g_k g_{2l+k} = \sum_k h_k h_{2l+k} = \delta_{0,l}, \text{ and } \sum_k g_k h_{2l+k} = 0, \quad l \in \mathbb{Z}.$$

There are many examples of filters designed for orthonormal scalar wavelet transforms. In this paper we make use of two types, due to Daubechies (1992), namely the four coefficient 'extremal phase' design, denoted D(4), and the eight coefficient 'least-asymmetric' design, denoted LA(8). Boundaries are handled in the standard way using circular periodization.

3.2 Multiscaling and wavelet functions

In practice multiscaling and wavelet functions are concerned with multiplicity $r = 2$. An important example was constructed by Geronimo, Hardin and Massopust (1994), which we shall refer to as the GHM system. GHM multiscaling functions have four coefficient matrices $\mathbf{G}_0, \mathbf{G}_1, \mathbf{G}_2$ and $\mathbf{G}_3$,

$$\mathbf{G}_0 = \begin{bmatrix} \frac{3}{5\sqrt{2}} & \frac{4}{5} \\ -\frac{1}{20} & -\frac{3}{10\sqrt{2}} \end{bmatrix}, \qquad \mathbf{G}_1 = \begin{bmatrix} \frac{3}{5\sqrt{2}} & 0 \\ \frac{9}{20} & \frac{1}{\sqrt{2}} \end{bmatrix}$$
$$\mathbf{G}_2 = \begin{bmatrix} 0 & 0 \\ \frac{9}{20} & -\frac{3}{10\sqrt{2}} \end{bmatrix}, \qquad \mathbf{G}_3 = \begin{bmatrix} 0 & 0 \\ -\frac{1}{20} & 0 \end{bmatrix}$$

and four coefficient matrices $\mathbf{H}_0, \mathbf{H}_1, \mathbf{H}_2$ and $\mathbf{H}_3$,

$$\mathbf{H}_0 = \tfrac{1}{10}\begin{bmatrix} -\frac{1}{2} & -\frac{3}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & 3 \end{bmatrix}, \qquad \mathbf{H}_1 = \tfrac{1}{10}\begin{bmatrix} \frac{9}{2} & -\frac{10}{\sqrt{2}} \\ -\frac{9}{\sqrt{2}} & 0 \end{bmatrix}$$
$$\mathbf{H}_2 = \tfrac{1}{10}\begin{bmatrix} \frac{9}{2} & -\frac{3}{\sqrt{2}} \\ \frac{9}{\sqrt{2}} & -3 \end{bmatrix}, \qquad \mathbf{H}_3 = \tfrac{1}{10}\begin{bmatrix} -\frac{1}{2} & 0 \\ -\frac{1}{\sqrt{2}} & 0 \end{bmatrix}.$$

These matrices satisfy the conditions in (2.5), (2.7) and (2.8). GHM scaling functions have four remarkable properties showing that multiwavelets can combine more useful features than scalar wavelets:

- Both scaling functions have short supports [0,1] and [0,2]. In the scalar case one would expect support [0, 3], for a scaling function satisfying a two-scale equation with 4 coefficients.
- The system has second order of approximation (constant and linear functions can be represented exactly by a linear combination of translates $\phi_1(t-k), \phi_2(t-k),\ k \in \mathbb{Z}$).
- Translates of the scaling functions are orthogonal.
- Both scaling functions and the wavelets are symmetric.

In the scalar case, symmetry, orthogonality and approximation order higher than 1 cannot be combined. The GHM multiscaling and multiwavelet functions are also quite smooth — almost differentiable.

Another example of symmetric orthogonal multiwavelets with approximation order 2 is due to Chui and Lian (1996). Both scaling functions are supported on $[0, 2]$, which is slightly longer than GHM. For the CL system, only three $\mathbf{G}$ matrices and three $\mathbf{H}$ matrices are required.

$$\mathbf{G}_0 = a_1\begin{bmatrix} 1 & -1 \\ \frac{\sqrt{7}}{2} & -\frac{\sqrt{7}}{2} \end{bmatrix}, \quad \mathbf{G}_1 = a_1\begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{G}_2 = a_1\begin{bmatrix} 1 & 1 \\ -\frac{\sqrt{7}}{2} & -\frac{\sqrt{7}}{2} \end{bmatrix}.$$
$$\mathbf{H}_0 = a_2\begin{bmatrix} 2 & -2 \\ -1 & 1 \end{bmatrix}, \quad \mathbf{H}_1 = a_2\begin{bmatrix} -4 & 0 \\ 0 & 2\sqrt{7} \end{bmatrix}, \quad \mathbf{H}_2 = a_2\begin{bmatrix} 2 & 2 \\ 1 & 1 \end{bmatrix}.$$

Here $a_1 = 1/(2\sqrt{2})$ and $a_2 = 1/(4\sqrt{2})$. CL scaling functions and wavelets are less smooth than GHM ones.

The multiwavelet analysis equations (2.9) and (2.10) can be realised as a matrix filter bank as in Fig. 1. With $r = 2$, as assumed hereafter, the 2-channel matrix filter bank operates on two input data streams, filtering them into four output data streams, each of which is downsampled by a factor of two. Although multifiltering is a convolution with dowsampling, the filter coefficients are matrices and the input has vector form. This novelty raises a problem which is unusual for the scalar case. In order to start the cascade algorithm one must preprocess scalar data in order to get vector input. There are different ways to do this preprocessing; see Section 5.

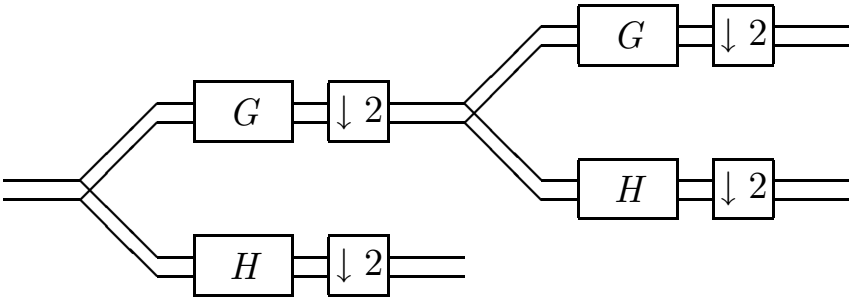


Figure 1: A multiwavelet filter bank, iterated once.

4 Biorthogonal multifilter banks

4.1 The scalar case

Here we used the biorthogonal wavelets described in Daubechies (1992) as "variations on the spline case." We chose the particular case with 9 and 7 filter coefficients for g_k and $\tilde{g}_k$; the filter coefficients are given in Daubechies (1992, Table 8.3). We shall refer to this as the Bi9-7 transform.

4.2 Multi scaling and wavelet functions and duals

Strela (1998) showed how to construct biorthogonal multiscaling functions. For the GHM system, one approximation order is taken from the analysis part of the multifilter bank and added to the synthesis part. GHM functions have two approximation orders, so the biorthogonal GHM (BiGHM) scaling functions $\tilde{\phi}_1(t), \tilde{\phi}_2(t)$ have approximation order 3, and the dual ones $\phi_1(t), \phi_2(t)$ have approximation order 1. Naturally therefore, $\tilde{\phi}_1(t), \tilde{\phi}_2(t)$ are smoother than $\phi_1(t), \phi_2(t)$. The scaling and wavelet coefficient matrices for the BiGHM

analysis and synthesis multifilters are given by

$$\mathbf{G}_0 = a_1 \begin{bmatrix} -\frac{1}{20} & \frac{1}{20} \\ -\frac{1}{20} & \frac{1}{20} \end{bmatrix}, \quad \mathbf{G}_1 = a_1 \begin{bmatrix} \frac{1}{2} & -1 \\ \frac{1}{2} & -1 \end{bmatrix}, \quad \mathbf{G}_2 = a_1 \begin{bmatrix} \frac{11}{10} & 0 \\ 0 & \frac{11}{10} \end{bmatrix},$$

$$\mathbf{G}_3 = a_1 \begin{bmatrix} \frac{1}{2} & 1 \\ -\frac{1}{2} & -1 \end{bmatrix}, \quad \mathbf{G}_4 = a_1 \begin{bmatrix} -\frac{1}{20} & -\frac{1}{20} \\ \frac{1}{20} & \frac{1}{20} \end{bmatrix}.$$

$$\mathbf{H}_0 = a_1 \begin{bmatrix} -1 & 3 \\ -\frac{2}{5} & \frac{7}{5} \end{bmatrix}, \quad \mathbf{H}_1 = a_1 \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}, \quad \mathbf{H}_2 = a_1 \begin{bmatrix} -1 & -3 \\ \frac{2}{5} & \frac{7}{5} \end{bmatrix}.$$

$$\tilde{\mathbf{G}}_0 = a_2 \begin{bmatrix} 10 & -4 \\ 15 & -7 \end{bmatrix}, \quad \tilde{\mathbf{G}}_1 = a_2 \begin{bmatrix} 20 & 0 \\ 0 & 10 \end{bmatrix}, \quad \tilde{\mathbf{G}}_2 = a_2 \begin{bmatrix} 10 & 4 \\ -15 & -7 \end{bmatrix}.$$

$$\tilde{\mathbf{H}}_0 = a_1 \begin{bmatrix} \frac{1}{16} & -\frac{3}{80} \\ \frac{1}{8} & -\frac{3}{40} \end{bmatrix}, \quad \tilde{\mathbf{H}}_1 = a_1 \begin{bmatrix} -\frac{1}{4} & \frac{1}{8} \\ -\frac{1}{2} & \frac{1}{4} \end{bmatrix}, \quad \tilde{\mathbf{H}}_2 = a_1 \begin{bmatrix} \frac{3}{8} & 0 \\ 0 & \frac{9}{20} \end{bmatrix},$$

$$\tilde{\mathbf{H}}_3 = a_1 \begin{bmatrix} -\frac{1}{4} & -\frac{1}{8} \\ \frac{1}{2} & \frac{1}{4} \end{bmatrix}, \quad \tilde{\mathbf{H}}_4 = a_1 \begin{bmatrix} \frac{1}{16} & \frac{3}{80} \\ -\frac{1}{8} & -\frac{3}{40} \end{bmatrix}.$$

Here $a_1 = 1/\sqrt{2}$, and $a_2 = 1/(20\sqrt{2})$. As usual, the smoother scaling functions are used for the synthesis step.

Another biorthogonal multifilter bank (BiHermite) comes from Hermite cubic 'finite elements.' These C^1 piecewise cubics $\tilde{\phi}_1(t)$ and $\tilde{\phi}_2(t)$ are supported on $[0, 2]$. Their value and slope are specified in the integers and their second derivative jumps at unity. The Hermite multiscaling function $\tilde{\phi}(t) = [\tilde{\phi}_1(t), \tilde{\phi}_2(t)]^T$ satisfies a dilation equation with three coefficient matrices (Strela and Strang, 1995):

$$\tilde{\mathbf{G}}_0 = \frac{\sqrt{2}}{16} \begin{bmatrix} 4 & -2 \\ 3 & -1 \end{bmatrix}, \quad \tilde{\mathbf{G}}_1 = \frac{\sqrt{2}}{16} \begin{bmatrix} 8 & 0 \\ 0 & 4 \end{bmatrix}, \quad \tilde{\mathbf{G}}_2 = \frac{\sqrt{2}}{16} \begin{bmatrix} 4 & 2 \\ -3 & -1 \end{bmatrix}.$$

Translates $\tilde{\phi}_1(t-k)$ and $\tilde{\phi}_2(t-k)$ form a (non-orthogonal) basis for the space V_0 of all C^1 piecewise cubics on unit intervals. Strela (1998) constructed a dual multiscaling function $\phi(t) = [\phi_1(t), \phi_2(t)]^T$ supported on $[0, 2]$ and with approximation order 4. Unfortunately it does not belong to L^2. In order to construct a more regular pair two approximation orders were given up and the obtained free parameters used towards numerical maximisation of the smoothness of $\phi_1(t), \phi_2(t)$. (This can be done by minimisation of the spectral radius of the transition operator). The scaling functions obtained have approximation order 2 and are continuous, (belonging to Sobolev subspace $H^{0.828}$). The scaling coefficient matrices are as follows:

$$\mathbf{G}_0 = \frac{1}{\sqrt{2}} \begin{bmatrix} -\frac{73}{648} & \frac{77}{324} \\ -\frac{773}{3240} & \frac{3229}{6480} \end{bmatrix} \quad \mathbf{G}_1 = \frac{1}{\sqrt{2}} \begin{bmatrix} \frac{1}{2} & -\frac{89}{162} \\ \frac{187}{180} & -\frac{91}{81} \end{bmatrix} \quad \mathbf{G}_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} \frac{397}{324} & 0 \\ 0 & \frac{6091}{3240} \end{bmatrix}$$

$$\mathbf{G}_3 = \frac{1}{\sqrt{2}} \begin{bmatrix} \frac{1}{2} & \frac{89}{162} \\ -\frac{187}{180} & -\frac{91}{81} \end{bmatrix} \quad \mathbf{G}_4 = \frac{1}{\sqrt{2}} \begin{bmatrix} -\frac{73}{648} & -\frac{77}{324} \\ \frac{773}{3240} & \frac{3229}{6480} \end{bmatrix}.$$

The corresponding wavelet coefficient matrices for the BiHermite biorthogonal pair are

$$\mathbf{H}_0 = a_1 \begin{bmatrix} -4 & 6 \\ -2 & 2 \end{bmatrix}, \quad \mathbf{H}_1 = a_1 \begin{bmatrix} 8 & 0 \\ 0 & 8 \end{bmatrix}, \quad \mathbf{H}_2 = a_1 \begin{bmatrix} -4 & -6 \\ 2 & 2 \end{bmatrix},$$

and

$$\tilde{\mathbf{H}}_0 = a_2 \begin{bmatrix} \frac{67}{3840} & -\frac{7}{11520} \\ \frac{95}{5184} & -\frac{1}{2592} \end{bmatrix}, \quad \tilde{\mathbf{H}}_1 = a_2 \begin{bmatrix} -\frac{1}{16} & \frac{187}{2880} \\ -\frac{89}{1296} & \frac{91}{1296} \end{bmatrix}, \quad \tilde{\mathbf{H}}_2 = a_2 \begin{bmatrix} \frac{173}{1920} & 0 \\ 0 & \frac{13}{72} \end{bmatrix},$$

$$\tilde{\mathbf{H}}_3 = a_2 \begin{bmatrix} -\frac{1}{16} & -\frac{187}{2880} \\ \frac{89}{1296} & \frac{91}{1296} \end{bmatrix}, \quad \tilde{\mathbf{H}}_4 = a_2 \begin{bmatrix} \frac{67}{3840} & \frac{7}{11520} \\ -\frac{95}{5184} & -\frac{1}{2592} \end{bmatrix}.$$

Here $a_1 = 1/(16\sqrt{2})$ and $a_2 = 8\sqrt{2}$. Again, we use the smoother Hermite cubics for the synthesis part.

5 Preprocessing and sampling

The aim of preprocessing is to associate the given scalar input signal of length N to a sequence of length-2 vectors $\{\mathbf{v}_{0,k}\}$ for input to the analysis algorithm shown in Fig. 1. Here N is assumed a multiple of a power of 2, and so is of even length. The signal $\{X_k\}$ is taken to be a function $x(\cdot)$ observed at integer time points. If the preprocessing produces N length-2 vectors it is said to be an oversampling scheme, while if it produces $N/2$ length-2 vectors the result is a critical sampling. After the wavelet reconstruction (synthesis) step, e.g., inverse DMWT, a postfilter is applied. Clearly prefiltering, wavelet transform, inverse transform, and postfiltering should recover the input signal exactly if nothing else has been done.

We say that a preprocessing preserves m approximation orders if an input sequence $X_k = k^j$, $j = 0, \ldots, m-1$, after preprocessing and filtering, gives zero high-pass output, (neglecting end effects).

5.1 Repeated row preprocessing: Oversampling

Here the input length-2 vectors are formed from the original time series via

$$\mathbf{v}_{0,k} = \begin{bmatrix} v_{0,k}^{(0)} \\ v_{0,k}^{(1)} \end{bmatrix} = \begin{bmatrix} X_k \\ \alpha X_k \end{bmatrix} \quad k = 0, ..., N-1.$$

Here α is a constant; it is typically chosen so that if $X_k = C$, a constant, for all k, then the output from the high-pass multifilter is zero. As shown in Heil, Strang and Strela (1996) this can always be done if the system has approximation order higher than zero. For the GHM case, this gives $\alpha = 1/\sqrt{2}$, since

$$\begin{bmatrix} w_{1,k}^{(0)} \\ w_{1,k}^{(1)} \end{bmatrix} = [\mathbf{H}_0 + \mathbf{H}_1 + \mathbf{H}_2 + \mathbf{H}_3] \begin{bmatrix} C \\ \frac{C}{\sqrt{2}} \end{bmatrix} = \frac{1}{10} \begin{bmatrix} 8 & -\frac{16}{\sqrt{2}} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} C \\ \frac{C}{\sqrt{2}} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

The output from the low-pass multifilter is simply a scaled version of the input:

$$\begin{bmatrix} v_{1,k}^{(0)} \\ v_{1,k}^{(1)} \end{bmatrix} = [\mathbf{G}_0 + \mathbf{G}_1 + \mathbf{G}_2 + \mathbf{G}_3] \begin{bmatrix} C \\ \frac{C}{\sqrt{2}} \end{bmatrix} = \frac{1}{5} \begin{bmatrix} \frac{6}{\sqrt{2}} & 4 \\ 4 & \sqrt{2} \end{bmatrix} \begin{bmatrix} C \\ \frac{C}{\sqrt{2}} \end{bmatrix} = \sqrt{2} \begin{bmatrix} C \\ \frac{C}{\sqrt{2}} \end{bmatrix}.$$

In the case of the CL, BiGHM and BiHermite systems $\alpha = 0$, reflecting the fact that one of the scaling functions is antisymmetric. Obviously repeated row preprocessing doubles the number of input data points, and preserves only one approximation order: if $X_k = k$ then after repeated row preprocessing the output of the high-pass multifilter will not be zero even if the corresponding multiwavelets have two vanishing moments.

