Chapter 5

Haar Bases, Haar Wavelets, Hadamard Matrices

In this chapter, we discuss two types of matrices that have applications in computer science and engineering:

(1) Haar matrices and the corresponding Haar wavelets, a fundamental tool in signal processing and computer graphics.

2) Hadamard matrices which have applications in error correcting codes, signal processing, and low rank approximation.

5.1 Introduction to Signal Compression Using Haar Wavelets

We begin by considering *Haar wavelets* in $\mathbb{R}^4$. Wavelets play an important role in audio and video signal processing, especially for *compressing* long signals into much smaller ones that still retain enough information so that when they are played, we can't see or hear any difference.

Consider the four vectors w_1, w_2, w_3, w_4 given by

$$w_1 = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \qquad w_2 = \begin{pmatrix} 1 \\ 1 \\ -1 \\ -1 \end{pmatrix} \qquad w_3 = \begin{pmatrix} 1 \\ -1 \\ 0 \\ 0 \end{pmatrix} \qquad w_4 = \begin{pmatrix} 0 \\ 0 \\ 1 \\ -1 \end{pmatrix}.$$

Note that these vectors are pairwise orthogonal, which means that their inner product is 0 (see Section 12.1, Example 12.1, and Section 12.2, Definition 12.2), so they are indeed linearly independent (see Proposition 12.4). Let $\mathcal{W} = \{w_1, w_2, w_3, w_4\}$ be the *Haar basis*, and let

$\mathcal{U} = \{e_1, e_2, e_3, e_4\}$ be the canonical basis of $\mathbb{R}^4$. The change of basis matrix $W = P_{\mathcal{W},\mathcal{U}}$ from $\mathcal{U}$ to $\mathcal{W}$ is given by

$$W = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & -1 & 0 \\ 1 & -1 & 0 & 1 \\ 1 & -1 & 0 & -1 \end{pmatrix},$$

and we easily find that the inverse of W is given by

$$W^{-1} = \begin{pmatrix} 1/4 & 0 & 0 & 0 \\ 0 & 1/4 & 0 & 0 \\ 0 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & 1/2 \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 \end{pmatrix}.$$

Observe that the second matrix in the above product is $W^\top$ and the first matrix in this product is $(W^\top W)^{-1}$. So the vector $v = (6, 4, 5, 1)$ over the basis $\mathcal{U}$ becomes $c = (c_1, c_2, c_3, c_4)$ over the Haar basis $\mathcal{W}$, with

$$\begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \end{pmatrix} = \begin{pmatrix} 1/4 & 0 & 0 & 0 \\ 0 & 1/4 & 0 & 0 \\ 0 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & 1/2 \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 \end{pmatrix} \begin{pmatrix} 6 \\ 4 \\ 5 \\ 1 \end{pmatrix} = \begin{pmatrix} 4 \\ 1 \\ 1 \\ 2 \end{pmatrix}.$$

Given a signal $v = (v_1, v_2, v_3, v_4)$, we first *transform* v into its coefficients $c = (c_1, c_2, c_3, c_4)$ over the Haar basis by computing $c = W^{-1}v$. Observe that

$$c_1 = \frac{v_1 + v_2 + v_3 + v_4}{4}$$

is the overall *average* value of the signal v. The coefficient c_1 corresponds to the background of the image (or of the sound). Then, c_2 gives the coarse details of v, whereas, c_3 gives the details in the first part of v, and c_4 gives the details in the second half of v.

Reconstruction of the signal consists in computing $v = Wc$. The trick for good *compression* is to throw away some of the coefficients of c (set them to zero), obtaining a *compressed signal* $\widehat{c}$, and still retain enough crucial information so that the reconstructed signal $\widehat{v} = W\widehat{c}$ looks almost as good as the original signal v. Thus, the steps are:

$$\text{input } v \longrightarrow \text{coefficients } c = W^{-1}v \longrightarrow \text{compressed } \widehat{c} \longrightarrow \text{compressed } \widehat{v} = W\widehat{c}.$$

This kind of compression scheme makes modern video conferencing possible.

It turns out that there is a faster way to find $c = W^{-1}v$, without actually using W^{-1}. This has to do with the multiscale nature of Haar wavelets.

Given the original signal $v = (6, 4, 5, 1)$ shown in Figure 5.1, we compute averages and half differences obtaining Figure 5.2. We get the coefficients $c_3 = 1$ and $c_4 = 2$. Then

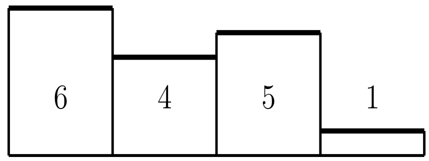

Figure 5.1: The original signal v.

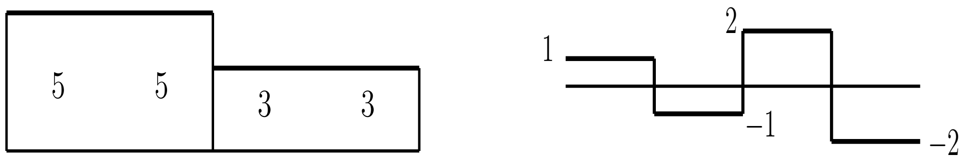

Figure 5.2: First averages and first half differences.

