

**Image de-noising by integer wavelet
transforms and generalized cross
validation**

*Maarten Jansen
Geert Uytterhoeven
Adhemar Bultheel*

Report TW 264, August 1997



Katholieke Universiteit Leuven
Department of Computer Science
Celestijnenlaan 200A – B-3001 Heverlee (Belgium)

Image de-noising by integer wavelet transforms and generalized cross validation

Maarten Jansen
Geert Uytterhoeven
Adhemar Bultheel

Report TW 264, August 1997

Department of Computer Science, K.U.Leuven

Abstract

De-noising algorithms based on wavelet thresholding replace small wavelet coefficients by zero and keep or shrink the coefficients with absolute value above the threshold. The optimal threshold minimizes the error of the result as compared to the unknown, exact data. To estimate this optimal threshold, we use Generalized Cross Validation. This procedure does not require an estimate for the noise energy. This paper illustrates the method for wavelet transforms that map integer grey-scale pixel values to integer wavelet coefficients.

Keywords : Noise reduction, wavelets, integer transforms, thresholding, cross validation.

AMS(MOS) Classification : 41A30, 93E14, 60G35, 62G07, 65D10, 68U10

Image de-noising by integer wavelet transforms and generalized cross validation

Maarten Jansen, Geert Uytterhoeven, Adhemar Bultheel
Katholieke Universiteit Leuven, Department of Computer Science

August 18, 1997

Abstract

De-noising algorithms based on wavelet thresholding replace small wavelet coefficients by zero and keep or shrink the coefficients with absolute value above the threshold. The optimal threshold minimizes the error of the result as compared to the unknown, exact data. To estimate this optimal threshold, we use Generalized Cross Validation. This procedure does not require an estimate for the noise energy. This paper illustrates the method for wavelet transforms that map integer grey-scale pixel values to integer wavelet coefficients.

keywords:

Noise reduction, wavelets, integer transforms, thresholding, cross validation.

Contents

1	Introduction	2
2	The discrete wavelet transform	2
3	Lifting and integer transforms	3
4	Thresholding and generalized cross validation	5
4.1	Wavelet thresholding	5
4.2	Threshold selection	6
5	Integer de-noising	8
5.1	Generalized cross validation for integer wavelet coefficients	8
5.2	The minimization procedure	9
5.3	An example	10
6	Conclusions	12
	Acknowledgements	12

1 Introduction

In spite of continuous improvements in image acquisition techniques and hardware, image enhancement remains a useful and often necessary step. In the last several years, wavelet thresholding has shown remarkable results in digital image de-noising.

A wavelet threshold procedure [5] starts with a discrete wavelet transform of the pixel matrix. In a second step, coefficients beneath a certain threshold are replaced by zero. Inverse transform yields the result. The main issue in this procedure is the selection of the threshold. This parameter should be chosen so that the eventual result is as close as possible to the unknown noise-free image. This paper uses the method of generalized cross validation, which does not need an estimate of the amount of noise.

A classical wavelet transform maps floating point numbers to floating point numbers. However, most images consist of integer values only. Recently, an invertible transform has been proposed to convert integers to integers [2]. This procedure is based on the so called lifting scheme [13, 14], which is in the first instance an alternative and faster algorithm for a classical wavelet transform. However, the structure of this lifting scheme allows to extend the classical algorithm to cases with non-regular grids and to adapt the floating point algorithm to an integer version.

This paper describes experiments with a wavelet based de-noising algorithm that uses integer transforms and a threshold selection procedure based on generalized cross validation. Section 2 repeats some basic wavelet material, as far as necessary for further reference. Section 3 gives a short introduction on the lifting scheme and briefly explains the idea of integer wavelet transforms. Section 4 discusses wavelet thresholding and threshold selection by generalized cross validation. In Section 5, we combine all this into our algorithm. Section 6 is a conclusion.

2 The discrete wavelet transform

A one dimensional discrete wavelet transform is a repeated filter bank algorithm. The input is a vector, represented by the row of circles on a black background in figure 1. Typically neighboring points in this vector show strong correlations. The objective of the transform is to use these correlations to obtain a sparse representation of the input. Therefore, in the first step, the input is convolved with a high pass filter \tilde{g} and a low pass filter \tilde{h} . The result of the latter convolution is a smoothed version of the input. The high frequency part is captured by the first convolution, and contains a lot of very small numbers, due to the high correlations among neighboring input points. Since these convolutions both give a result with a size equal to that of the input, this procedure doubles the total number of data. Therefore, we omit half of these data by *subsampling*. The resulting high frequency coefficients are wavelet coefficients at the finest level. The low frequency output are scaling coefficients. In the second and following steps, the algorithm repeats the same procedure on this smoothed version of the input. In each step, the resulting wavelet coefficients contain information about a certain degree of detail. All together, these coefficients constitute a multi-resolution analysis of the input. These output coefficients have a grey background in Figure 1