If we represent the preprocessing step by a matrix multiplication, $\mathcal{P}\mathbf{X} = \mathbf{V}_0$, where $\mathcal{P}$ is $2N \times N$, $\mathbf{X}$ is $N \times 1$, and $\mathbf{V}_0$ is $2N \times 1$, then in detail we have

$$\begin{bmatrix} 1 & 0 & 0 & \cdots \\ \alpha & 0 & 0 & \cdots \\ 0 & 1 & 0 & \cdots \\ 0 & \alpha & 0 & \cdots \\ & & & \ddots \end{bmatrix} \begin{bmatrix} X_0 \\ X_1 \\ X_2 \\ X_3 \\ \vdots \end{bmatrix} = \begin{bmatrix} X_0 \\ \alpha X_0 \\ X_1 \\ \alpha X_1 \\ \vdots \end{bmatrix} = \begin{bmatrix} v_{0,0}^{(0)} \\ v_{0,0}^{(1)} \\ v_{0,1}^{(0)} \\ v_{0,1}^{(1)} \\ \vdots \end{bmatrix}.$$

5.2 Matrix (approximation) preprocessing: Critical sampling

For the GHM multifilter bank we can conveniently use another type of preprocessing which preserves the sampling rate ('critical sampling'), and approximation order 2. This method is called 'approximation' preprocessing (Strela *et al.* 1999; Xia *et al.* 1996). Let $x(t)$ belong to V_0 generated by translates of the GHM scaling functions, so that

$$x(t) = \sum_k \left[v_{0,k}^{(0)} \phi_1(t-k) + v_{0,k}^{(1)} \phi_2(t-k) \right].$$

Suppose that the input sequence samples $x(\cdot)$ at the half-integers, $X_{2k} = x(k), \quad X_{2k+1} = x(k+1/2)$. Now $\phi_1(t)$ vanishes at all integer points and the only integer at which $\phi_2(t)$ is non-zero is unity. Hence,

$$\begin{aligned} X_{2k} &= v_{0,k-1}^{(1)} \phi_2(1), \\ X_{2k+1} &= v_{0,k}^{(0)} \phi_1(1/2) + v_{0,k-1}^{(1)} \phi_2(3/2) + v_{0,k}^{(1)} \phi_2(1/2), \\ X_{2k+2} &= v_{0,k}^{(1)} \phi_2(1). \end{aligned}$$

Thus,

$$\begin{aligned} v_{0,k-1}^{(1)} &= X_{2k}/\phi_2(1), \\ v_{0,k}^{(1)} &= X_{2k+2}/\phi_2(1), \end{aligned} \tag{5.1}$$

$$\begin{aligned} v_{0,k}^{(0)} &= \frac{X_{2k+1} - v_{0,k-1}^{(1)}\phi_2(3/2) - v_{0,k}^{(1)}\phi_2(1/2)}{\phi_1(1/2)} \\ &= \frac{X_{2k+1}}{\phi_1(1/2)} - \frac{X_{2k}\phi_2(3/2)}{\phi_1(1/2)\phi_2(1)} - \frac{X_{2k+2}\phi_2(1/2)}{\phi_1(1/2)\phi_2(1)} \\ &= \frac{X_{2k+1}\phi_2(1) - \phi_2(1/2)[X_{2k} + X_{2k+2}]}{\phi_1(1/2)\phi_2(1)}, \end{aligned} \tag{5.2}$$

where we have used the fact that $\phi_2(1/2) = \phi_2(3/2)$. Equations (5.1) and (5.2) give the required vector $\mathbf{v}_{0,k}$ for $k = 0, ..., N/2 - 1$.

In fact approximation prefiltering is just a special case of matrix prefiltering,

$$\mathbf{v}_{0,k} = \sum_{m=0}^{M} \mathbf{P}_m \begin{bmatrix} X_{2(m+k)} \\ X_{2(m+k)+1} \end{bmatrix},$$

(where $\mathbf{P}_0, \mathbf{P}_1, ..., \mathbf{P}_M$ are 2×2 matrices), for which $M = 1$ and

$$\begin{aligned} \mathbf{P}_0 &= \begin{bmatrix} \frac{-\phi_2(1/2)}{\phi_1(1/2)\phi_2(1)} & \frac{1}{\phi_1(1/2)} \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} \frac{3}{8\sqrt{6}} & \frac{10}{8\sqrt{6}} \\ 0 & 0 \end{bmatrix} \\ \mathbf{P}_1 &= \begin{bmatrix} \frac{-\phi_2(1/2)}{\phi_1(1/2)\phi_2(1)} & 0 \\ \frac{1}{\phi_2(1)} & 0 \end{bmatrix} = \begin{bmatrix} \frac{3}{8\sqrt{6}} & 0 \\ \frac{1}{\sqrt{3}} & 0 \end{bmatrix}. \end{aligned} \tag{5.3}$$

For CL, BiGHM and BiHermite systems, a single matrix preprocessor, preserving two approximation orders, can be constructed. The single matrix is given by

$$\mathbf{P}_0 = \begin{bmatrix} \frac{1}{4} & \frac{1}{4} \\ \frac{1}{b} & \frac{1}{-b} \end{bmatrix}. \tag{5.4}$$

where $b = 1 + \sqrt{7}$ for CL, $b = -6$ for BiGHM, $b = -3$ for BiHermite. One can easily check that in response to constant $X_k = 1$ and linear $X_k = k$ inputs, this preprocessor produces output

$$\mathbf{v}_{0,k} = \mathbf{P}_0 \begin{bmatrix} X_{2k} \\ X_{2k+1} \end{bmatrix},$$

such that after high pass filtering $\sum_m \mathbf{H}_{m-2k}\mathbf{v}_{0,m}$ the result is zero (disregarding end effects). Furthermore, it can be shown that for any multiwavelet system with more than one vanishing moment, one scaling function symmetric, the other antisymmetric and both of them supported on $[0, 2]$, such a matrix prefilter exists. Parameter b in $\mathbf{P}_0$ should be equal to twice the value of the derivative of the antisymmetric scaling function at $t = 1$. Zero-length ($M = 0$) prefilters are very desirable because they do not introduce correlation between pairs $\mathbf{v}_{0,k}$; such prefiltering will also be called 'approximation' here.

For general length prefilters, if we represent the matrix preprocessing by a single matrix multiplication, $\mathcal{P}\mathbf{X} = \mathbf{V}_0$, where now $\mathcal{P}$ is $N \times N$, $\mathbf{X}$ is $N \times 1$, and $\mathbf{V}_0$ is $N \times 1$, then in detail we have

$$\begin{bmatrix} \mathbf{P}_0 & \mathbf{P}_1 & \dots & \mathbf{P}_M & \mathbf{0}_2 & \dots & \dots & \dots \\ \mathbf{0}_2 & \mathbf{P}_0 & \mathbf{P}_1 & \dots & \mathbf{P}_M & \mathbf{0}_2 & \dots & \dots \\ & & & & & & & \ddots \end{bmatrix} \begin{bmatrix} X_0 \\ X_1 \\ X_2 \\ X_3 \\ \vdots \end{bmatrix} = \begin{bmatrix} v_{0,0}^{(0)} \\ v_{0,0}^{(1)} \\ v_{0,1}^{(0)} \\ v_{0,1}^{(1)} \\ \vdots \end{bmatrix},$$

where of course the $\mathbf{P}_m$ matrices are 2×2 and $\mathbf{0}_2$ is a 2×2 matrix of zeros.

5.3 Boundaries

Boundaries are handled by symmetric data extension for both critically sampled and oversampled schemes. As pointed out in Strela *et al.* (1999), symmetric extension — which is particularly well-matched to approximation preprocessing — can be implemented only with symmetric and/or antisymmetric filter banks, exactly the types under study here.

5.4 Energy preservation

Let us represent whichever wavelet transformation is chosen (scalar, multi, orthogonal or biorthogonal), by the matrix transformation $\mathcal{M}$. If repeated row preprocessing is used, $\mathcal{M}$ is $2N \times 2N$ while if critical sampling is applied, $\mathcal{M}$ is $N \times N$.

- If $\mathcal{M}$ is an orthonormal transform, then the energy in the sequence $\mathcal{P}\mathbf{X}$, (i.e., the sum of squares of the elements), is unchanged after application of the transform, $||\mathcal{M}\mathcal{P}\mathbf{X}||^2 = ||\mathcal{P}\mathbf{X}||^2$. In this case the only opportunity for a change in scale is the preprocessing, $||\mathbf{X}||^2 \to ||\mathcal{P}\mathbf{X}||^2$.
- If the transform is biorthogonal, (scalar or multiwavelet), then both the preprocessing and matrix transformation will change the input energy. See Section 6 for details.

These facts have important ramifications when studying the variance of stochastic sequences subjected to preprocessing and the DMWT, as will now be investigated.

6 Stochastic sequences and transform covariance

Suppose, as is often the case, the observed signal comprises true (deterministic) signal $\{D_k\}$ plus (stochastic) zero-mean noise, $\{\epsilon_k\}$, i.e., $X_k = D_k + \epsilon_k$, and

then the effect of preprocessing is given by $\mathcal{P}\mathbf{X} = \mathcal{P}\mathbf{D} + \mathcal{P}\boldsymbol{\epsilon}$. The preprocessing is followed by the DMWT in one of the manifestations discussed in Sections 3 and 4. Hence,

$$\mathcal{M}\mathcal{P}\mathbf{X} = \mathcal{M}\mathcal{P}\mathbf{D} + \mathcal{M}\mathcal{P}\boldsymbol{\epsilon}. \tag{6.1}$$

The stochastic part, $\mathcal{M}\mathcal{P}\boldsymbol{\epsilon}$, denoted $\mathbf{Z}$ for convenience, has variance-covariance matrix given by $\mathrm{cov}\{\mathbf{Z}\} = \mathcal{M}\mathcal{P}\boldsymbol{\Sigma}\mathcal{P}^T\mathcal{M}^T$, where $\boldsymbol{\Sigma}$ is the variance-covariance matrix of the noise $\{\epsilon_k\}$. Henceforth we let the input be white noise with mean zero and variance σ_ϵ^2, so that $\boldsymbol{\Sigma}$ is the identity matrix multiplied by σ_ϵ^2, and

$$\mathrm{cov}\{\mathbf{Z}\} = \mathcal{M}[\sigma_\epsilon^2\mathcal{P}\mathcal{P}^T]\mathcal{M}^T = \sigma_\epsilon^2\mathcal{M}\mathcal{P}\mathcal{P}^T\mathcal{M}^T. \tag{6.2}$$

Note that the matrix $\sigma_\epsilon^2\mathcal{P}\mathcal{P}^T$ is due to the input white noise and preprocessing; if this matrix was the identity, then an orthonormal DMWT would imply that $\mathrm{cov}\{\mathbf{Z}\}$ would be the identity, since for an orthonormal transform $\mathcal{M}\mathcal{M}^T$ would be equal to the identity matrix.

Apart from the scale factor σ_ϵ^2, the variance of the white noise, the covariance matrix $\mathrm{cov}\{\mathbf{Z}\}$ depends only on the form of preprocessing used, through $\mathcal{P}$, and the choice of wavelet transform, through $\mathcal{M}$. We have computed this covariance matrix for a number of useful cases, with σ_ϵ^2 set to unity. The covariance matrices are all 128×128. Rows numbered 65-128 hence correspond to level $j = 1$, those numbered 33-64 are for level $j = 2$, and 17-32 for level $j = 3$. For repeated row preprocessing N is 64, while for critical sampling $N = 128$.

6.1 GHM multiwavelet system

Let us look first at the GHM system using repeated row preprocessing. To understand the structure of the $\sigma_\epsilon^2\mathcal{P}\mathcal{P}^T$ matrix, consider first the case $N = 2$. Then the covariance matrix of $[v_{0,0}^{(0)}, v_{0,0}^{(1)}, v_{0,1}^{(0)}, v_{0,1}^{(1)}]^T$ is given by $\sigma_\epsilon^2\mathcal{P}\mathcal{P}^T$ i.e., by

$$\sigma_\epsilon^2 \begin{bmatrix} 1 & 0 \\ \alpha & 0 \\ 0 & 1 \\ 0 & \alpha \end{bmatrix} \begin{bmatrix} 1 & \alpha & 0 & 0 \\ 0 & 0 & 1 & \alpha \end{bmatrix} = \sigma_\epsilon^2 \begin{bmatrix} 1 & \alpha & 0 & 0 \\ \alpha & \alpha^2 & 0 & 0 \\ 0 & 0 & 1 & \alpha \\ 0 & 0 & \alpha & \alpha^2 \end{bmatrix}.$$

With $\alpha = 1/\sqrt{2}$, we see that the variances on the main diagonal are alternately σ_ϵ^2 and $\sigma_\epsilon^2/2$, i.e., an 'average' variance of $0.75\sigma_\epsilon^2$, while the off-diagonal terms are each $\sigma_\epsilon^2/\sqrt{2}$. Such a covariance matrix is far from the identity matrix multiplied by σ_ϵ^2, so that even though the GHM DMWT is orthogonal, the covariance matrix of the result, $\mathrm{cov}\{\mathbf{Z}\}$ will be far from a scaled identity matrix. Fig. 2 shows the covariance $\mathrm{cov}\{\mathbf{Z}\}$ for the GHM system using repeated row preprocessing. Looking first at the variances along the diagonal in Fig. 2, we note these increase with level j. Fig. 3 shows various rows through the matrix; we see that at level 1, row 120, that there is significant covariance just off the

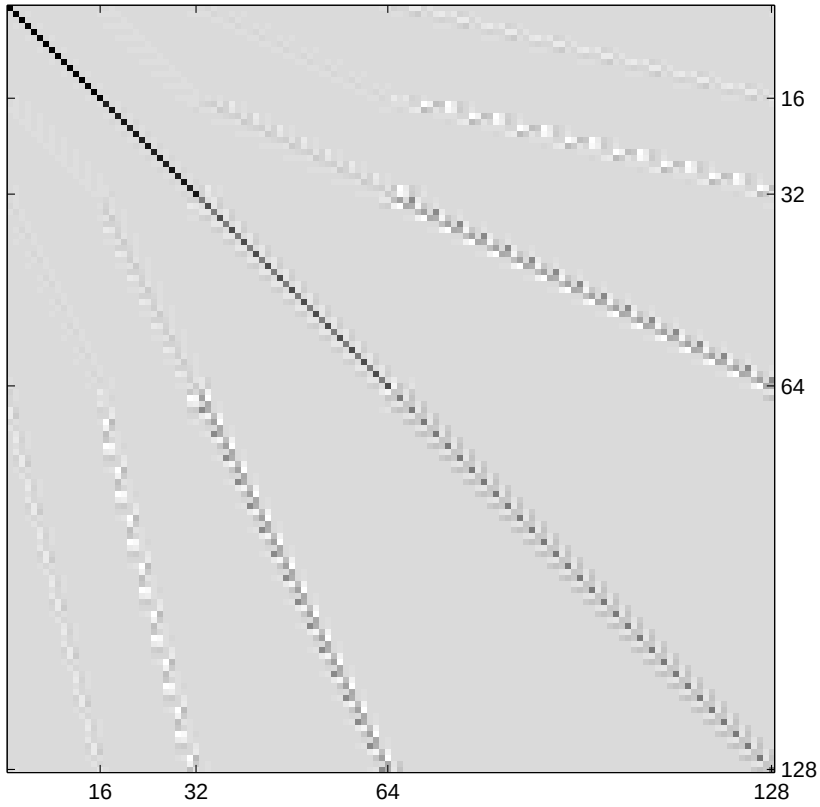


Figure 2: Grey-scale covariance matrix $\mathrm{cov}\{\mathbf{Z}\}$ of size 128×128 for the GHM system using repeated row preprocessing with $\sigma_\epsilon^2 = 1$. Positive values are darker than the background, negative values are lighter.

main diagonal, but on the adjacent line 119 this is markedly different due to the alternating variances on the inputs to the GHM DMWT. Moving through the levels, we see that the covariance just off the main diagonal essentially disappears; see for example the main diagonal spikes on lines 29 and 30 ($j = 3$.) There is notable covariance between all scales.

If we let c denote a variance deflation, or inflation factor, by which σ_ϵ^2 is changed to an average variance $c\sigma_\epsilon^2$, then we see that for GHM repeated row preprocessing $c = 0.75$. Since the GHM DMWT is orthonormal, it will not change the scale of the stochastic input. If we add the variances of the coefficients at all levels in Fig. 2, and divide by the total number of coefficients, we obtain an 'average' variance deflation factor of $c = 0.75$, as expected.

