A WAVELETS INTRODUCTION[*]

STUDENT PAPER

Mark Fontenot
McNeese State University
P.O. Box 92340
Lake Charles, LA 70609
fontmark@earthlink.net

INTRODUCTION

Today's society is one of data transmission. Data is being transmitted between nearly everyone in nearly all societies. Methods include electronic format such as e-mail and the Internet, broadcast dissemination such as television and radio, and even non-technological means such as postal mail. Transmitting information in electronic format has been a topic of great discussion for many years. Many parts of the world have experienced some electronic transmission, be it from the days of the 1200 baud modem or the OC3 connections of major corporations. But as the bandwidth of data transmission lines has increased, so has the demand for it.

It can be argued that as bandwidth increases over time, the demand on it will increase at least at the same rate if not faster. Therefore, a need exists for some more efficient means of data transfer. One way to improve the transfer of data is to compress it before it is sent and then decompress it on the other end of the transmission. This is obviously not a new idea. It has been around for years, and many methods have been developed for this function. However, the degree of compression and the amount of data-loss are of much concern. This would indicate the need for a method combining a high compression ratio with a low (or even zero) amount of data-loss.

Wavelets are one answer to this need. Wavelets are mathematical functions that satisfy certain properties and can be used to transform one function representation into another. Transforms like this have been around for many years. However, choosing the appropriate wavelet transform for a particular problem can provide a high compression ratio with no data loss. Wavelet transforms have applications to image compression as well as a variety of other applications.

EXAMPLE WITH HAAR WAVELET

We will now perform a Haar wavelet transform on a sample data set, but later we will examine just a bit more theory. Consider the following string of numbers. For example, they could be the first row of an 8 x 8 pixel image. They could also be the functional values of a discrete signal. There is nothing significant about these particular numbers as they are simply chosen to avoid fractions in the calculations.

12 20 16 28 32 32 22 14

We will employ the process of *averaging* and *differencing* on this sample data. This process of averaging and differencing will occur three times due to the fact of there being eight elements and $8 = 2^3$. The formula for this operation will be $f(x) = (a+b)/2$. We will take the 8 numbers and group them into pairs of two adjacent numbers. Pair *one* will be (12 20), pair *two* will be (16 28), and so on. First we will take the averages of the numbers in each one of these pairs and place the four averages at the beginning of a new row. Then we will subtract the two numbers and divide by two. These numbers will be placed on the same row after the averages. The first four numbers in the second row are the averages and the last four are the differences (referred to as the detail elements). This process is continued on the remaining averages utill there is only one average left. Detail elements are carried down from row to row and the new detail elements are added in front of the old ones. Here are the 3 steps to arrive at the transformed row (the bottom row).

12	20	16	28	32	32	22	14
16	22	32	18	**-4**	**-6**	**0**	**4**
19	25	**-3**	**7**	**-4**	**-6**	**0**	**4**
22	**-3**	**-3**	**7**	**-4**	**-6**	**0**	**4**

In the bottom row, we now have only one *average* but we have seven detail elements. We can now choose some ξ to serve as a threshold. Any detail element that has an absolute value less than or equal to this threshold will be replaced with 0 (zero). The detail element with the smallest absolute value (greater than 0) is –3. If we choose $\xi = 3$, then our new row would be:

22 **0** **0** **7** **-4** **-6** **0** **4**

One nice thing about this transform is that it is easily reversible because it is simply arithmetic. In performing the inverse operation on this transformed row, we get:

18 26 16 28 29 29 11 19

Notice that some of the elements of the original string reappear, but some are obviously not the same. Of course these are sample data, and it seems that there is quite a bit of data loss involved here. This may be acceptable though. Now if we were to extend this process to a matrix, we would perform the above transform on each row and then each column of the matrix. This is equivalent to performing averaging and differencing on each row then performing the operations on each row of the transpose of the row-transformed matrix. If we were to perform these calculations on an 8 x 8 matrix, P, there would exist exactly one average element and 63 detail elements. This transformed matrix, M, would be called the *Haar transform of* P. (*Note*: The above method does not conserve the energy of the signal. Multiplying each average and difference by the square root of two will conserve energy. However, it is still possible to reverse this method to get the original signal values.)