The basic building block of a wavelet transform is a filter bank with filters \tilde{g} and \tilde{h} . This is

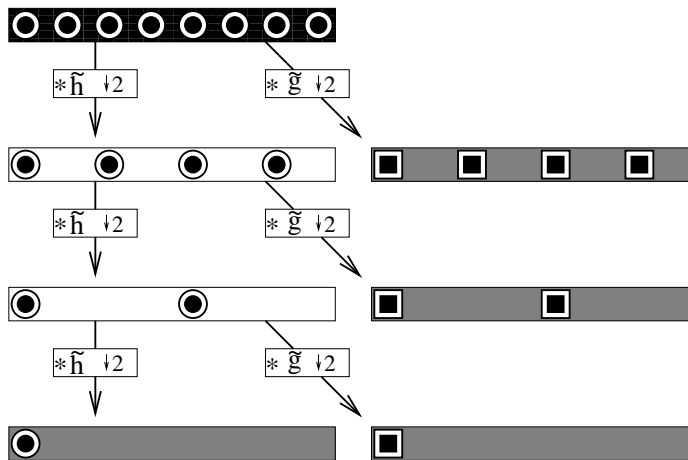


Figure 1: Successive steps of a fast decimated wavelet transform. After convolving the input with two filters \tilde{h} and \tilde{g} , the algorithm drops half of the result (*down-sampling*). Under certain conditions, perfect reconstruction of the original signal is possible from this decimated transform.

represented in Figure 2. This figure also shows the reconstruction, corresponding to one step of the filter bank decomposition. This reconstruction starts by *upsampling* the rows of coefficients: a zero is put in between two elements. The next step is a convolution with filters g and h . The results of these convolutions are added. Of course, some conditions on \tilde{g} , \tilde{h} , g , and h are necessary to make a perfect reconstruction of the input possible from the resulting data. We do not go into detail on this problem.

In two dimensions, we first apply one step on the row vectors and then on the column vectors. Figure 3 shows how this results in four classes of coefficients. Coefficients that result from a convolution with \tilde{g} in both directions (HH) represent diagonal features of the image, whereas a convolution with \tilde{h} in one direction and with \tilde{g} in the other, reflects vertical and horizontal information (HG and GH). In the next step we proceed with the low pass (LL) coefficients, which are results of a convolution with \tilde{h} in both directions. We end up with a transformed image in which each resolution level consists of coefficients for three *components*: LH, HL, and HH. At the coarsest level, we also keep low pass coefficients LL.

3 Lifting and integer transforms

The lifting scheme decomposes a filter bank operation in a number of consecutive lifting steps [4]. This series starts by splitting of the input vector into points with odd and even index (Figure 4). There exists two kinds of lifting operations. The first, called dual lifting, subtracts a filtered version of the “even” input to the “odd” input. The second, called primal lifting, adds a filtered version of the dual lifting output to the so far untouched “even” input. One way to interpret the dual lifting step, is the following: we assume that the “even” input and the “odd” input are highly correlated. This is certainly true in the first lifting step, where the “even” and the “odd” input are directly originating from one input signal. We now try to *predict* the odd samples by

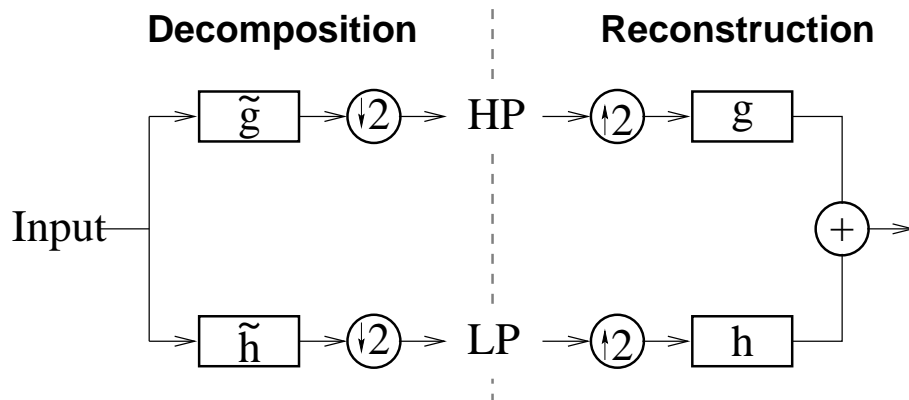


Figure 2: One step of a wavelet decomposition and its reconstruction. This is a filter bank: The input is filtered and downsampled to get a low pass signal LP and a high pass signal HP . Reconstruction starts with upsampling by introducing zeroes between every pair of points in LP and HP .