By way of contrast Fig. 4 shows the covariance for the GHM system using approximation preprocessing. In implementing approximation preprocessing we have rescaled the elements of the matrix prefilters in (5.3) by multiplying

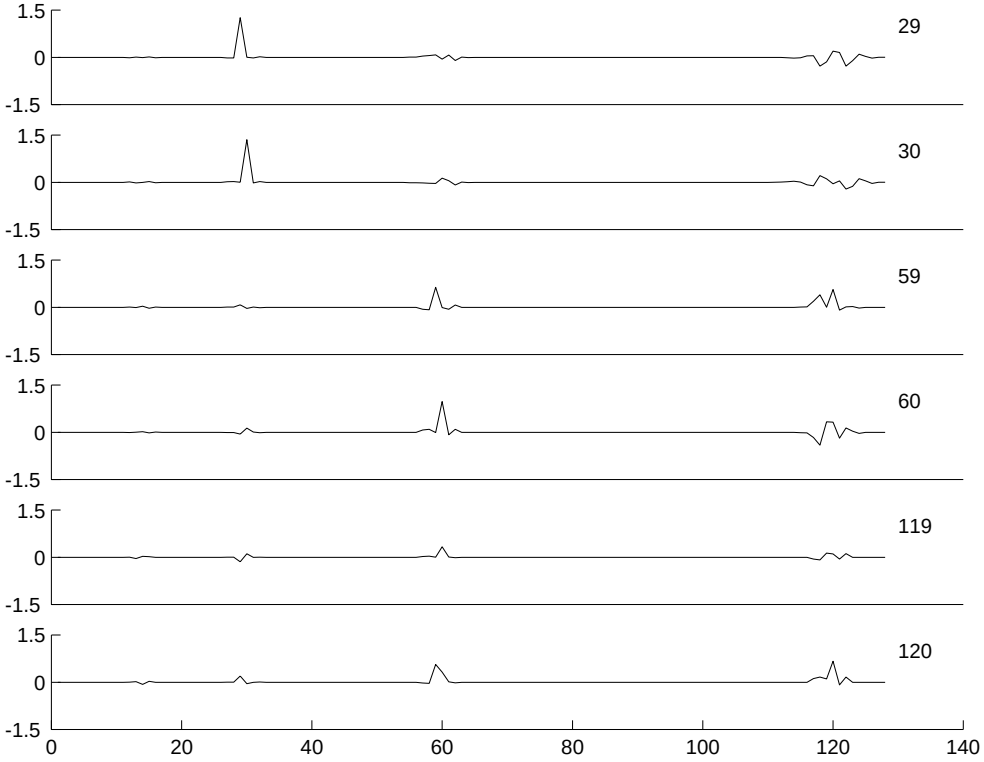


Figure 3: Various rows through the matrix.

all elements by $\sqrt{3}$. Then, (5.2) and (5.1) become

$$v_{0,k}^{(0)} = \frac{3}{8\sqrt{2}} X_{2k} + \frac{10}{8\sqrt{2}} X_{2k+1} + \frac{3}{8\sqrt{2}} X_{2k+2} \text{ and } v_{0,k}^{(1)} = X_{2k+2}.$$

It then follows for example that

$$\begin{aligned} \text{var}\{v_{0,0}^{(0)}\} &= (118/128)\sigma_\epsilon^2 \approx 0.922\sigma_\epsilon^2 \text{ and } \text{var}\{v_{0,0}^{(1)}\} = \sigma_\epsilon^2, \\ \text{cov}\{v_{0,0}^{(0)}, v_{0,0}^{(1)}\} &= \frac{3}{8\sqrt{2}}\sigma_\epsilon^2 \approx 0.265\sigma_\epsilon^2 = \text{cov}\{v_{0,0}^{(1)}, v_{0,1}^{(0)}\}, \\ \text{cov}\{v_{0,0}^{(0)}, v_{0,1}^{(0)}\} &= \frac{9}{128}\sigma_\epsilon^2 \approx 0.070\sigma_\epsilon^2, \end{aligned}$$

etc, so that for $N = 4$ the covariance matrix of $[v_{0,0}^{(0)}, v_{0,0}^{(1)}, v_{0,1}^{(0)}, v_{0,1}^{(1)}]^T$ is given approximately by $\sigma_\epsilon^2 \mathcal{P}\mathcal{P}^T$ i.e., by

$$\sigma_\epsilon^2 \begin{bmatrix} 0.922 & 0.265 & 0.070 & 0 \\ 0.265 & 1 & 0.265 & 0 \\ 0.070 & 0.265 & 0.922 & 0.265 \\ 0 & 0 & 0.265 & 1.0 \end{bmatrix}.$$

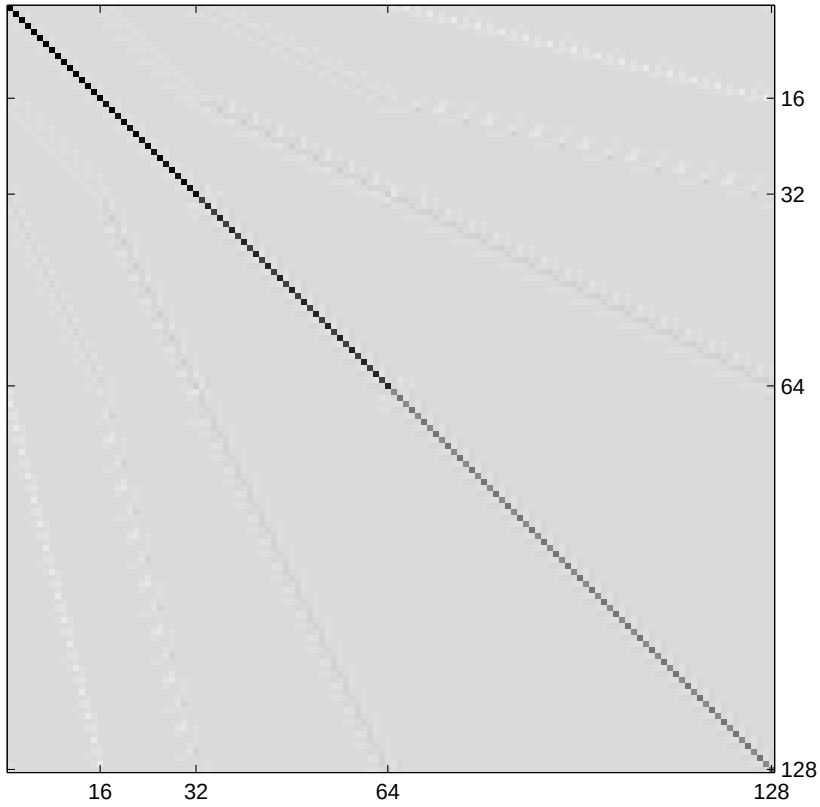


Figure 4: Grey-scale covariance matrix cov$\{\mathbf{Z}\}$ of size 128×128 for the GHM system using approximation preprocessing with $\sigma_\epsilon^2 = 1$.

The variances on the main diagonal are alternately $0.922\sigma_\epsilon^2$ and σ_ϵ^2, i.e., an 'average' variance of $0.961\sigma_\epsilon^2$, while the off-diagonal terms are relatively small; such a covariance matrix is not that far from a scaled identity matrix, so the covariance matrix after the GHM DMWT, cov$\{\mathbf{Z}\}$ will not be too far from one either. Referring to Fig. 4 we see that the variances along the diagonal again increase with level, but there is virtually zero covariance just off the main diagonal, and the covariances between scales are very small. If we add the variances of the coefficients at all levels in Fig. 4, and divide by the total number of coefficients, we obtain the expected 'average' variance deflation factor of $c = 0.961$.

6.2 CL multiwavelet system

Let us look now at the CL system, firstly using repeated row preprocessing. Considering the case $N = 2$ the covariance matrix of $[v_{0,0}^{(0)}, v_{0,0}^{(1)}, v_{0,1}^{(0)}, v_{0,1}^{(1)}]^T$ is

given by $\sigma_\epsilon^2 \mathcal{P}\mathcal{P}^T$ i.e., by

$$\sigma_\epsilon^2 \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{6.3}$$

While there are no off-diagonal terms, the variances on the main diagonal are alternately σ_ϵ^2 and zero, i.e., an 'average' variance of $0.5\sigma_\epsilon^2$. Such a covariance matrix is far from a scaled identity matrix. Fig. 5 shows the covariance cov$\{\mathbf{Z}\}$

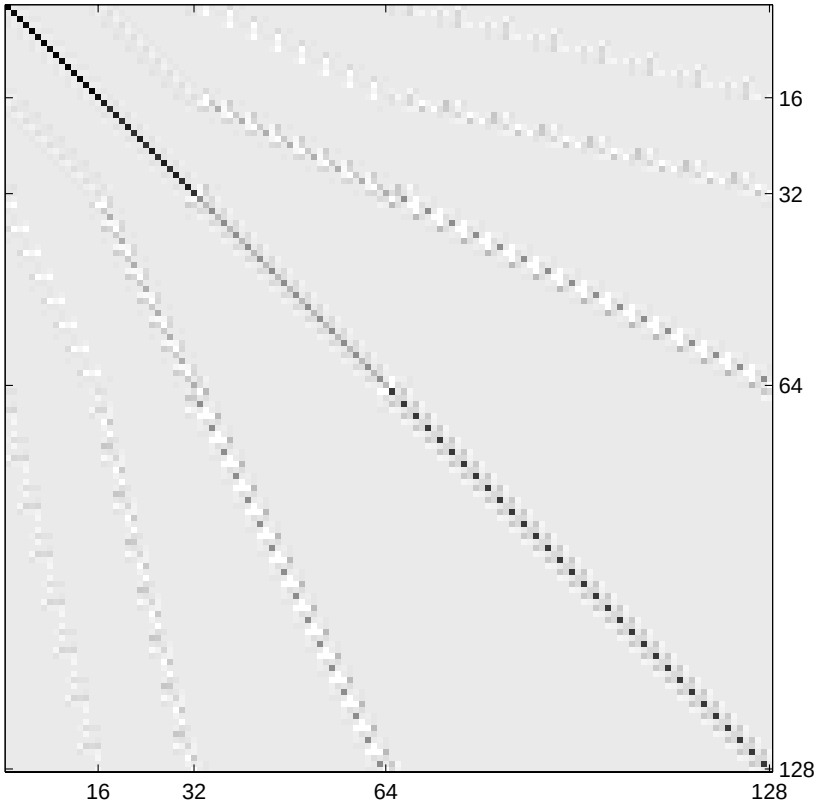


Figure 5: Grey-scale covariance matrix cov$\{\mathbf{Z}\}$ of size 128×128 for the CL system using repeated row preprocessing with $\sigma_\epsilon^2 = 1$.

for the CL system using repeated row preprocessing. Fig. 6 shows various rows through the matrix; we see that at level 1, row 119, that there is significant covariance just off the main diagonal, but on the adjacent line 120 there is little variance on the main diagonal due to the alternating variances on the inputs to the CL DMWT. With increasing level the covariance just off the main diagonal essentially disappears; see for example the main diagonal spikes on lines 29 and 30. There is notable covariance between all scales. One notable feature in Fig. 5 is the apparent sharp decrease in variance at level 2. This is due to the alternately low and high diagonal variance values at level 1, being thoroughly

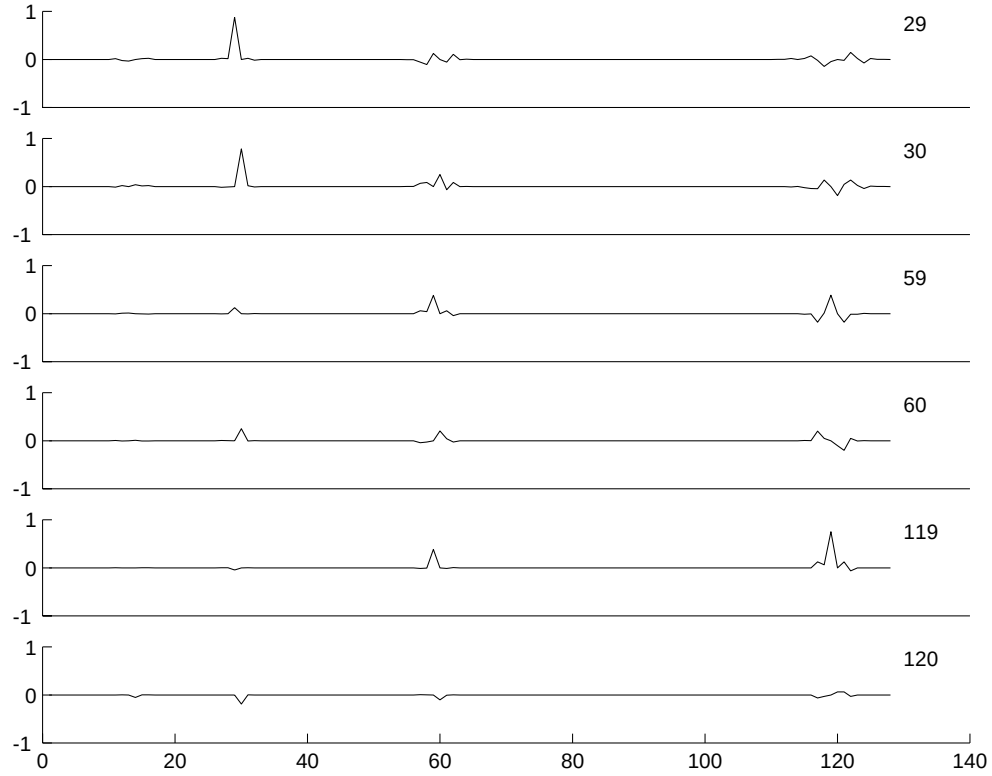


Figure 6: Various rows through the matrix.

redistributed by the multifiltering which gives the level 2 coefficients.

If we add the variances of the coefficients at all levels in Fig. 5 and divide by the total number of coefficients, we obtain the expected 'average' variance deflation factor of $c = 0.5$.

For CL with preprocessing using the matrix prefilter given by (5.4) with $b = 1 + \sqrt{7}$,

$$v_{0,k}^{(0)} = \frac{1}{4}X_{2k} + \frac{1}{4}X_{2k+1} \text{ and } v_{0,k}^{(1)} = \frac{1}{1+\sqrt{7}}X_{2k} - \frac{1}{1+\sqrt{7}}X_{2k+1}.$$

It then follows for example that

$$\text{var}\{v_{0,k}^{(0)}\} = (2/16)\sigma_\epsilon^2 = 0.125\sigma_\epsilon^2 \text{ and } \text{var}\{v_{0,k}^{(1)}\} = [2/(1+\sqrt{7})^2]\sigma_\epsilon^2 \approx 0.15\sigma_\epsilon^2.$$

Since $\mathcal{P}\mathcal{P}^T$ is diagonal, all covariances are null, so that for $N = 4$ for example the covariance matrix of $[v_{0,0}^{(0)}, v_{0,0}^{(1)}, v_{0,1}^{(0)}, v_{0,1}^{(1)}]^T$ is given by $\sigma_\epsilon^2 \mathcal{P}\mathcal{P}^T$ i.e., approximately by

$$\sigma_\epsilon^2 \begin{bmatrix} 0.125 & 0 & 0 & 0 \\ 0 & 0.15 & 0 & 0 \\ 0 & 0 & 0.125 & 0 \\ 0 & 0 & 0 & 0.15 \end{bmatrix}.$$

The average variance is 0.138. The alternating-size diagonal (variance) el-

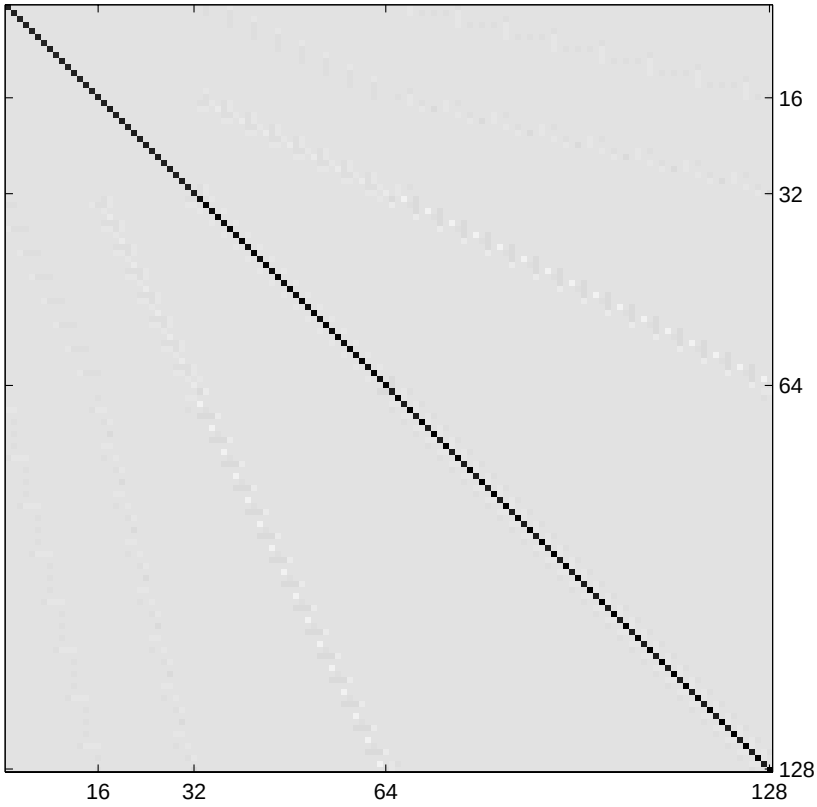


Figure 7: Grey-scale covariance matrix cov$\{\mathbf{Z}\}$ of size 128×128 for the CL system using approximation preprocessing with $\sigma_\epsilon^2 = 1$.

ements are the only manifestation of a non-white noise as input to the CL DMWT. Following the CL DMWT the covariance matrix of Fig. 7 is obtained. As would be anticipated, this is close to a scaled identity matrix. At level 1 the alternating diagonal elements have been changed to 0.131 and 0.149, while at level 3 they are almost equal at 0.128 and 0.131.

If we add the variances of the coefficients at all levels in Fig. 7, and divide by the total number of coefficients, we obtain the expected 'average' variance deflation factor of $c = 0.138$.

6.3 Biorthogonal GHM and Hermite cubics multiwavelet system

The covariance matrix cov$\{\mathbf{Z}\}$ for the BiGHM system using repeated row preprocessing with $\alpha = 0$ was computed. The matrix $\sigma_\epsilon^2 \mathcal{P}\mathcal{P}^T$ is as for the CL system (e.g., equation (6.3)). The variances on the main diagonal again alternate in size. For example, at level 1 they are 0.16 and 3, and at level 3

they are 5.19 and 15.31.

For BiGHM with preprocessing using the matrix prefilter given by (5.4) with $b = -6$, $\mathcal{P}\mathcal{P}^T$ is diagonal and all covariances are null. The variances on the main diagonal of $\text{cov}\{\mathbf{Z}\}$ again alternate in size. For example, at level 1 they are 0.24 and 0.88, and at level 3 they are 1.02 and 2.84.