After performing the Haar transform and then setting all detail elements less than or equal to the threshold to zero, very often a sparse matrix might occur. A sparse matrix is one that has a large number of zero-elements. It is obvious that a sparse matrix can take up considerably less amount of space (depending on the number of entries equal to zero) than the original matrix in which there were very few zero elements. Therefore this process (transform) could greatly reduce the amount of space needed to store and/or transmit an image or any other type of information that it could be applied to.

It should be noted however that in some cases the compression standard used requires lossless compression. This means that there is absolutely no data loss throughout the compression and decompression cycle. Lossless compression is employed in file compression formats such as the common zip format. Lossless compression is also quite important when compressing any medical image. Image reconstruction must be exact. This requirement is mostly a legal issue though. But many other applications do not require this amount of information retention. A great deal of data could be lost through the compression of an image, and the human eye would not be able to recognize any difference. Examine figure 1. If the pictures were not adjacent and resized, it would be very difficult to tell that one of them is a compressed image, and the other is the original.

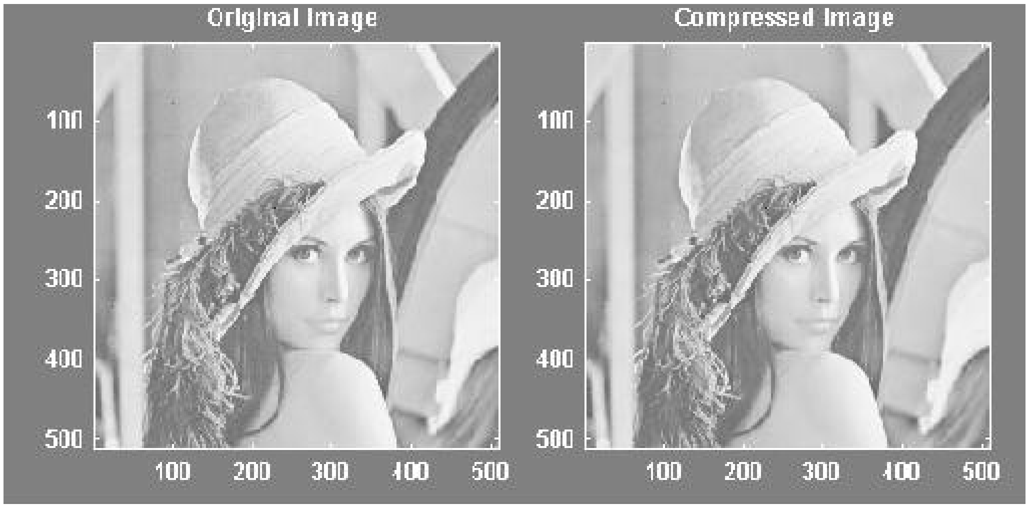

Figure 1. Example of 1-level Haar Wavelet Transform. Processed in Matlab

A TOUCH OF LINEAR ALGEBRA

If we consider the same string of numbers as we did above, we can use matrix multiplication to arrive at the same transformed vector. Our original string of numbers now defined as a vector is:

$$v = \begin{bmatrix} 12 & 20 & 16 & 28 & 32 & 32 & 22 & 14 \end{bmatrix}$$

To calculate the four *averages*, we will need *n/2* wavelet functions where *n* is the number of elements in v. They are in the form:

$$W_1^1 = \left[1/2, 1/2, 0, 0, \ldots, 0 \right]$$

$$W_2^1 = \left[0, 0, 1/2, 1/2, 0, 0, \ldots, 0 \right]$$

$$\vdots$$

$$W_{N/2}^1 = \left[0, 0, \ldots, 0, 1/2, 1/2 \right]$$

To transform v, we will use W_1^1through W_4^1. Thus to get the average (a_1) of the first two elements of v, we multiply it by W_1^1. To get the second average (a_2), v is multiplied by W_2^1, and so on until we have all the averages for our vector v. Generally,

$$a_p = \boldsymbol{v} \times \boldsymbol{W}_p^1 \text{, for } p = 1, 2, \ldots, N/2.$$