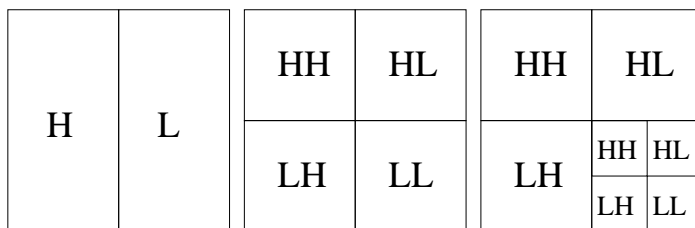


Figure 3: A two dimensional wavelet transform. First we apply one step of the one dimensional transform to all rows (left). Then, we repeat the same for all columns (middle). In the next step, we proceed with the coefficients that result from a convolution with \tilde{h} in both directions (right).

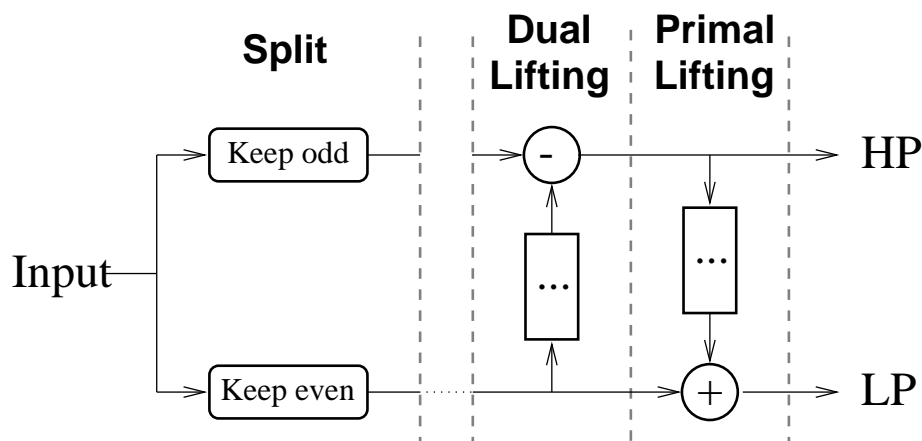


Figure 4: Decomposition of a filter bank into lifting steps.

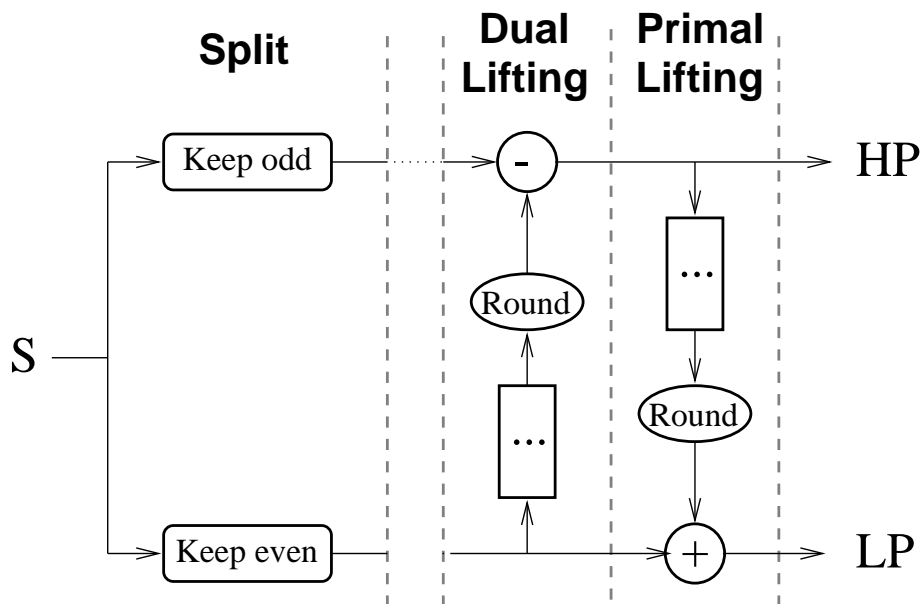


Figure 5: Integer wavelet transform.

a prediction operator (a filter) on the even ones. By subtracting this prediction from the odd samples, we reduce redundancy.

The inverse transform is now very easy to construct: follow the arrows in opposite directions and replace all plus-signs by minus-signs and vice versa. Note that this is not possible in the classical filter bank algorithm. It is possible in the lifting scheme, because at all places where a filter operation is performed, we also keep the input of this filter for the next step.