The covariance matrix $\text{cov}\{\mathbf{Z}\}$ for the BiHermite system using repeated row preprocessing with $\alpha = 0$ was also computed. At level 1 the alternating variances are 0.19 and 0.016, and at level 3 they are 1.81 and 1.63.

For BiHermite with preprocessing using the matrix prefilter given by (5.4) with $b = -3$, $\mathcal{P}\mathcal{P}^T$ is diagonal and all covariances are null. The variances on the main diagonal of $\text{cov}\{\mathbf{Z}\}$ again alternate in size; at level 1 they are 0.055 and 0.033, and at level 3 they are 0.32 and 0.29.

These biorthogonal transforms are not orthonormal. Table 1 shows the ratio of output variance to input variance for the above biorthogonal transforms. The results are an average over 100 different input Gaussian white noise sequences of the lengths given, and are related to the maximum levels to which the transforms are taken. (Note that a given N means approximation preprocessing methods would start with two vectors of length $N/2$, and repeated row preprocessing would start with two vectors of length N. Also if the maximum level $J = j_0$, then j_0 steps of the cascade algorithm were used for repeated row methods and Bi9-7, and $j_0 - 1$ steps of the cascade were used for approximation methods, in order that the number of untouched scaling coefficients is the same.) For any particular transform, we see that the variance inflation/deflation factors are almost independent of sample size, but depend strongly on the maximum level to which the transform is taken.

6.4 Scalar Bi9-7 transform

The covariance matrix for the biorthogonal 9-7 transform is not far from the identity, meaning that this transform is not far from orthonormal. This is perhaps not too surprising in view of the discussion on "Biorthogonal bases close to an orthonormal basis" in Daubechies (1992, Section 8.3.5). Table 1 shows that 1.026 is a good overall variance inflation factor for different sample sizes and maximum levels of the transform; hence, as expected, scale is nearly preserved ($c \approx 1$).

6.5 D4 and LA8 scalar transforms

Both these transforms are orthonormal, so that the covariance matrix is given simply by $\text{cov}\{\mathbf{Z}\} = \sigma_\epsilon^2 \mathbf{I}_N$.

N	max. level J	BiGHMr	BiGHMa	BiHr	BiHa	Bi9-7
256	5	4.059	0.960	0.805	0.188	1.032
	4	3.609	0.842	0.935	0.220	1.032
	3	2.704	0.636	1.178	0.271	1.029
	2	1.537	0.346	1.294	0.284	1.024
512	6	4.432	1.045	0.737	0.170	1.034
	5	4.105	0.975	0.806	0.189	1.029
	4	3.617	0.849	0.957	0.223	1.028
	3	2.711	0.633	1.171	0.271	1.026
	2	1.543	0.350	1.312	0.293	1.022
1024	7	4.579	1.089	0.696	0.161	1.029
	6	4.440	1.054	0.739	0.172	1.028
	5	4.211	0.994	0.811	0.189	1.027
	4	3.631	0.851	0.943	0.220	1.026
	3	2.710	0.631	1.195	0.276	1.024
	2	1.540	0.350	1.325	0.293	1.020
2048	8	4.709	1.113	0.686	0.158	1.028
	7	4.627	1.102	0.709	0.164	1.027
	6	4.458	1.057	0.745	0.173	1.027
	5	4.172	0.987	0.815	0.190	1.026
	4	3.626	0.852	0.958	0.225	1.025
	3	2.729	0.637	1.192	0.277	1.023
	2	1.540	0.350	1.313	0.291	1.020
4096	9	4.723	1.124	0.674	0.155	1.027
	8	4.721	1.122	0.687	0.158	1.026
	7	4.622	1.095	0.703	0.162	1.026
	6	4.462	1.056	0.745	0.173	1.027
	5	4.171	0.991	0.818	0.192	1.025
	4	3.638	0.855	0.967	0.226	1.025
	3	2.726	0.635	1.178	0.271	1.023
	2	1.539	0.348	1.319	0.292	1.020

Table 1: Variance inflation/deflation factors for biorthogonal transforms.

7 Thresholding

Starting with a signal $X_0, ..., X_{N-1}$ and referring to Fig. 1, we recall that with repeated row preprocessing $N' = N$ vectors $\mathbf{v}_{0,k}$ enter the pyramid algorithm, while with critically-sampled preprocessing (such as approximation preprocessing) $N' = N/2$ vectors $\mathbf{v}_{0,k}$ enter the pyramid algorithm. After J steps of the pyramid algorithm of Fig. 1 we can write the wavelet coefficients in (6.1)

in element-by-element form as

$$\mathbf{w}_{j,k} = \mathbf{s}_{j,k} + \mathbf{z}_{j,k}, \qquad j = 1, ..., J; k = 0, ..., N'_j - 1, \tag{7.1}$$

where $N'_j = N'/2^j$, $\{\mathbf{w}_{j,k}\}$ are the coefficients corresponding to the transformed observations $\mathcal{MP}\mathbf{X}$, $\{\mathbf{s}_{j,k}\}$ are the coefficients corresponding to the transformed deterministic signal $\mathcal{MP}\mathbf{D}$, and $\{\mathbf{z}_{j,k}\}$ are the coefficients corresponding to the transformed stochastic noise $\mathcal{MP}\boldsymbol{\epsilon}$. There are also scaling coefficients $\mathbf{v}_{J,k},\ k = 0, ..., N'_J - 1$. The idea of thresholding is to produce a denoised estimate using the following steps:

1. Apply preprocessing and the DMWT in one of the forms discussed out to level J.
2. Apply a thresholding procedure to the resulting empirical wavelet coefficients $\mathbf{w}_{j,k}, j = 1, ..., J; k = 0, ..., N'_j - 1$, (leaving the remaining empirical scaling coefficients $\mathbf{v}_{J,k},\ k = 0, ..., N'_J - 1$ entirely alone as these carry low-frequency 'smooth' information).
3. Invert the DMWT and apply the post-processor, producing a denoised estimate $\{\hat{X}_0, ..., \hat{X}_{N-1}\}$ of the input signal.

The thresholding step can be carried out as a scalar or vector operation, as discussed below.

7.1 Scalar thresholding

Suppose that we composite together the empirical wavelet coefficient vectors $\mathbf{w}_{j,k}, j = 1, ..., J; k = 0, ..., N'_j - 1$, and treat them as one scalar sequence of coefficients,

$$w^{(0)}_{1,0}, w^{(1)}_{1,0}, ..., w^{(0)}_{1,N'_1-1}, w^{(1)}_{1,N'_1-1}, ..., w^{(0)}_{J,0}, w^{(1)}_{J,0}, ..., w^{(0)}_{J,N'_J-1}, w^{(1)}_{J,N'_J-1}.$$

We know that *if* both preprocessing and transform were orthonormal, then white noise $\{\epsilon_k\}$ with mean zero and variance σ^2_ϵ would be transformed to another scalar sequence of white noise coefficients with mean zero and variance σ^2_ϵ.

The idea of universal thresholding for such scalar sequences of white noise wavelet coefficients was introduced by Donoho and Johnstone (1994). The basic idea is that if the signal component is in fact zero, then with high probability the combination of (zero) signal plus noise should not exceed the threshold level. Suppose $\epsilon_1, \ldots, \epsilon_M$ is a sequence of independent and identically distributed Gaussian (normal) random variables with mean zero and variance σ^2_ϵ, then as $M \to \infty$,

$$P_M \equiv Pr\left[\max_i |\epsilon_i| > \sqrt{(2\sigma^2_\epsilon \log M)}\right] \to 0.$$

This means that, asymptotically, if the signal component is in fact zero, then the probability of a 'false alarm' will tend to zero, so that the combination of (zero) signal plus noise will not exceed the threshold level, and will hence be set to zero. The scalar threshold is thus chosen as $T = T_s = \sqrt{(2\sigma_\epsilon^2 \log M)}$. The wavelet coefficients $w_{j,k}^{(i)}$ are thresholded using

$$\hat{w}_{j,k}^{(i)} = \delta(w_{j,k}^{(i)}, T_s), \;\; i = 0, 1; 1 \leq j \leq J; k = 0, ..., N_j' - 1.$$

where $\delta(\cdot, \cdot)$ could be the hard-threshold rule:

$$\delta_H(w_{j,k}^{(i)}, T_s) = w_{j,k}^{(i)} \, 1\{|w_{j,k}^{(i)}| > T_s\}.$$

The inverse DMWT is then applied to the $\{\hat{\mathbf{w}}_{j,k}\}$ followed by the post-processor.

The universal threshold is attractively simple, but is strictly suitable only when thresholding a Gaussian scalar sequence of white noise wavelet coefficients of mean zero and variance σ_ϵ^2. While the scalar DWT will not change the variance of input noise, we have seen that following preprocessing and the DMWT, the variances of the wavelet coefficients $w_{j,k}^{(0)}$ and $w_{j,k}^{(1)}$ can differ substantially, and we have defined 'average variances;' for example for GHM with repeated row preprocessing the noise variance σ_ϵ^2 is deflated by a factor $c = 0.75$ to $0.75\sigma_\epsilon^2$. Not only this, but for repeated row preprocessing $N' = 2N$. Hence, for GHM with repeated row preprocessing, and ignoring covariances, $T_s = \sqrt{(2 \times 0.75\sigma_\epsilon^2 \log 2N)}$. This contrasts with for example GHM with approximation preprocessing, which is critically sampled, and for which the noise variance is essentially preserved ($0.96\sigma_\epsilon^2$), so that $T_s = \sqrt{(2\sigma_\epsilon^2 \log N)}$. Similarly, for the CL method with repeated row preprocessing, we used $T_s = \sqrt{(2 \times 0.5\sigma_\epsilon^2 \log 2N)}$, and for the CL method with matrix (approximation) preprocessing we used $T_s = \sqrt{(2 \times 0.138\sigma_\epsilon^2 \log N}$; see Table 2.

We need to see how correlation in the wavelet coefficients due to non-orthogonal preprocessing damages the universal thresholding approach to denoising.

7.2 Vector thresholding

If the noise $\{\epsilon_k\}$ in (6.1) comes from zero-mean white Gaussian noise, then $\mathbf{z}_{j,k}$ in (7.1) has a multivariate Gaussian distribution with mean zero and covariance matrix given by the appropriate 2×2 block of the covariance matrix $\text{cov}\{\mathbf{Z}\} = \sigma_\epsilon^2 \mathcal{M}\mathcal{P}\mathcal{P}^T\mathcal{M}^T$. For $j = 1$ and referring to Fig. 2 in terms of row and column numbers, such 2×2 blocks are given by, for example, the following elements: $(128, 128), (128, 127)$, $(127, 127), (127, 128)$ and $(126, 126), (126, 125)$, $(125, 125), (125, 126)$. At any level j all such 2×2 blocks are the same and hence we can write the 2×2 covariance matrix as $\boldsymbol{\Lambda}_j$ so that

$$\mathbf{z}_{j,k} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Lambda}_j).$$

no.	method	preprocessing	N'	c
1.	GHM	repeated row ($\alpha = 1/\sqrt{2}$)	$2N$	0.750
2.	GHM	approximation	N	0.961
3.	CL	repeated row ($\alpha = 0$)	$2N$	0.500
4.	CL	matrix (5.4), with $b = (1 + \sqrt{7})$	N	0.138
5.	Bi9-7*	none (scalar)	N	1.026
6.	D4	none (scalar)	N	1.0
7.	LA8	none (scalar)	N	1.0

Table 2: Appropriate values of N' and c to use in the scalar threshold $T_s = \sqrt{(2\,c\,\sigma_\epsilon^2 \log N')}$ for orthonormal multiwavelet transforms, a scalar biorthogonal (nearly orthogonal) transform, and scalar orthonormal wavelet transforms. *Note that for this non-orthogonal transform c will vary slightly with transform level and with N.

In the absence of a signal component, Downie and Silverman (1998) pointed out that $\theta_{j,k} = \mathbf{w}_{j,k}^T \boldsymbol{\Lambda}_j^{-1} \mathbf{w}_{j,k}$ has a χ_2^2 distribution. The vector wavelet coefficients $\mathbf{w}_{j,k}$ could be thresholded using

$$\hat{\mathbf{w}}_{j,k} = \delta(\mathbf{w}_{j,k}, T_v),\ 1 \le j \le J; k = 0, ..., N_j' - 1,$$

where $\delta(\cdot, \cdot)$ could be the hard-threshold rule:

$$\delta_H(\mathbf{w}_{j,k}, T_v) = \mathbf{w}_{j,k}\, 1\{\theta_{j,k} > T_v\}.$$

A suitable vector threshold level T_v is required. The inverse DMWT is then applied to the $\{\hat{\mathbf{w}}_{j,k}\}$ followed by the post-processor.

If $\eta_1, \ldots, \eta_M$ is a sequence of independent and identically distributed χ_2^2 random variables, then (e.g., Galambos, 1978) as $M \to \infty$

$$P_M \equiv Pr\left[\max_i \eta_i > 2 \log M\right] \to 0.$$

Hence the vector threshold is chosen as $T_v = 2 \log M$, as used by Downie and Silverman (1998). It is clear from our covariance matrices that the wavelet coefficients are generally correlated, to a greater or lesser extent, between scales, but this is not taken into account here. However, the vector threshold is at least scale-free, and need only be changed from $T_v = 2 \log 2N$, for repeated row preprocessing to $T_v = 2 \log N$, for critical sampling.

7.3 Numerical simulations

For the reasons discussed above neither the scalar nor the vector thresholding methods are ideally matched to the properties of multiwavelet coefficients,

although vector thresholding does at least build-in the covariance structure of the $\mathbf{z}_{j,k}$ vectors. Also, multiwavelet theory was largely developed with compression properties in mind, rather than multiwavelet thresholding. In this section we discuss simulation results in which methods 1-7 listed in Table 2 were applied. For scalar thresholding T_s used values of N' and c as given in Table 2, while for vector thresholding T_v was set to $2\log 2N$ for repeated row preprocessing and to $2\log N$ for critical sampling. (For the biorthogonal multiwavelets methods the value of c depends on J and to a lesser extent, N, as in Table 1. This is less convenient in practice than using methods 1-7 of Table 2. In thresholding experiments, biorthogonal multiwavelets performed quite well, but for reasons of brevity details are not reported here.)

Four types of true (deterministic) signal were utilised: (i) the mixed signal used in Strela *et al* (1999), and (ii) 'bumps,' (iii) 'blocks,' and (iv) 'Heavisine,' these latter three being standard test signals since their specification in Donoho and Johnstone (1994). The 'bumps' signal was also used by Downie and Silverman (1998). White Gaussian noise was added to create different noisy realisations. With the same definition of signal-to-noise (S/N) ratio as in Donoho and Johnstone (1994), i.e., the ratio of signal to noise standard deviations, we looked at S/N ratios of 2, 4 and 8 (variance ratios of 6, 12 and 18dB, respectively). For each signal type (i)-(iv) and for each S/N level, the following was done:

1. 100 realizations of the noisy signal were constructed. For (i) $N = 512$, and for (ii)-(iv), $N = 2048$, in line with previous studies.
2. Each realization was subjected in turn to each of the methods 1-7 in Table 2; for multiwavelet methods this consists of preprocessing, DMWT, hard thresholding, inverse DMWT, postprocessing, and for scalar methods this consists of DWT, hard thresholding, inverse DWT. For (i) $J = 4$ for repeated row preprocessing, and $J = 3$ for critical sampling; for (ii)-(iv) $J = 6$ for repeated row preprocessing and $J = 5$ for critical sampling. This leaves 32 pairs of coefficients untouched by thresholding in all cases, agreeing with Downie and Silverman (1998).
3. The sample root mean squared error (rmse) was computed for each realization and each method. This rmse is defined as $\sqrt{[\frac{1}{N}\sum_{k=0}^{N-1}(X_t - \hat{X}_t)^2]}$. It was also recorded whether or not the vector or scalar thresholding produced the smaller rmse for each realization and each multiwavelet method.

The average (over realizations) rmse results are shown in Fig. 8.

For the mixed signal (i), we see that for S/N=2 the best result is achieved using CL repeated row preprocessing with scalar thresholding; for S/N=4 the best result is achieved using GHM repeated row preprocessing with scalar

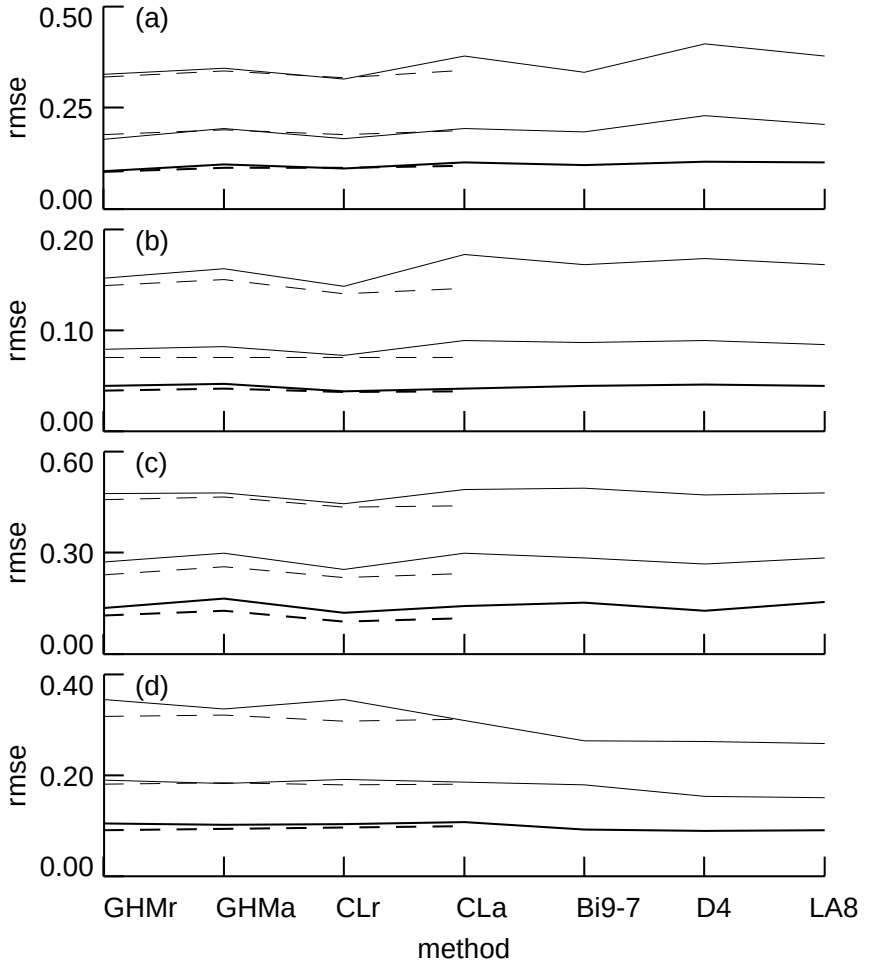


Figure 8: Average (over 100 realizations) rmse results for methods 1-7, for (a) mixed signal, (b) bumps, (c) blocks, and (d) Heavisine. Vector thresholding results are shown as dashed lines; signal-to-noise ratios are 8 (thick lines), 4 (medium) and 2 (thin).

thresholding, while for S/N=8 the best result corresponds again to GHM repeated row preprocessing, but with both scalar and vector thresholding producing essentially the same rmse.