To calculate the detail elements of our image, the process is quite similar. The detail functions are:

$$D_1^1 = \left[1/2, -1/2, 0, 0, \ldots, 0 \right]$$

$$D_2^1 = \left[0, 0, 1/2, -1/2, 0, 0, \ldots, 0 \right]$$

$$\vdots$$

$$D_{N/2}^1 = \left[0, 0, \ldots, 0, 1/2, -1/2 \right]$$

In the same manner as the average elements, the first detail element is calculated by multiplying v by D_1^1, and so forth in the same manner. Generally,

$$d_p = \boldsymbol{v} \times \boldsymbol{D}_p^1 \text{ , for } p=1, 2, \ldots, N/2.$$

If we place all the ***W***'s and ***D***'s in the columns of an N x N matrix P, then all that is needed to perform the first level of the Haar transform is to multiply ***v*** by P. *Note*: This will not produce a vector that has only one average element with n - 1 detail elements (only a 1st level transform). It produces *n/2* averages and *n/2* differences.

APPLICATION OF WAVELETS

There are currently many applications where wavelets are used. Take, for instance, the United States Federal Bureau of Investigation's (FBI) collection of fingerprints. Currently, it around 50 million sets of fingerprints. It is estimated that 5,000 new submissions are made each day to this already enormous set. Each fingerprint is digitally scanned at a resolution of 512 pixels per inch. Each pixel will house one of a possible 256 shades of gray. Therefore, one fingerprint translates to about 700,000 pixels. Roughly 0.6 Mbytes of space is needed to store this digitized image. So, a pair of hands with a total of 10 fingers requires about 6 Mbytes of storage space.

It is estimated that digitizing the entire fingerprint library of the FBI would consume about 200 terabytes of space. Using the estimation of 5000 new submissions per day would result in an increase of 30 Gbytes per day. It is obviously reasonable to think that some kind of compression is needed. The FBI has adopted a standard of compression that employs wavelet compression. Currently, the FBI is able to achieve a compression ration of about 26:1. That would be a reduction from 30 Gbytes to less than 1.5 Gbytes of storage space required.

Wavelets are also useful in the arena of denoising noisy data. The idea is to use a averaging filter to obtain averages and detail elements. Then a thresholding procedure is applied which is similar to the above discussed process. Some threshold is chosen and all detail elements less than that threshold are set to zero. The inverse transform is then performed to retrieve the less noisy signal. Using wavelets in this context is quite useful because the wavelet transform retains all the important features while discarding the un-useful noise.

A company named Summus (http://www.summus.com) is currently involved in producing a plug-in for Netscape Navigator and Internet Explorer. The name of this plug-in is *Wavelet Image Viewer*. This plug-in is used in a method called progressive image transmission. What happens is that when an image is requested, a wavelet encoded copy is sent to the requestor. The overall average and larger detail coefficients are sent and then the smaller detail coefficients are sent. Eventually all the detail coefficients are sent resulting in the original image being displayed. The beauty lies in the fact that when the user is satisfied with the quality, the retrieval of more detail coefficients can be halted.

CONCLUSION

This paper has only touched the very tip of an enormous iceberg. Mathematicians have spent countless hours working out the gory details of the Haar transform as well as many other wavelet transforms. The author is by no means a mathematician and is interested more in the application and implementation of wavelets than the deep theory behind them. Numerous papers have been written on the subject of wavelets and there will definitely be many more in the future. However, one thing is for sure, wavelets will be incorporated into more and more applications and continue to emerge as the method of choice for many uses.

REFERENCES

Walker, James S. *A Primer on Wavelets and their Scientific Applications*. Chapman, New York. 1999.

Graps, Amara. "An Introduction to Wavelets". IEEE Computational Science and Engineering. Vol. 2, No. 2.

Mulcahy, Colm. "Image Compression Using the Haar Wavelet Transform". Spelman College Science & Mathematics Journal. Vol1, No1, Apr. 1997, 22-31.

Mulcahy, Colm. "Plotting and Scheming with Wavelets". Mathematics Magazine. Vol. 69, No.5, Dec. 1996, 323-343.