To obtain an invertible transform, the filter operation in the dual and primal lifting step need not even be linear. For instance, we could round the result of a given operation. This is the integer transform [2], illustrated in Figure 5. If the input is a vector with integer values, all intermediate results consist of integers.

4 Thresholding and generalized cross validation

4.1 Wavelet thresholding

We now suppose the input image is affected by additive, stationary noise. This noise can be colored or white. It can be proved [11] that the (classical) wavelet transform of stationary noise is stationary at each resolution level and within each component (vertical, horizontal, diagonal). Figure 6 illustrates this observation: the noise is spread evenly over all coefficients within each resolution level.

Secondly, we assume that the noise-free image can be well represented by a limited number of large wavelet coefficients. This decorrelating property also justifies the numerous wavelet based compression algorithms.

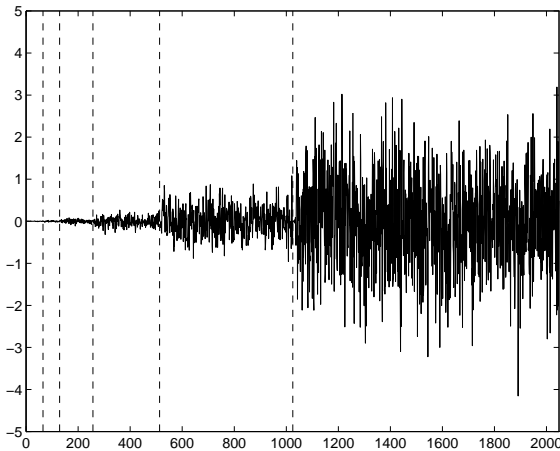


Figure 6: Classical wavelet transform of stationary, colored noise. We use Cohen-Daubechies-Feauveau (2,2)-wavelet filters. The noise was obtained as a convolution of white gaussian noise with a FIR-high pass-filter.

Thirdly, the noise should not be “too large”. In that case, the noise has a relatively small influence on the important large clean coefficients.

These three observations suggest to replace the small coefficients by zero, because they are dominated by noise and carry only a small amount of information. At the j -th resolution level, and for a given component c , all wavelet coefficients with absolute value below a certain, level-dependent, threshold δ_j^c , are classified as “noisy”. The algorithm replaces them by zero. The coefficients above the threshold are shrunk with its value δ_j^c . Figure 7 compares this “soft-thresholding” operation with the “hard-thresholding” alternative. The hard-thresholding procedure does not shrink the coefficients with absolute value above the threshold. Although at first sight this may seem a more natural approach, soft-thresholding is a more continuous operation, and it is mathematically more tractable. The figure also shows a more sophisticated *shrinking function*. It provides a continuous approach, while keeping the largest coefficients untouched. Not all shrinking methods use a threshold parameter. However, these more sophisticated shrinkage policies are generally computationally more intensive. The soft-thresholding function is thus a compromise between a fast and straightforward method and a continuous approach.

4.2 Threshold selection

The main question in this procedure is how to choose the threshold, at each level and for each component. If this threshold is too small, the result is still noisy. On the other hand, a large threshold also removes important image features and thus causes a bias.

The optimal threshold for subband j $\delta_{j,\text{opt}}$ minimizes the mean square error of the result, as compared with the unknown, noise-free coefficients:

$$R_j^c(\delta) = \frac{1}{N_j^c} \|\mathbf{w}_{j,\delta}^c - \mathbf{v}_j^c\|^2, \quad (1)$$

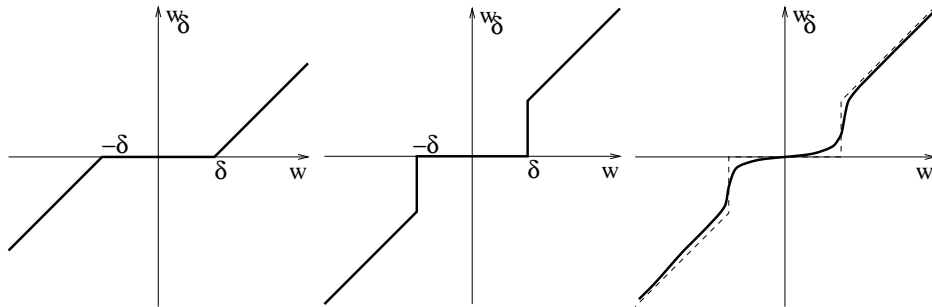


Figure 7: Noise reduction by wavelet shrinking. On the left: Soft-thresholding: a wavelet coefficient w with an absolute value below the threshold δ is replaced by 0. Coefficients with higher absolute values are shrunk. Middle: Hard-thresholding: Coefficients with an absolute value above the threshold are kept. Right: a more sophisticated shrinking function.

where \mathbf{v}_j^c is the vector of noise-free coefficients, $\mathbf{w}_{j,\delta}^c$ is the vector of shrunk coefficients for a given threshold δ , and N_j^c represents the number of wavelet coefficients at level j and component c .