For the 'bumps' signal (ii), with S/N=2 the best result is achieved using CL repeated row preprocessing with vector thresholding; for S/N=4 all the vector thresholding methods perform virtually identically, while for S/N=8 the best result corresponds to CL repeated row preprocessing, with vector thresholding. Note that for the first three methods the rmse differences between scalar and vector thresholding are quite small, and again the multiwavelet approach appears to offer most advantage over the scalar wavelet approach especially for lower S/N. The 'bumps' signal with S/N=4 was also analyzed by Downie

	gain (dB)							
S/N	Mixed	Bumps	Blocks	H'sine	Lenna	F'print	Boat	Bear
2	5.4	8.0	8.8	12.9	9.9	10.3	9.0	6.3
4	4.7	7.6	7.8	12.1	8.0	8.6	6.7	4.3
8	3.5	7.1	8.0	11.4	5.7	6.4	4.3	3.9

Table 3: Processing gains (dB) for different signals/images and S/N ratios. CL method with repeated row preprocessing and scalar thresholding used throughout. For image details see the next section.

and Silverman (1998, Table 2). Our rmse results agree with theirs for vector thresholding, but for scalar thresholding using GHM with repeated row preprocessing and with approximation preprocessing we obtained 0.0815 and 0.0840, respectively, compared to their 0.113 and 0.164. We believe our improved results arise due to our careful formulation of the form of the universal threshold.

For the 'blocks' signal (iii), the best result is achieved for all three S/N ratios using CL repeated row preprocessing, with vector thresholding.

For the Heavisine signal (iv), scalar wavelets give the best results especially for low S/N levels, with LA(8) best for S/N=2 and 4. All the scalar methods performed approximately equally well for S/N=8.

We thus see that multiwavelets outperform scalar wavelets for all models except Heavisine and vector thresholding does not always outperform scalar thresholding. Generally repeated row preprocessing gives better results than matrix preprocessing. The average rmse of the (output) noise in the reconstructed signals across methods, for any S/N ratio, varies from about 1/4 to 2/3 (depending on the model) of the actual standard deviation of the (input) simulated noise; the smallest reduction in this intrinsic noise occurs for the mixed signal, and the largest for the Heavisine. Hence intrinsic noise is reduced by practically useful amounts. CL repeated row preprocessing seems a good general method; only for the Heavisine signal and S/N=2 does it perform significantly worse than a competing method. Table 3 gives the processing gains for this method against S/N ratio and signal type, where processing gain is defined as

$$\text{gain} = 20 \log_{10} \left(\frac{\text{standard deviation of input noise}}{\text{average rmse in reconstruction}} \right) \text{dB}.$$

8 Two-dimensional signals

When dealing with two-dimensional signals (images) we use the direct products of the scaling functions and wavelets. From the algorithmic point of view,

the procedure is similar to the one-dimensional case discussed above.

1. First we preprocess all rows, then all columns (of the previously row-preprocessed data). The amount of data stays unchanged if the matrix (critical sampling) preprocessing is used, but is quadrupled if repeated row preprocessing is used.
2. Next we perform the 2D wavelet cascade. During each step a multiwavelet filter bank (Fig. 1) is applied first to the rows, and then to the columns, of the low-low output from the previous step.

8.1 Algorithm illustrated

Let us describe this procedure in terms of matrix elements. We assume critical sampling preprocessing for ease of description, but the structure for repeated row preprocessing follows analogously. We start with an $N \times N$ matrix of elements

$$\begin{bmatrix} X_{0,0} & \dots & X_{0,N-1} \\ \vdots & \ddots & \vdots \\ X_{N-1,0} & \dots & X_{N-1,N-1} \end{bmatrix}.$$

After horizontal critical preprocessing the matrix is converted to

$$\begin{bmatrix} u^{(0)}_{0,0} & \dots & u^{(0)}_{0,\frac{N}{2}-1} & u^{(1)}_{0,0} & \dots & u^{(1)}_{0,\frac{N}{2}-1} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ u^{(0)}_{N-1,0} & \dots & u^{(0)}_{N-1,\frac{N}{2}-1} & u^{(1)}_{N-1,0} & \dots & u^{(1)}_{N-1,\frac{N}{2}-1} \end{bmatrix},$$

where the vector of second components follows the vector of first components in each row. Next, vertical critical preprocessing is carried out to obtain

$$\mathbf{U} = \begin{bmatrix} u^{(0,0)}_{0,0} & \dots & u^{(0,0)}_{0,\frac{N}{2}-1} & u^{(0,1)}_{0,0} & \dots & u^{(0,1)}_{0,\frac{N}{2}-1} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ u^{(0,0)}_{\frac{N}{2}-1,0} & \dots & u^{(0,0)}_{\frac{N}{2}-1,\frac{N}{2}-1} & u^{(0,1)}_{\frac{N}{2}-1,0} & \dots & u^{(0,1)}_{\frac{N}{2}-1,\frac{N}{2}-1} \\ u^{(1,0)}_{0,0} & \dots & u^{(1,0)}_{0,\frac{N}{2}-1} & u^{(1,1)}_{0,0} & \dots & u^{(1,1)}_{0,\frac{N}{2}-1} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ u^{(1,0)}_{\frac{N}{2}-1,0} & \dots & u^{(1,0)}_{\frac{N}{2}-1,\frac{N}{2}-1} & u^{(1,1)}_{\frac{N}{2}-1,0} & \dots & u^{(1,1)}_{\frac{N}{2}-1,\frac{N}{2}-1} \end{bmatrix}.$$

The vector of second components follows the vector of first components in each column. For example the superscript $(0,1)$ indicates the first vector component is in the vertical direction and the second vector component is in the horizontal direction. This ends step 1 above.

Next we perform one step of the multiwavelet transform in the horizontal direction. Consider the first $N/2$ rows. For $n = 0, \dots, (N/2) - 1$ we feed the

multiwavelet transform the vectors $[u_{n,m}^{(0,0)}, u_{n,m}^{(0,1)}]^T$, $m = 0, \ldots, (N/2) - 1$. We thus obtain

$$\begin{bmatrix} v_{n,l}^{(0,0)} \\ v_{n,l}^{(0,1)} \end{bmatrix} = \sum_{m=0}^{\frac{N}{2}-1} \mathbf{G}_{m-2l} \begin{bmatrix} u_{n,m}^{(0,0)} \\ u_{n,m}^{(0,1)} \end{bmatrix}; \qquad \begin{bmatrix} w_{n,l}^{(0,0)} \\ w_{n,l}^{(0,1)} \end{bmatrix} = \sum_{m=0}^{\frac{N}{2}-1} \mathbf{H}_{m-2l} \begin{bmatrix} u_{n,m}^{(0,0)} \\ u_{n,m}^{(0,1)} \end{bmatrix},$$

for $l = 0, \ldots, (N/4)-1$. For the next $N/2$ rows, for $n = 0, \ldots, (N/2)-1$, we feed the multiwavelet transform the vectors $[u_{n,m}^{(1,0)}, u_{n,m}^{(1,1)}]^T$, $m = 0, \ldots, (N/2) - 1$. We thus obtain

$$\begin{bmatrix} v_{n,l}^{(1,0)} \\ v_{n,l}^{(1,1)} \end{bmatrix} = \sum_{m=0}^{\frac{N}{2}-1} \mathbf{G}_{m-2l} \begin{bmatrix} u_{n,m}^{(1,0)} \\ u_{n,m}^{(1,1)} \end{bmatrix}; \qquad \begin{bmatrix} w_{n,l}^{(1,0)} \\ w_{n,l}^{(1,1)} \end{bmatrix} = \sum_{m=0}^{\frac{N}{2}-1} \mathbf{H}_{m-2l} \begin{bmatrix} u_{n,m}^{(1,0)} \\ u_{n,m}^{(1,1)} \end{bmatrix},$$

for $l = 0, \ldots, (N/4) - 1$. The resulting matrix takes the form

$$\begin{bmatrix} v_{0,0}^{(0,0)} & \cdots & v_{0,0}^{(0,1)} & \cdots & w_{0,0}^{(0,0)} & \cdots & w_{0,0}^{(0,1)} & \cdots \\ v_{1,0}^{(0,0)} & \cdots & v_{1,0}^{(0,1)} & \cdots & w_{1,0}^{(0,0)} & \cdots & w_{1,0}^{(0,1)} & \cdots \\ \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots \\ v_{\frac{N}{2}-1,0}^{(0,0)} & \cdots & v_{\frac{N}{2}-1,0}^{(0,1)} & \cdots & w_{\frac{N}{2}-1,0}^{(0,0)} & \cdots & w_{\frac{N}{2}-1,0}^{(0,1)} & \cdots \\ v_{0,0}^{(1,0)} & \cdots & v_{0,0}^{(1,1)} & \cdots & w_{0,0}^{(1,0)} & \cdots & w_{0,0}^{(1,1)} & \cdots \\ v_{1,0}^{(1,0)} & \cdots & v_{1,0}^{(1,1)} & \cdots & w_{1,0}^{(1,0)} & \cdots & w_{1,0}^{(1,1)} & \cdots \\ \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots \\ v_{\frac{N}{2}-1,0}^{(1,0)} & \cdots & v_{\frac{N}{2}-1,0}^{(1,1)} & \cdots & w_{\frac{N}{2}-1,0}^{(1,0)} & \cdots & w_{\frac{N}{2}-1,0}^{(1,1)} & \cdots \end{bmatrix},$$

where we have only given the first element of each row sequence. Finally we perform one step of the multiwavelet transform in the vertical direction. Consider the first $N/4$ columns. For $l = 0, \ldots, (N/4)-1$ we feed the multiwavelet transform the vectors $[v_{n,l}^{(0,0)}, v_{n,l}^{(1,0)}]^T$, $n = 0, \ldots, (N/2) - 1$ to obtain

$$\begin{bmatrix} vv_{k,l}^{(0,0)} \\ vv_{k,l}^{(1,0)} \end{bmatrix} = \sum_{n=0}^{\frac{N}{2}-1} \mathbf{G}_{n-2k} \begin{bmatrix} v_{n,l}^{(0,0)} \\ v_{n,l}^{(1,0)} \end{bmatrix}; \qquad \begin{bmatrix} wv_{k,l}^{(0,0)} \\ wv_{k,l}^{(1,0)} \end{bmatrix} = \sum_{n=0}^{\frac{N}{2}-1} \mathbf{H}_{n-2k} \begin{bmatrix} v_{n,l}^{(0,0)} \\ v_{n,l}^{(1,0)} \end{bmatrix},$$

for $k = 0, \ldots, (N/4) - 1$. With respect to the next $N/4$ columns, for $l = 0, \ldots, (N/4)-1$ we feed the multiwavelet transform the vectors $[v_{n,l}^{(0,1)}, v_{n,l}^{(1,1)}]^T$, $n = 0, \ldots, (N/2) - 1$ to obtain

$$\begin{bmatrix} vv_{k,l}^{(0,1)} \\ vv_{k,l}^{(1,1)} \end{bmatrix} = \sum_{n=0}^{\frac{N}{2}-1} \mathbf{G}_{n-2k} \begin{bmatrix} v_{n,l}^{(0,1)} \\ v_{n,l}^{(1,1)} \end{bmatrix}; \qquad \begin{bmatrix} wv_{k,l}^{(0,1)} \\ wv_{k,l}^{(1,1)} \end{bmatrix} = \sum_{n=0}^{\frac{N}{2}-1} \mathbf{H}_{n-2k} \begin{bmatrix} v_{n,l}^{(0,1)} \\ v_{n,l}^{(1,1)} \end{bmatrix},$$

for $k = 0, \ldots, (N/4)-1$. With respect to the third set of $N/4$ columns, for $l = 0, \ldots, (N/4)-1$ we feed the multiwavelet transform the vectors $[w_{n,l}^{(0,0)}, w_{n,l}^{(1,0)}]^T$,

$n = 0, \ldots, (N/2) - 1$ to obtain

$$\begin{bmatrix} vw_{k,l}^{(0,0)} \\ vw_{k,l}^{(1,0)} \end{bmatrix} = \sum_{n=0}^{\frac{N}{2}-1} \mathbf{G}_{n-2k} \begin{bmatrix} w_{n,l}^{(0,0)} \\ w_{n,l}^{(1,0)} \end{bmatrix}; \qquad \begin{bmatrix} ww_{k,l}^{(0,0)} \\ ww_{k,l}^{(1,0)} \end{bmatrix} = \sum_{n=0}^{\frac{N}{2}-1} \mathbf{H}_{n-2k} \begin{bmatrix} w_{n,l}^{(0,0)} \\ w_{n,l}^{(1,0)} \end{bmatrix},$$

for $k = 0, \ldots, (N/4)-1$. For the final $N/4$ columns, for $l = 0, \ldots, (N/4)-1$ we feed the multiwavelet transform the vectors $[w_{n,l}^{(0,1)}, w_{n,l}^{(1,1)}]^T$, $n = 0, \ldots, (N/2)-1$ to obtain

$$\begin{bmatrix} vw_{k,l}^{(0,1)} \\ vw_{k,l}^{(1,1)} \end{bmatrix} = \sum_{n=0}^{\frac{N}{2}-1} \mathbf{G}_{n-2k} \begin{bmatrix} w_{n,l}^{(0,1)} \\ w_{n,l}^{(1,1)} \end{bmatrix}; \qquad \begin{bmatrix} ww_{k,l}^{(0,1)} \\ wv_{k,l}^{(1,1)} \end{bmatrix} = \sum_{n=0}^{\frac{N}{2}-1} \mathbf{H}_{n-2k} \begin{bmatrix} w_{n,l}^{(0,1)} \\ w_{n,l}^{(1,1)} \end{bmatrix},$$

for $k = 0, \ldots, (N/4) - 1$. The final matrix obtained takes the form:

$$\left[\begin{pmatrix} vv_{0,0}^{(0,0)} & \ldots & vv_{0,0}^{(0,1)} & \ldots \\ vv_{1,0}^{(0,0)} & \ldots & vv_{1,0}^{(0,1)} & \ldots \\ \vdots & \ddots & \vdots & \ddots \\ vv_{\frac{N}{4}-1,0}^{(0,0)} & \ldots & vv_{\frac{N}{4}-1,0}^{(0,1)} & \ldots \\ vv_{0,0}^{(1,0)} & \ldots & vv_{0,0}^{(1,1)} & \ldots \\ vv_{1,0}^{(1,0)} & \ldots & vv_{1,0}^{(1,1)} & \ldots \\ \vdots & \ddots & \vdots & \ddots \\ vv_{\frac{N}{4}-1,0}^{(1,0)} & \ldots & vv_{\frac{N}{4}-1,0}^{(1,1)} & \ldots \end{pmatrix} \begin{pmatrix} vw_{0,0}^{(0,0)} & \ldots & vw_{0,0}^{(0,1)} & \ldots \\ vw_{1,0}^{(0,0)} & \ldots & vw_{1,0}^{(0,1)} & \ldots \\ \vdots & \ddots & \vdots & \ddots \\ vw_{\frac{N}{4}-1,0}^{(0,0)} & \ldots & vw_{\frac{N}{4}-1,0}^{(0,1)} & \ldots \\ vw_{0,0}^{(1,0)} & \ldots & vw_{0,0}^{(1,1)} & \ldots \\ vw_{1,0}^{(1,0)} & \ldots & vw_{1,0}^{(1,1)} & \ldots \\ \vdots & \ddots & \vdots & \ddots \\ vw_{\frac{N}{4}-1,0}^{(1,0)} & \ldots & vw_{\frac{N}{4}-1,0}^{(1,1)} & \ldots \end{pmatrix} \right.$$
$$\left. \begin{pmatrix} wv_{0,0}^{(0,0)} & \ldots & wv_{0,0}^{(0,1)} & \ldots \\ wv_{1,0}^{(0,0)} & \ldots & wv_{1,0}^{(0,1)} & \ldots \\ \vdots & \ddots & \vdots & \ddots \\ wv_{\frac{N}{4}-1,0}^{(0,0)} & \ldots & wv_{\frac{N}{4}-1,0}^{(0,1)} & \ldots \\ wv_{0,0}^{(1,0)} & \ldots & wv_{0,0}^{(1,1)} & \ldots \\ wv_{1,0}^{(1,0)} & \ldots & wv_{1,0}^{(1,1)} & \ldots \\ \vdots & \ddots & \vdots & \ddots \\ wv_{\frac{N}{4}-1,0}^{(1,0)} & \ldots & wv_{\frac{N}{4}-1,0}^{(1,1)} & \ldots \end{pmatrix} \begin{pmatrix} ww_{0,0}^{(0,0)} & \ldots & ww_{0,0}^{(0,1)} & \ldots \\ ww_{1,0}^{(0,0)} & \ldots & ww_{1,0}^{(0,1)} & \ldots \\ \vdots & \ddots & \vdots & \ddots \\ ww_{\frac{N}{4}-1,0}^{(0,0)} & \ldots & ww_{\frac{N}{4}-1,0}^{(0,1)} & \ldots \\ ww_{0,0}^{(1,0)} & \ldots & ww_{0,0}^{(1,1)} & \ldots \\ ww_{1,0}^{(1,0)} & \ldots & ww_{1,0}^{(1,1)} & \ldots \\ \vdots & \ddots & \vdots & \ddots \\ ww_{\frac{N}{4}-1,0}^{(1,0)} & \ldots & ww_{\frac{N}{4}-1,0}^{(1,1)} & \ldots \end{pmatrix} \right],$$

where we have again only given the first element of each row sequence. The full matrix has been split into four submatrices which may be denoted

$$\begin{bmatrix} \mathbf{GG} & \mathbf{GH} \\ \mathbf{HG} & \mathbf{HH} \end{bmatrix},$$

where, e.g., '**GH**' denotes application of the '**H**' filter coefficients in the horizontal direction, followed by application of the '**G**' filter coefficients in the

vertical direction. These four submatrices are each further subdivided into four matrices, i.e., those with superscripts $(0,0), (0,1), (1,0)$ and $(1,1)$. From the original $N \times N$ array we thus obtain 16 submatrices of sizes $N/4 \times N/4$. Examples of this one-step processing are given in Figs. 9 and 10 for GHM and CL, respectively, both with approximation preprocessing. The image is the well-known photo of Lenna.

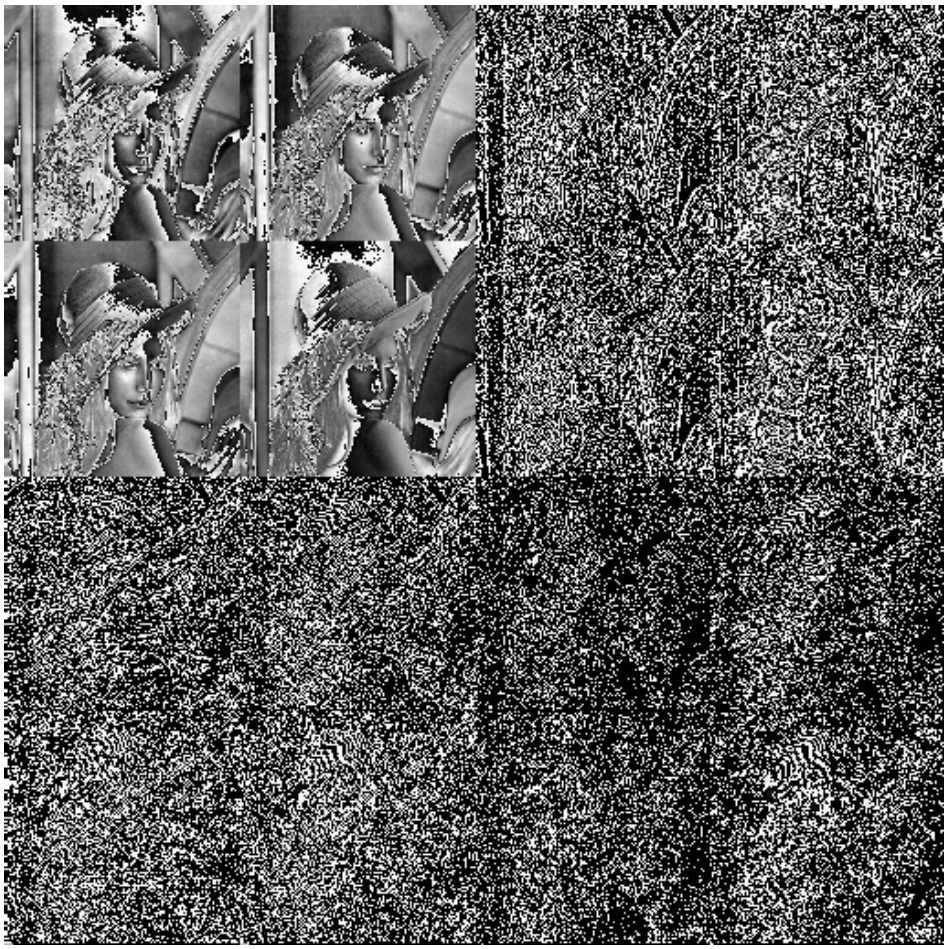


Figure 9: Result of one step of the 2D wavelet cascade algorithm for GHM with approximation preprocessing.

In the CL one-step plot almost all information is concentrated in one block of **GG** with superscripts $(0,0)$ (in the scalar case we would expect a picture similar to the **GG** block after one step of the cascade algorithm). This reflects the fact that for CL one scaling function is symmetric and the other is anti-symmetric. The $(0,0)$ block of the **GG** matrix contains the inner products of the 2D image function $F(x,y)$, say, with the product of the shifted scaling

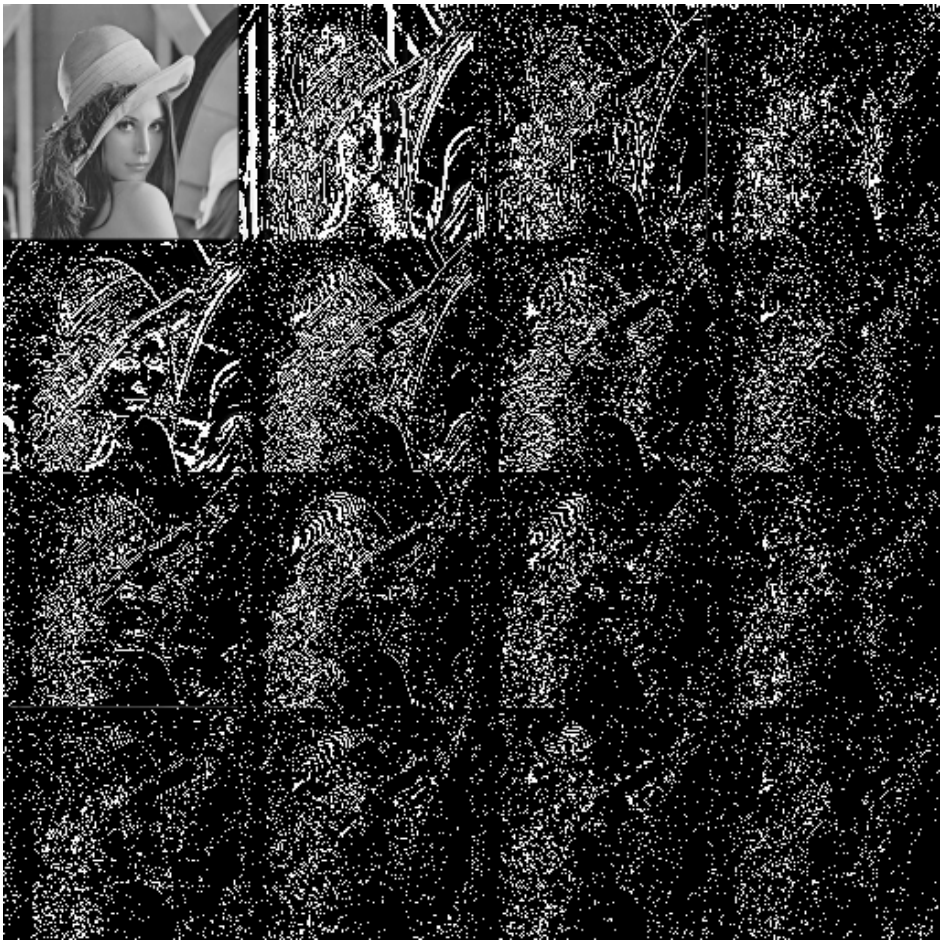


Figure 10: Result of one step of the 2D wavelet cascade algorithm for CL with approximation preprocessing.

function $\phi_1(x)\phi_1(y)$ (see (2.9) for the 1D case):

$$vv_{k,l}^{(0,0)} = \int\int F(x,y)\phi_1(x-l)\phi_1(y-k)dxdy.$$

Analogously,

$$vv_{k,l}^{(1,0)} = \int\int F(x,y)\phi_1(x-l)\phi_2(y-k)dxdy,$$

$$vv_{k,l}^{(0,1)} = \int\int F(x,y)\phi_2(x-l)\phi_1(y-k)dxdy,$$

$$vv_{k,l}^{(1,1)} = \int\int F(x,y)\phi_2(x-l)\phi_2(y-k)dxdy.$$

Here ϕ_1 is the symmetric scaling function of the CL pair, and ϕ_2 is the antisymmetric one. As is well known, the inner product of the signal with a

symmetric function corresponds to low-pass filtering, while the inner product with an antisymmetric function corresponds to high-pass filtering. This means that the $(0,0)$ block contains low-pass data in both directions. On the other hand, $(0,1), (1,0)$ and $(0,0)$ blocks correspond to high-pass filtering in one of the directions and contain mostly the information about edges. Nevertheless all four parts of $\mathbf{GG}$ come from the inner products with the components of the 2D multiscaling function and should be considered as the result of *low-pass multifiltering*.

In the GHM one-step plot all four sub-matrices of $\mathbf{GG}$ contribute in a similar fashion, since both scaling functions are symmetric, and hence $(0,0), (0,1)$, $(1,0)$ and $(1,1)$ coefficients are all a result of low-pass filtering in both directions.

From the structures of the submatrices we find that

$$\begin{bmatrix} vv_{k,l}^{(0,0)} \\ vv_{k,l}^{(0,1)} \\ vv_{k,l}^{(1,0)} \\ vv_{k,l}^{(1,1)} \end{bmatrix} = \sum_{n=0}^{\frac{N}{2}-1} \sum_{m=0}^{\frac{N}{2}-1} \mathbf{G}_{n-2k} \otimes \mathbf{G}_{m-2l} \begin{bmatrix} u_{n,m}^{(0,0)} \\ u_{n,m}^{(0,1)} \\ u_{n,m}^{(1,0)} \\ u_{n,m}^{(1,1)} \end{bmatrix},$$

$$\begin{bmatrix} vw_{k,l}^{(0,0)} \\ vw_{k,l}^{(0,1)} \\ vw_{k,l}^{(1,0)} \\ vw_{k,l}^{(1,1)} \end{bmatrix} = \sum_{n=0}^{\frac{N}{2}-1} \sum_{m=0}^{\frac{N}{2}-1} \mathbf{G}_{n-2k} \otimes \mathbf{H}_{m-2l} \begin{bmatrix} u_{n,m}^{(0,0)} \\ u_{n,m}^{(0,1)} \\ u_{n,m}^{(1,0)} \\ u_{n,m}^{(1,1)} \end{bmatrix},$$

$$\begin{bmatrix} wv_{k,l}^{(0,0)} \\ wv_{k,l}^{(0,1)} \\ wv_{k,l}^{(1,0)} \\ wv_{k,l}^{(1,1)} \end{bmatrix} = \sum_{n=0}^{\frac{N}{2}-1} \sum_{m=0}^{\frac{N}{2}-1} \mathbf{H}_{n-2k} \otimes \mathbf{G}_{m-2l} \begin{bmatrix} u_{n,m}^{(0,0)} \\ u_{n,m}^{(0,1)} \\ u_{n,m}^{(1,0)} \\ u_{n,m}^{(1,1)} \end{bmatrix},$$

$$\begin{bmatrix} ww_{k,l}^{(0,0)} \\ ww_{k,l}^{(0,1)} \\ ww_{k,l}^{(1,0)} \\ ww_{k,l}^{(1,1)} \end{bmatrix} = \sum_{n=0}^{\frac{N}{2}-1} \sum_{m=0}^{\frac{N}{2}-1} \mathbf{H}_{n-2k} \otimes \mathbf{H}_{m-2l} \begin{bmatrix} u_{n,m}^{(0,0)} \\ u_{n,m}^{(0,1)} \\ u_{n,m}^{(1,0)} \\ u_{n,m}^{(1,1)} \end{bmatrix},$$

where $\otimes$ denotes matrix Kronecker product, and $0 \le n, m \le (N/2) - 1$ and $0 \le k, l \le (N/4) - 1$.

The next step of the algorithm is to apply the multiwavelet transform horizontally and then vertically to the submatrix $\mathbf{GG}$, which has exactly the same layout as $\mathbf{U}$, except that the dimension of the matrix is halved. The process then continues in cascade fashion.

Initial data can be reconstructed exactly from the multiwavelet coefficients. Reconstruction starts with the submatrix $\begin{bmatrix} \mathbf{GG}^J & \mathbf{GH}^J \\ \mathbf{HG}^J & \mathbf{HH}^J \end{bmatrix}$, of the highest decomposition level J and employs formulas of the type (2.12) first in the vertical

and then in the horizontal direction in order to get the low-pass matrix $\mathbf{G}\mathbf{G}^{J-1}$ of the next level. The process then continues until the matrix $\mathbf{U}$ is reconstructed. Finally, the initial data can be recovered from $\mathbf{U}$ by post-processing vertically and then horizontally.

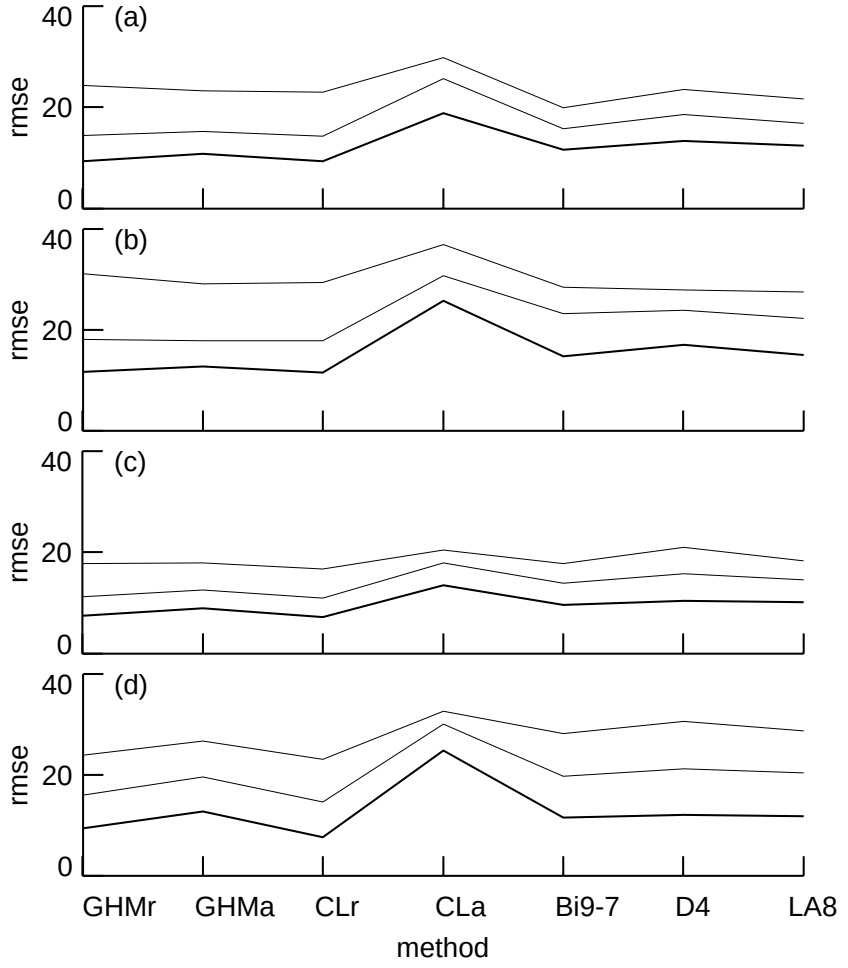


Figure 11: Average (over 100 realizations) rmse results for methods 1-7 of Table 2, for (a) Lenna, (b) a fingerprint, (c) the ship 'Picardie,' and (d) cartoon bear. Scalar thresholding used throughout; signal-to-noise ratios are 8 (thick lines), 4 (medium) and 2 (thin).

8.2 Image denoising

Taking into account the results of our 1D experiments, we decided to restrict ourselves to orthogonal multifilters for the denoising of images. In this case, the only change in variance of the input noise is due to the preprocessing. Analogously to the 1D case, for the thresholding we make use of the 'aver-

age' variance of the preprocessed and transformed coefficients; the variance deflation/inflation factor is the square of that obtained in the 1D case.