The “universal threshold” by Donoho and Johnstone [6]:

$$\delta_j^c = \sqrt{2 \log(N_j^c)} \sigma_j^c, \quad (2)$$

is widely known and states that the optimal threshold value δ is proportional to the amount of noise. In fact, for finite N_j this is an upper bound for the optimal threshold. This choice is asymptotically optimal.

Other choices include the “SURE”-threshold [7] and Bayesian estimates [1, 12, 16]. In this paper, we use the method of generalized cross validation (GCV) [17, 18, 10].

This method minimizes at each level j the following function:

$$GCV_j^c(\delta) = \frac{\frac{1}{N_j^c} \|\mathbf{w}_j^c - \mathbf{w}_{j,\delta}^c\|^2}{\left[\frac{N_{j_0}^c}{N_j^c} \right]^2}, \quad (3)$$

where \mathbf{w}_j^c is the vector of wavelet coefficients at the resolution level j and for component c ; $\mathbf{w}_{j,\delta}^c$ is the vector of shrunk coefficients for a given threshold δ and $N_{j_0}^c$ is the number of zero elements in this vector. It is important to note that in this formula no estimate for the noise energy σ_j^c is needed. In [10, 9] this method is proved to be asymptotically optimal, i.e. for a large number of coefficients, the minimizer of $GCV_j^c(\delta)$ also minimizes the mean square error of the thresholded coefficients. For finite N_j it gives better estimates of the optimal threshold than the universal threshold. The mathematical description of the method in [10] assumes a continuous threshold operation. This is why hard-thresholding is not allowed. In principle, it also requires Gaussian noise. However, experiments show that other zero-mean density or probability functions mostly perform well.

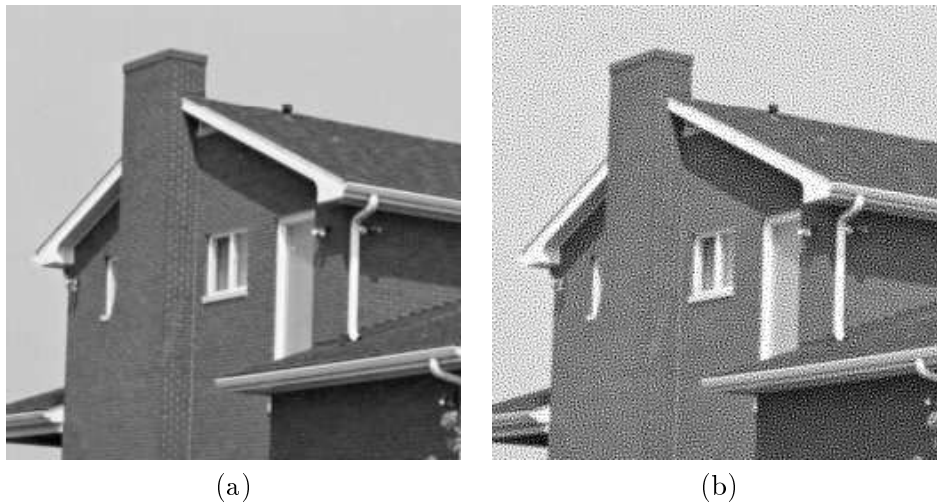


Figure 8: An artificial test example. (a) a clean test image. (b) The same image with artificial, additive and correlated noise. The noise is the result of a convolution of white noise with a FIR high pass filter. Signal-to-noise ratio is 4.98 dB.

5 Integer de-noising

5.1 Generalized cross validation for integer wavelet coefficients

Since the input for the algorithm of this paper are integers, the noise cannot take an arbitrary value and so its distribution cannot be Gaussian. Moreover, the integer wavelet transform is non-linear and so it does not preserve additivity nor stationarity. All these conditions are stricto sensu necessary for a correct use of a GCV-threshold estimation.

However, an artificial test example illustrates that, in practice, these conditions do not pose serious problems. Figure 8a shows a noise-free test image. In Figure 8b, we add artificial, colored noise. This noise was the result of a convolution of white noise with a FIR high pass-filter. The signal-to-noise ratio is 4.98 dB, where we define signal-to-noise ratio (*SNR*) as:

$$SNR = 20 \log_{10} \frac{\|\mathbf{f}\|}{\|\boldsymbol{\varepsilon}\|}, \quad (4)$$

with \mathbf{f} the noise-free image and $\boldsymbol{\varepsilon}$ the noise.

We compute the integer wavelet transform of the noisy and the noise-free image and estimate the optimal threshold at each resolution level and for each component by the GCV-procedure. Since we know the noise-free wavelet coefficients, we can compare the GCV-function with the mean square error as a function of the threshold. Figure 9a shows this comparison for one particular component at the one but finest resolution level. Both $GCV_j^\varepsilon(\delta)$ and $R_j^\varepsilon(\delta)$ have about the same minimum. Figure 9b is the result after thresholding and inverse transform. Signal-to-noise ratio is now 15.35 dB.

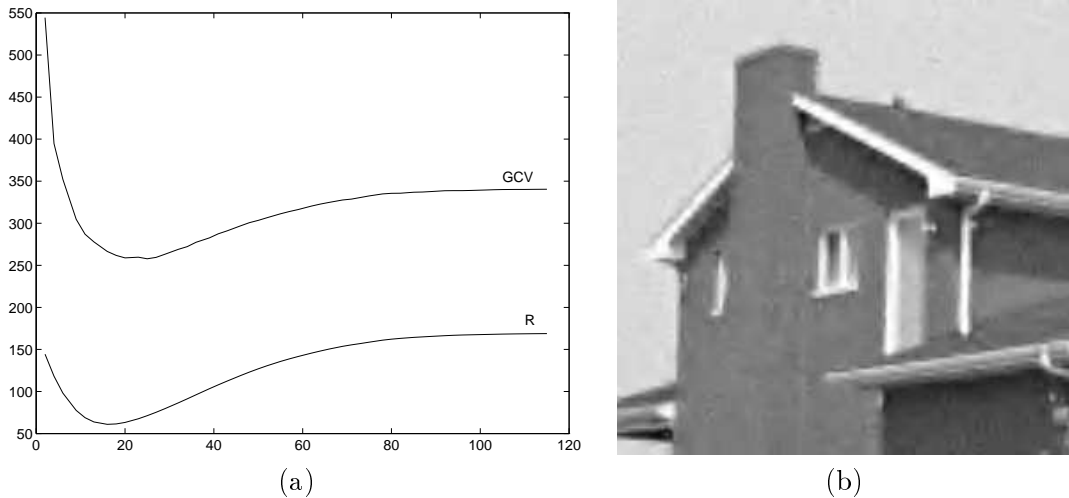


Figure 9: An artificial test example. (a) A comparison of $GCV_j^c(\delta)$ and $R_j^c(\delta)$ for one component at the one but finest resolution level. Both have about the same minimum. (b) The result after thresholding and inverse transform..

5.2 The minimization procedure

If we assume that, for continuously varying δ , $GCV_j^c(\delta)$ is a convex function, we can use a Fibonacci minimization procedure. For a given relative accuracy, this procedure only requires a limited and fixed number of function evaluations.

Since we are working with integers only, we only consider integer threshold values. Soft-thresholding with a non-integer value would yield non-integer wavelet coefficients. We could try to find the integer threshold that minimizes $GCV_j^c(\delta)$ among all integer δ . Since this is an absolute accuracy condition, the number of function evaluations would depend logarithmically on the largest possible threshold (the smallest being assumed to be zero), which is the largest wavelet coefficient at the given resolution level and for the given component.

However, in practice, $GCV_j^c(\delta)$ is not strictly convex. Figure 10 shows a detail of the GCV-function, depicted in Figure 9a. This is due to the jumps of the denominator: if, for a given wavelet coefficient w , the threshold value increases from $w - \varepsilon$ to $w + \varepsilon$, the value of N_0 in the denominator decreases at least by one. Note that the numerator is continuous since we use the continuous soft-threshold operation. Since most wavelet coefficients are small, most of the jumps of $GCV_j^c(\delta)$ appear at small threshold values. Experiments have shown that this lack of convexity mostly poses no problem for application of the GCV-procedure. We also remark that $GCV_j^c(\delta)$ is just an *estimate* of the mean square error as a function of δ . So, it is no use spending too much energy in finding the optimal threshold, if a fast algorithm finds a good approximation.

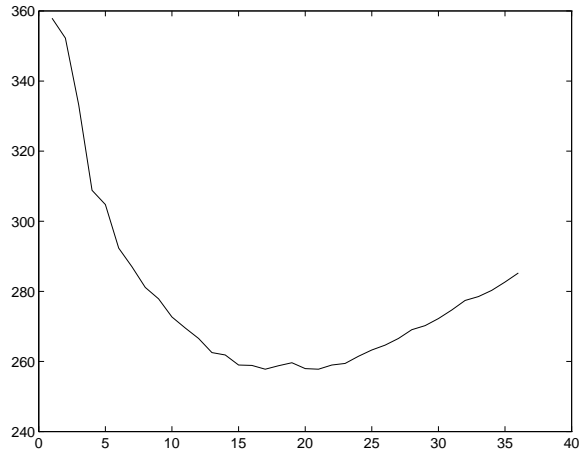


Figure 10: Detail of $GCV_j^c(\delta)$, depicted in Figure 9a. $GCV_j^c(\delta)$ is not strictly convex.