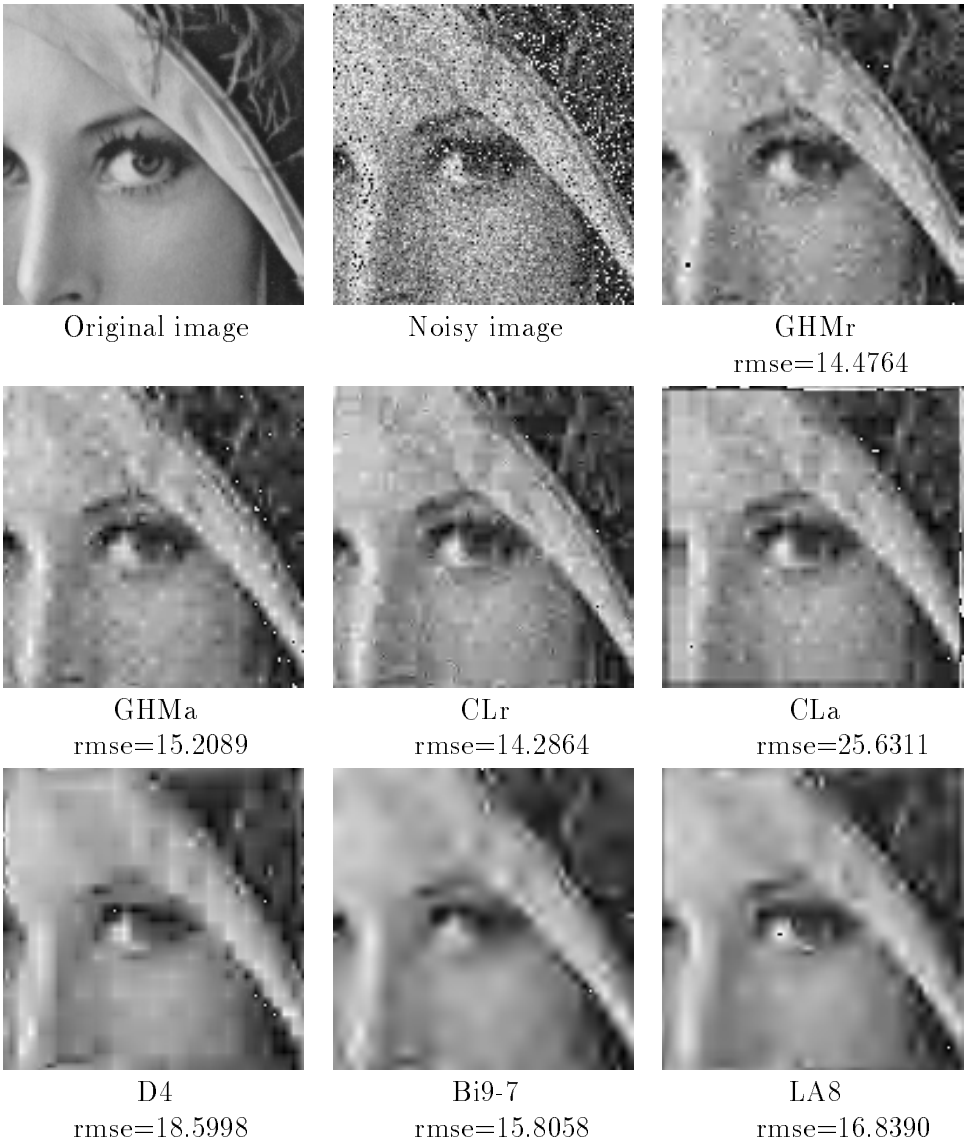


Figure 12: Example of 2D processing methods applied to Lenna image, for S/N=4.

To see this, consider an $N \times N$ image where each cell has noise with variance σ_ϵ^2. Then, for example, for 2D GHM repeated row preprocessing, which uses $\alpha = 1/\sqrt{2}$, we firstly preprocess all rows, thus producing an unaltered $N \times N$ image, A say, and another scaled $N \times N$ image 'to its right,' B say, where each cell has been scaled by $1/\sqrt{2}$ relative to A. Next we preprocess all the

$2N$ columns in A and B. This gives four $N \times N$ images, A left unaltered, plus a scaled version 'below it,' C say, where each cell has been scaled by $1/\sqrt{2}$ relative to A; the image B is left unaltered, while 'below it' is created a scaled version of B, where each cell has been scaled by $1/\sqrt{2} \times 1/\sqrt{2} = 1/2$ relative to A. We thus have N^2 entries with noise variance σ_ϵ^2, $2N^2$ entries with variance $\sigma_\epsilon^2/2$, and N^2 entries with variance $\sigma_\epsilon^2/4$. The average variance is then $(9/16)\sigma_\epsilon^2$, and $c = 9/16$ is the square of the factor $(3/4)$ obtained in the 1D case (e.g., Table 2). For 2D CL repeated row preprocessing, which uses $\alpha = 0$, only N^2 coefficients of the $4N^2$ produced are non-zero, and these have variance σ_ϵ^2, so that the average variance is $\sigma_\epsilon^2/4$, and $c = 1/4$ is the square of the value of $1/2$ obtained in the 1D case. For GHM or CL with matrix (critical sampling) preprocessing, the same rescaling takes place in the row direction and column direction, so that the 1D variance deflation/inflation factor c in Table 2 becomes squared in the 2D case.

The performance of methods 1-7 of Table 2 on images was tested with four different 128×128-cell images. In implementing the scalar thresholding for these images, for which $N = 128$, N' and c in Table 2 are replaced by their squares, as explained above. The four images used were (i) of Lenna again, (ii) a fingerprint, (iii) a photo of part of a ship including its name, 'Picardie,' and (iv) a textural image containing part of a cartoon bear. For each image, 100 realizations of the noisy image were constructed, for S/N ratios of 2, 4 and 8. After preprocessing, DMWT, hard thresholding, inverse DMWT, postprocessing, (and, for scalar methods, DWT, hard thresholding, inverse DWT), the rmse was computed for each method and each realization.

The average rmse over realizations was computed, and is shown in Fig. 11.

For the Lenna image (i), the best result is achieved using CL repeated row preprocessing for S/N=8 and 4, but for S/N=2 Bi 9-7 was best; however in this case CL repeated row preprocessing was still the best of the multiwavelet methods. Examples of the processing are shown in Figs. 12 & 13 for S/N=4 and 2, respectively, where CL repeated row preprocessing is seen visually to perform well, and would probably be felt to have performed *visually* better than Bi 9-7 for S/N=2 in this example.

For the fingerprint image (ii), CL repeated row preprocessing is best for S/N=8, is tie-best with GHM approximation preprocessing for S/N=4, but for S/N=2, LA(8) performed best. Examples of the processing are shown in Figs. 14 & 15 for S/N=4 and 2, respectively, where CL repeated row preprocessing is seen visually to perform well for all S/N ratios, and is visually better than LA(8) for S/N=2 in this example.

For the ship image (iii), CL repeated row preprocessing is best for all S/N ratios. Processing examples are given in Figs. 16 & 17 for for S/N=4 and 2. Some letters of the name are interpretable at S/N=4 for GHM and CL repeated row preprocessing; when the S/N is as low as 2, these same methods

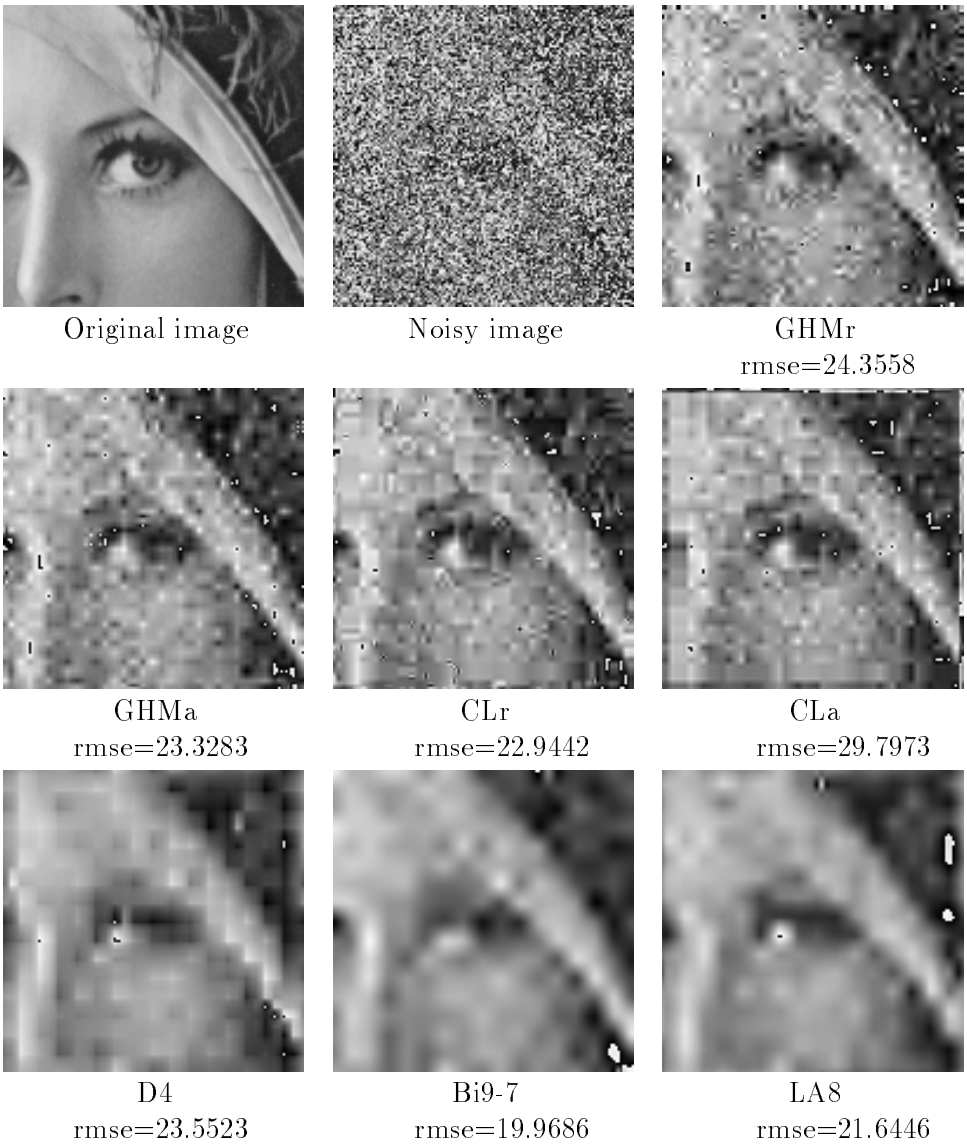


Figure 13: Example of 2D processing methods applied to Lenna image, for S/N=2.

preserve the main characteristics of the image, but the name is unreadable.

For the textural cartoon-bear image (iv), CL repeated row preprocessing is best for all S/N ratios. GHM repeated row preprocessing does nearly as well. Processing examples are given in Figs. 18 & 19 for S/N=4 and 2. GHM and CL repeated row preprocessing do well in preserving the speckle texture in the upper right of the image, and in preserving the continuity of lines, even for S/N=2.

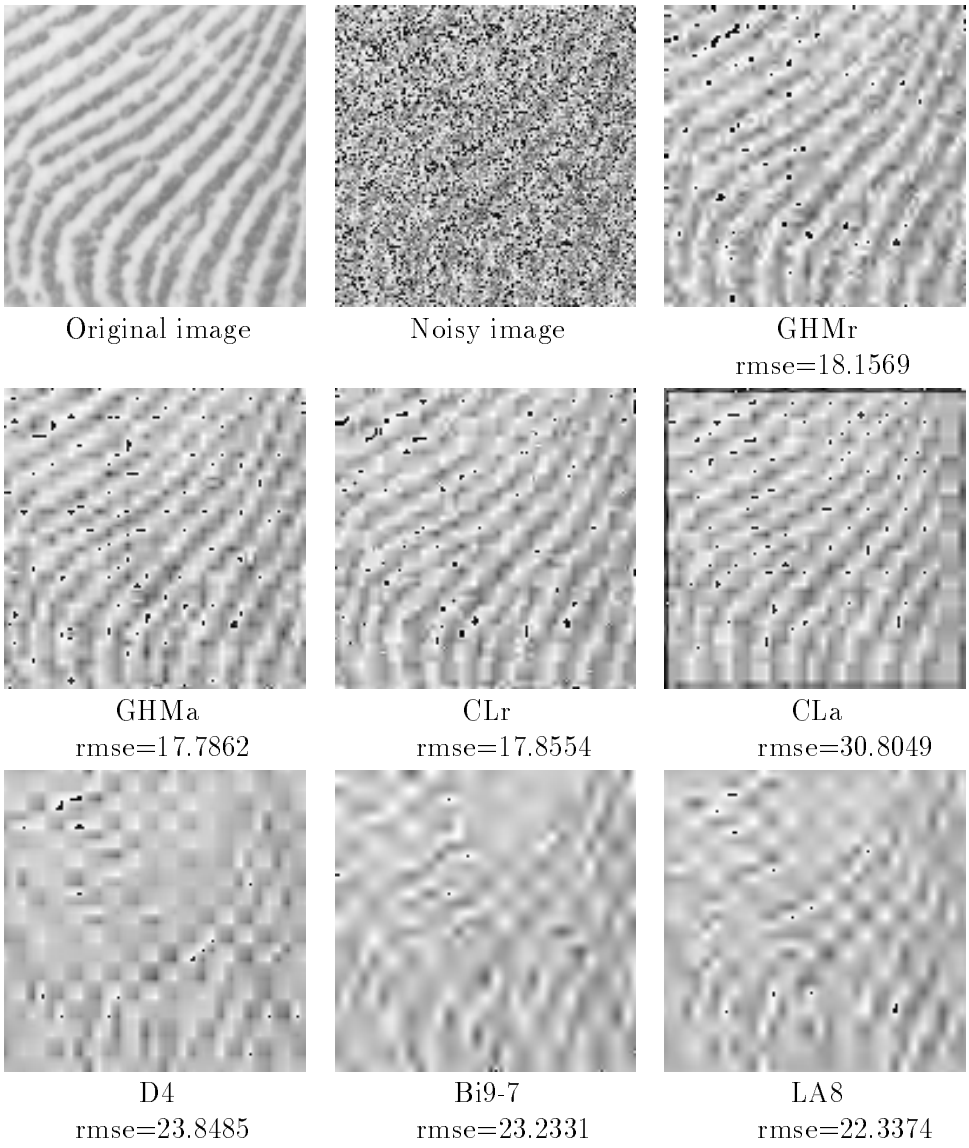


Figure 14: Example of 2D processing methods applied to fingerprint image, for S/N=4.

Multiwavelets generally outperform scalar wavelets for image denoising for all models; only for Lenna and fingerprints with S/N=2 do scalar wavelets perform best. The rmse obtained for the best method varies (over image type) between 45-65% of the actual standard deviation of the simulated noise for S/N=8, between 35-60% for S/N=4, and between 25-50% for S/N=2. Again we see intrinsic noise can be reduced by practically useful amounts. As for 1D signal processing, CL repeated row preprocessing seems a good general

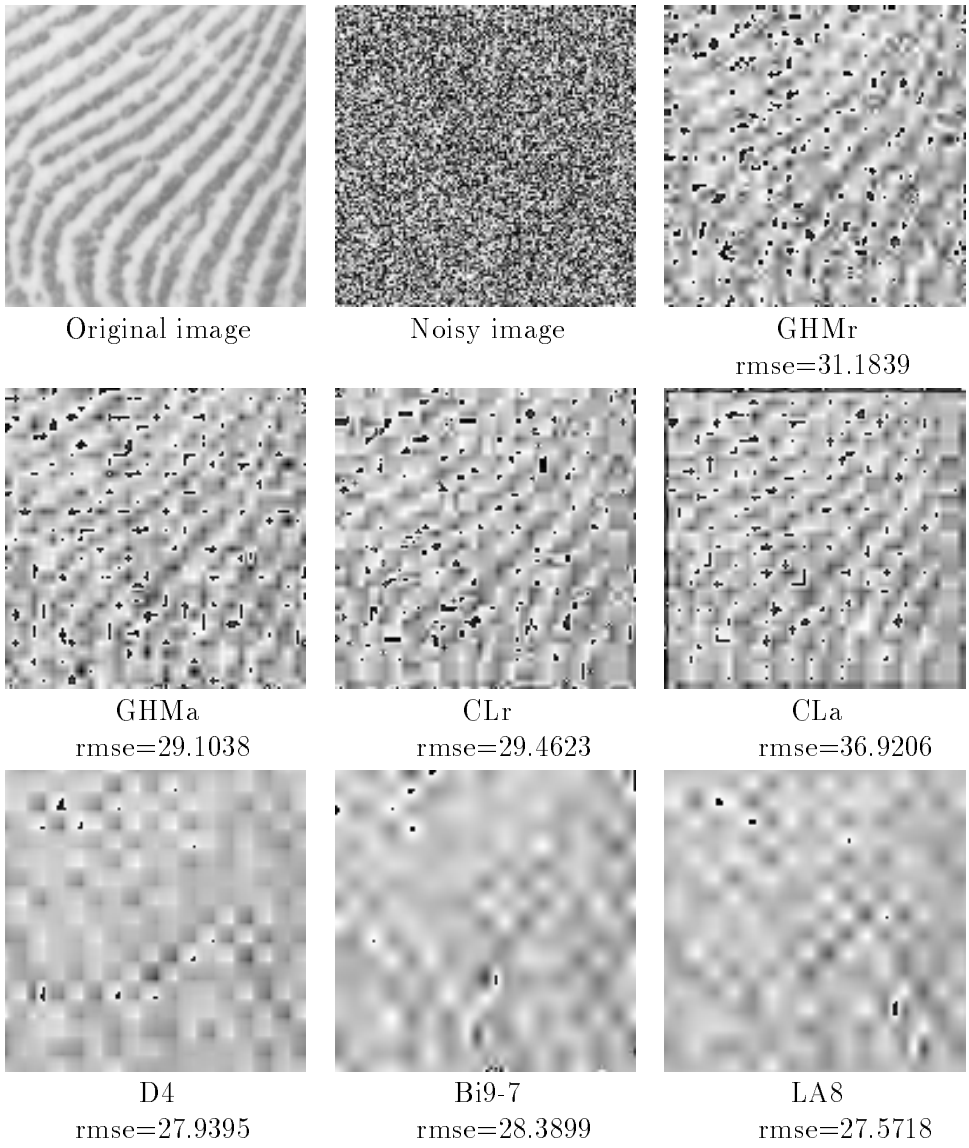


Figure 15: Example of 2D processing methods applied to fingerprint image, for S/N=2.

method; Table 3 gives the processing gains for this method against S/N ratio and image type.