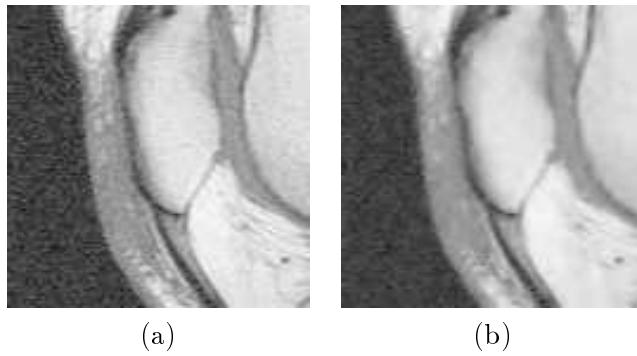


Figure 11: An example. (a) The input image, an MRI image (128×128 pixels) with noise. (b) Result after thresholding the wavelet coefficients at the first and second resolution level.

5.3 An example

We now illustrate the algorithm with an example. Figure 11a shows an MRI-image of 128 by 128 pixels. Figure 11b contains the result of the de-noising algorithm. We used biorthogonal Cohen-Daubechies-Feauveau (2,2)-wavelet filters [3]. Figure 12 shows the GCV-functions of the first (finest) and second resolution level. Since the GCV-procedure is only asymptotically optimal, it is no use applying it for components with only a small number of coefficients. Experiments learned that 1000 coefficients is about the minimum to guarantee a successful use of GCV. Therefore, we only shrink coefficients at the finest level (64×64 coefficients in each component) and at the second level (32×32 coefficients in each component).

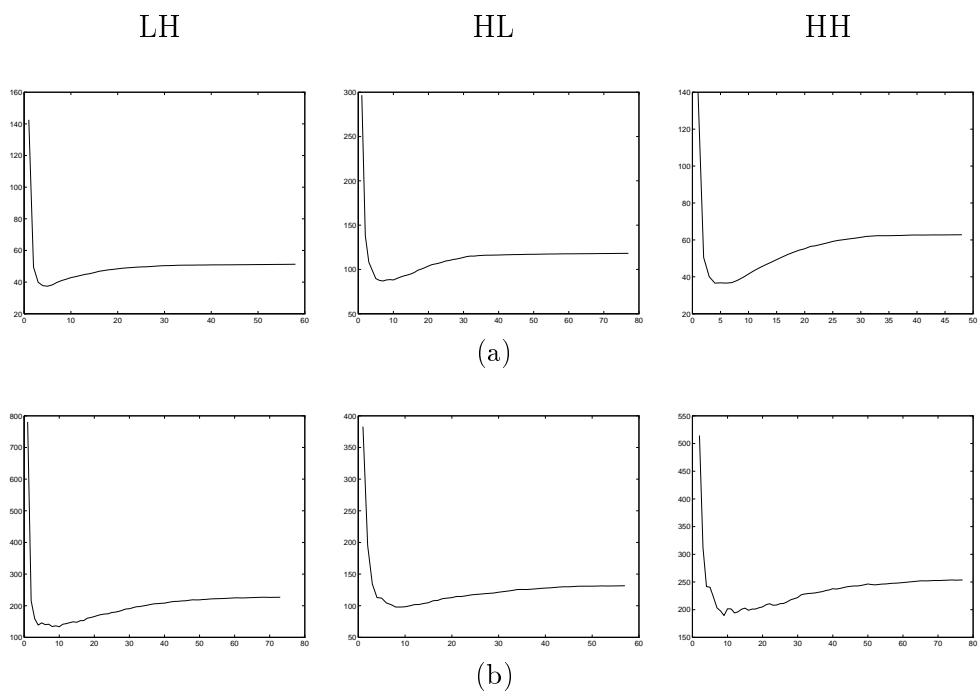


Figure 12: GCV-functions for wavelet transform of MRI-image of figure 11. (a) The three components at the finest resolution level. (b) The three components at the second resolution level.

6 Conclusions

We have presented an image de-noising algorithm, based on wavelet thresholding, that combines integer transforms with the fully automatic threshold selection method by generalized cross validation. Although the theoretical conditions for application of generalized cross validation are not strictly fulfilled in an integer transform framework, experiments show that the minimizer of the generalized cross validation function is a good estimate for the optimal threshold.

However, thresholding itself may cause artifacts in the result. To reduce these artifacts, we are now trying to use an integer version of the non-decimated wavelet transform. Such an overcomplete wavelet transform could introduce additional smoothing [8, 9].

Acknowledgements

The first author is financed by a grant from the Flemish Institute for the Promotion of Scientific and Technological Research in the Industry (IWT). Research of the second author is supported by the Flemish Information Technology Action Program ('Vlaams Actieprogramma Informatietechnologie'), project number ITA/950244. We used the software library WAILL, developed at K.U.Leuven, Department of Computer Science [15].

References

- [1] F. Abramovich, F. Sapatinas, and B. W. Silverman. Wavelet thresholding via a bayesian approach. *Journal of the Royal Statistical Society, Series B*, 58, 1997.
- [2] R. Calderbank, I. Daubechies, W. Sweldens, and B.-L. Yeo. Wavelet transforms that map integers to integers. Technical report, Department of Mathematics, Princeton University, 1996.
- [3] A. Cohen, I. Daubechiesr, and J. Feauveau. Bi-orthogonal bases of compactly supported wavelets. *Comm. Pure Appl. Math.*, 45:485–560, 1992.
- [4] I. Daubechies and W. Sweldens. Factoring wavelet transforms into lifting steps. Technical report, Bell Laboratories, Lucent Technologies, 1996.
- [5] D. L. Donoho. De-noising by soft-thresholding. *IEEE Transactions on Information Theory*, 41(3):613–627, May 1995.
- [6] D. L. Donoho and I. M. Johnstone. Ideal spatial adaptation via wavelet shrinkage. *Biometrika*, 81:425–455, 1994.
- [7] D. L. Donoho and I. M. Johnstone. Adapting to unknown smoothness via wavelet shrinkage. *J. Amer. Statist. Assoc.*, 90:1200–1224, 1995.
- [8] M. Jansen and A. Bultheel. Experiments with wavelet based image de-noising using generalized cross validation. In K. M. Hanson, editor, *Medical Imaging 1997: Image Processing*, volume 3034 of *SPIE Proceedings*, pages 206–214, February 1997.

- [9] M. Jansen and A. Bultheel. Multiple wavelet threshold estimation by generalized cross validation for images with correlated noise. Submitted for publication, February 1997.
- [10] M. Jansen, M. Malfait, and A. Bultheel. Generalized cross validation for wavelet thresholding. *Signal Processing*, 56(1):33–44, January 1997.
- [11] I. M. Johnstone and B. W. Silverman. Wavelet threshold estimators for data with correlated noise. *Journal of the Royal Statistical Society, Series B*, 59:319–351, 1997.
- [12] F. Ruggeri and B. Vidakovic. A bayesian decision theoretic approach to wavelet thresholding. Preprint 95-35, Duke University, Durham, NC, 1995.
- [13] W. Sweldens. The lifting scheme: A new philosophy in biorthogonal wavelet constructions. In A. F. Laine and M. Unser, editors, *Wavelet Applications in Signal and Image Processing III*, pages 68–79. Proc. SPIE 2569, 1995.
- [14] W. Sweldens. The lifting scheme: A custom-design construction of biorthogonal wavelets. *Appl. Comput. Harmon. Anal.*, 3(2):186–200, 1996.
- [15] G. Uytterhoeven, F. Van Wulpen, M. Jansen, D. Roose, and A. Bultheel. WAILI: Wavelets with Integer Lifting. TW Report 262, Department of Computer Science, Katholieke Universiteit Leuven, Belgium, July 1997.
- [16] B. Vidakovic. Nonlinear wavelet shrinkage with Bayes rules and Bayes factors. Preprint 94-24, Duke University, Durham, NC, 1994.
- [17] G. Wahba. *Spline Models for Observational Data*, chapter 4, pages 45–65. CBMS-NSF Regional Conf. Series in Appl. Math. Society for Industrial and Applied Mathematics, Philadelphia, PA, 1990.
- [18] N. Weyrich and G. T. Warhola. De-noising using wavelets and cross validation. In S.P. Singh, editor, *Approximation Theory, Wavelets and Applications*, volume 454 of *NATO ASI Series C*, pages 523–532, 1995.

Correspondence address:

Maarten Jansen
 K.U.Leuven
 Department of Computer Science
 Celestijnenlaan 200A
 B-3001 Heverlee
 BELGIUM
 Phone ++32 16 32 7080; Fax ++32 16 32 7996
 E-mail maarten.jansen@cs.kuleuven.ac.be
<http://www.cs.kuleuven.ac.be/~maarten/>