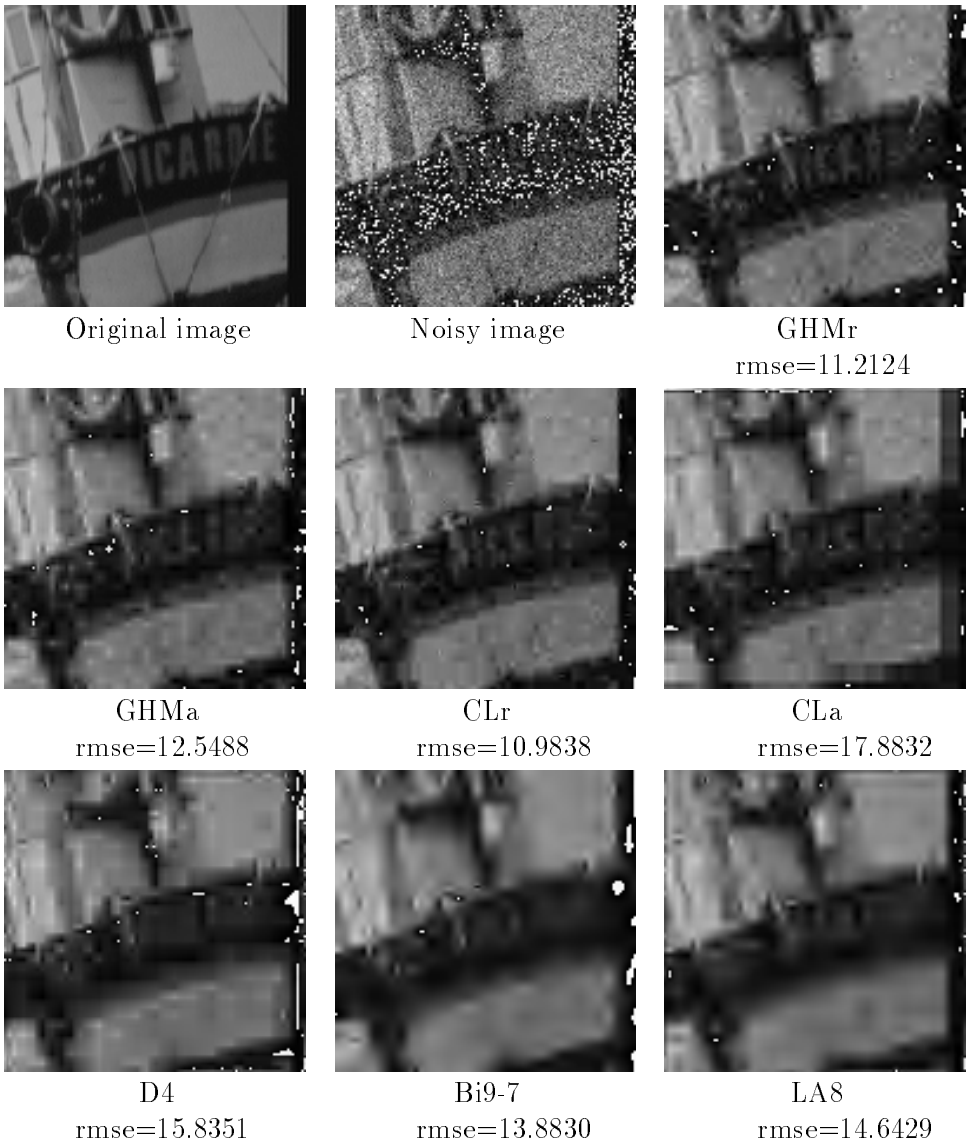


Figure 16: Example of 2D processing methods applied to 'Picardie' boat image, for S/N=4.

9 Concluding remarks

In this paper we studied the covariance structure of 1D scalar orthogonal, scalar biorthogonal and multiple orthogonal and multiple biorthogonal wavelet transforms with appropriate preprocessing. This led to a careful formulation of the universal threshold for scalar thresholding. Vector thresholding was also considered.

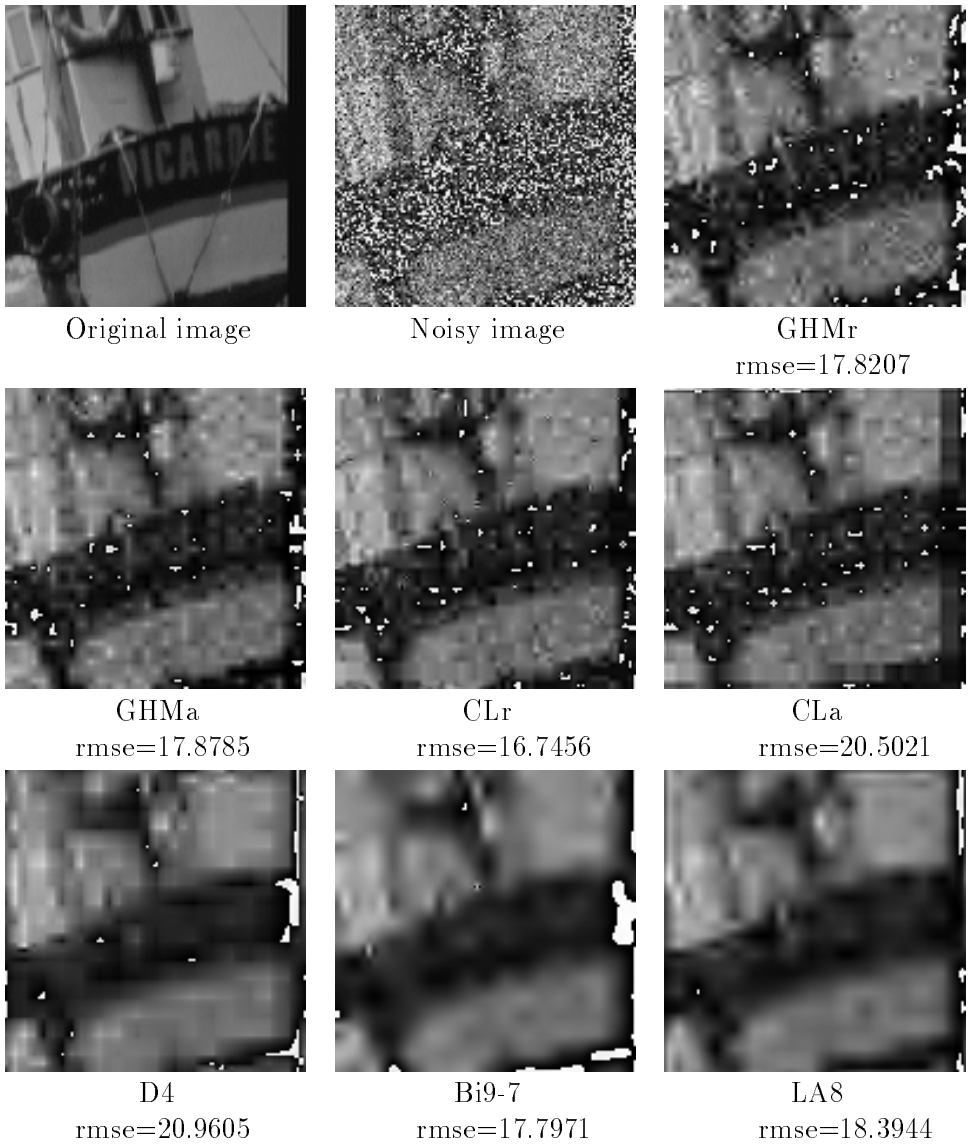


Figure 17: Example of 2D processing methods applied to 'Picardie' boat image, for S/N=2.

Multiwavelets outperform scalar wavelets for three out of four noisy 1D test signals. Vector thresholding does not always outperform scalar thresholding. Multiwavelets generally outperform scalar wavelets for image denoising for all four noisy 2D test images, and the results are visually very impressive. Chui-Lian scaling functions and wavelets combined with repeated row preprocessing appears to be a good general method. For both 1D and 2D cases, the reconstructed signals derived from such a good general method demon-

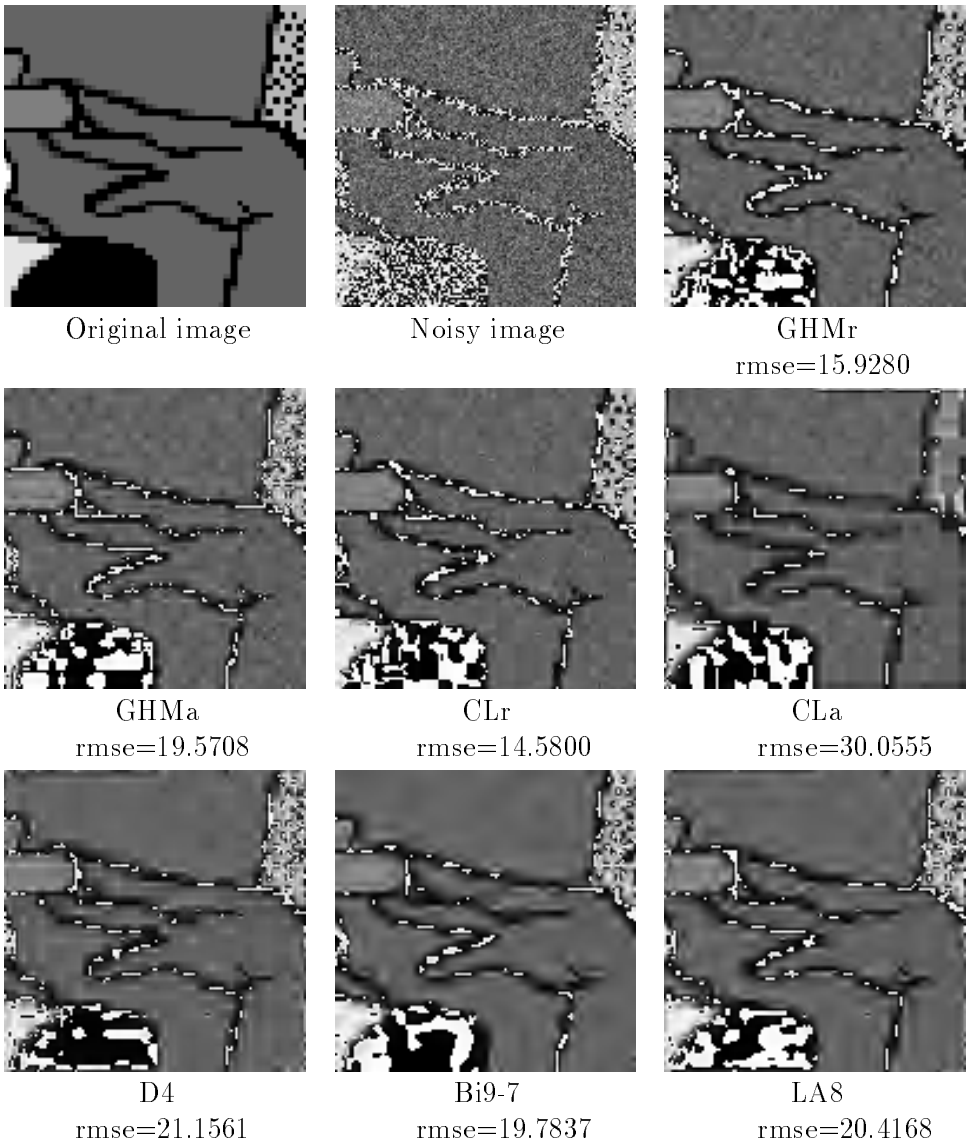


Figure 18: Example of 2D processing methods applied to cartoon bear image, for S/N=4.

strate much reduced noise levels — an average noise reduction gain of 7.5dB is achieved over test examples. Still better results could be achievable using an undecimated multiple wavelet transform due to its shift invariance.

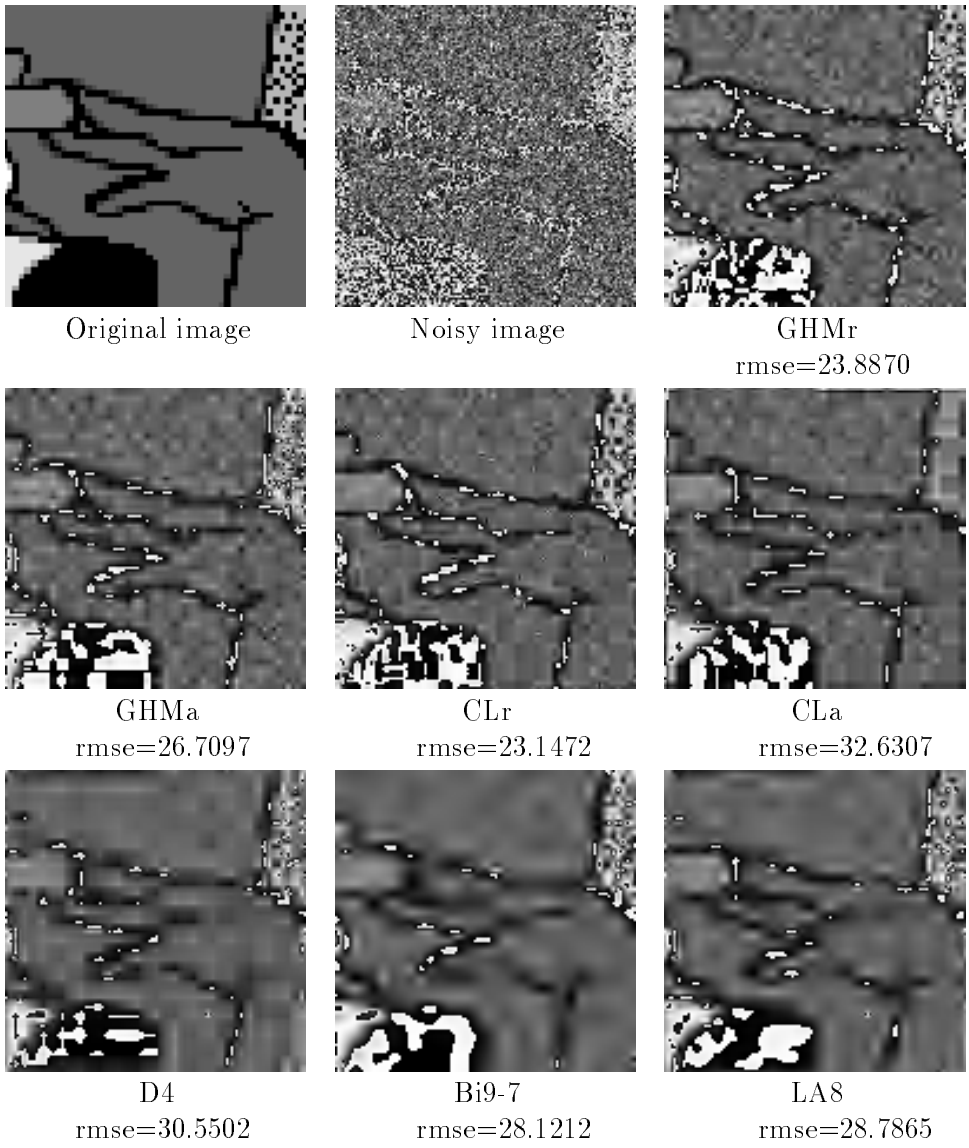


Figure 19: Example of 2D processing methods applied to cartoon bear image, for S/N=2.

Acknowledgements

This work was done while Vasily Strela was at Imperial College on EPSRC grant GR/L11182. The authors are very grateful for this support. A version of this paper was presented on 7 December 1998 at the Data Analysis Workshop, part of the programme on Nonlinear and Nonstationary Signal Processing at the Isaac Newton Institute for Mathematical Sciences, Cambridge, U.K.

References

Chui, C. K. and Lian J. A. (1996) 'A study of orthonormal multi-wavelets,' *Applied Numerical Mathematics* **20**, 273–98.

Cotronei, M., Montefusco, L. B. and Puccio, L. (1998) 'Multiwavelet analysis and signal processing,' *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing* **45**, 970–87.

Daubechies, I. (1988) 'Orthonormal bases of compactly supported wavelets,' *Communications in Pure and Applied Mathematics* **41**, 909–96.

Daubechies, I. (1992) 'Ten Lectures on Wavelets.' Philadelphia: SIAM.

Donoho, D. L. and Johnstone, I. M. (1994) 'Ideal spatial adaptation by wavelet shrinkage,' *Biometrika* **81**, 425–55.

Donoho, D. L. and Johnstone, I. M. (1995) 'Adapting to unknown smoothness via wavelet shrinkage,' *Journal of the American Statistical Association* **90**, 1200–24.

Downie, T. R. and Silverman, B. W. (1998) 'The discrete multiple wavelet transform and thresholding methods,' *IEEE Transactions on Signal Processing* **46**, 2558–61.

Galambos, J. (1978) 'The Asymptotic Theory of Extreme Order Statistics.' New York: Wiley.

Geronimo, J., Hardin, D. and Massopust P. R. (1994) 'Fractal functions and wavelet expansions based on several functions,' *Journal of Approximation Theory* **78**, 373–401.

Goodman, T. N. T. and Lee, S. L. (1994) 'Wavelets of multiplicity r,' *Transactions of the American Mathematical Society* **342**, 307–24.

Heil, C., Strang, G. and Strela, V. (1996) 'Approximation by translates of refinable functions,' *Numerische Mathematik* **73**, 75–94.

Mallat, S. G. (1989) 'A theory for multiresolution signal decomposition: the wavelet representation,' *IEEE Transactions on Pattern Analysis and Machine Intelligence* **11**, 674–93.

Strela, V. (1998) 'A note on construction of biorthogonal multi-scaling functions,' in Contemporary Mathematics: Wavelets, Multiwavelets and their Applications, edited by A. Aldoubi and E. B. Lin, Philadelphia: AMS.

Strela, V. and Strang, G. (1995) 'Finite element multiwavelets,' in Approximation Theory, Wavelets and Applications, edited by S. P. Singh, Kluwer, 485–96.

Strela, V., Heller, P. N., Strang, G., Topiwala, P. and Heil, C. (1999) 'The application of multiwavelet filter banks to signal and image processing,' *IEEE Transactions on Image Processing* **8**, 548–63.

Xia, X.-G., Geronimo, J. S., Hardin, D. P. and Suter, B.W. (1996) 'Design of prefilters for discrete multiwavelet transforms,' *IEEE Transactions on Signal Processing* **44**, 25–35.