again we compute averages and half differences obtaining Figure 5.3. We get the coefficients $c_1 = 4$ and $c_2 = 1$. Note that the original signal v can be reconstructed from the two signals in Figure 5.2, and the signal on the left of Figure 5.2 can be reconstructed from the two signals in Figure 5.3. In particular, the data from Figure 5.2 gives us

$$
\begin{aligned}
5+1 &= \frac{v_1+v_2}{2} + \frac{v_1-v_2}{2} = v_1\\
5-1 &= \frac{v_1+v_2}{2} - \frac{v_1-v_2}{2} = v_2\\
3+2 &= \frac{v_3+v_4}{2} + \frac{v_3-v_4}{2} = v_3\\
3-2 &= \frac{v_3+v_4}{2} - \frac{v_3-v_4}{2} = v_4.
\end{aligned}
$$

5.2 Haar Bases and Haar Matrices, Scaling Properties of Haar Wavelets

The method discussed in Section 5.1 can be generalized to signals of any length 2^n. The previous case corresponds to $n = 2$. Let us consider the case $n = 3$. The *Haar basis*

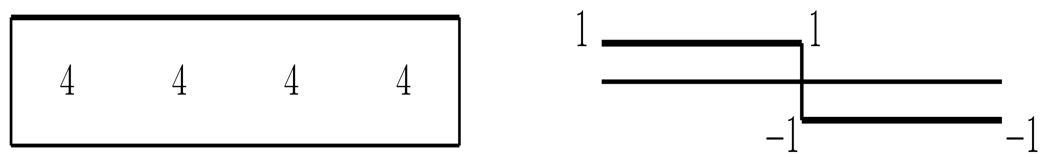

Figure 5.3: Second averages and second half differences.

$(w_1, w_2, w_3, w_4, w_5, w_6, w_7, w_8)$ is given by the matrix

$$W = \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & -1 & 0 & 0 & 0 \\ 1 & 1 & -1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & -1 & 0 & 0 & -1 & 0 & 0 \\ 1 & -1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & -1 & 0 & 1 & 0 & 0 & -1 & 0 \\ 1 & -1 & 0 & -1 & 0 & 0 & 0 & 1 \\ 1 & -1 & 0 & -1 & 0 & 0 & 0 & -1 \end{pmatrix}.$$

The columns of this matrix are orthogonal, and it is easy to see that

$$W^{-1} = \mathrm{diag}(1/8, 1/8, 1/4, 1/4, 1/2, 1/2, 1/2, 1/2)W^{\top}.$$

A pattern is beginning to emerge. It looks like the second Haar basis vector w_2 is the "mother" of all the other basis vectors, except the first, whose purpose is to perform averaging. Indeed, in general, given

$$w_2 = \underbrace{(1, \ldots, 1, -1, \ldots, -1)}_{2^n},$$

the other Haar basis vectors are obtained by a "scaling and shifting process." Starting from w_2, the scaling process generates the vectors

$$w_3, w_5, w_9, \ldots, w_{2^j+1}, \ldots, w_{2^{n-1}+1},$$

such that $w_{2^{j+1}+1}$ is obtained from w_{2^j+1} by forming two consecutive blocks of 1 and -1 of half the size of the blocks in w_{2^j+1}, and setting all other entries to zero. Observe that w_{2^j+1} has 2^j blocks of 2^{n-j} elements. The shifting process consists in shifting the blocks of 1 and -1 in w_{2^j+1} to the right by inserting a block of $(k-1)2^{n-j}$ zeros from the left, with $0 \leq j \leq n-1$ and $1 \leq k \leq 2^j$. Note that our convention is to use j as the scaling index and k as the shifting index. Thus, we obtain the following formula for w_{2^j+k}:

$$w_{2^j+k}(i) = \begin{cases} 0 & 1 \leq i \leq (k-1)2^{n-j} \\ 1 & (k-1)2^{n-j} + 1 \leq i \leq (k-1)2^{n-j} + 2^{n-j-1} \\ -1 & (k-1)2^{n-j} + 2^{n-j-1} + 1 \leq i \leq k2^{n-j} \\ 0 & k2^{n-j} + 1 \leq i \leq 2^n, \end{cases}$$

with $0 \leq j \leq n-1$ and $1 \leq k \leq 2^j$. Of course

$$w_1 = \underbrace{(1, \ldots, 1)}_{2^n}.$$

The above formulae look a little better if we change our indexing slightly by letting k vary from 0 to $2^j - 1$, and using the index j instead of 2^j.

Definition 5.1. The vectors of the *Haar basis* of dimension 2^n are denoted by

$$w_1, h_0^0, h_0^1, h_1^1, h_0^2, h_1^2, h_2^2, h_3^2, \ldots, h_k^j, \ldots, h_{2^{n-1}-1}^{n-1},$$

where

$$h_k^j(i) = \begin{cases} 0 & 1 \leq i \leq k2^{n-j} \\ 1 & k2^{n-j} + 1 \leq i \leq k2^{n-j} + 2^{n-j-1} \\ -1 & k2^{n-j} + 2^{n-j-1} + 1 \leq i \leq (k+1)2^{n-j} \\ 0 & (k+1)2^{n-j} + 1 \leq i \leq 2^n, \end{cases}$$

with $0 \leq j \leq n-1$ and $0 \leq k \leq 2^j - 1$. The $2^n \times 2^n$ matrix whose columns are the vectors

$$w_1, h_0^0, h_0^1, h_1^1, h_0^2, h_1^2, h_2^2, h_3^2, \ldots, h_k^j, \ldots, h_{2^{n-1}-1}^{n-1},$$

(in that order), is called the *Haar matrix* of dimension 2^n, and is denoted by W_n.

It turns out that there is a way to understand these formulae better if we interpret a vector $u = (u_1, \ldots, u_m)$ as a piecewise linear function over the interval $[0, 1)$.

Definition 5.2. Given a vector $u = (u_1, \ldots, u_m)$, the *piecewise linear function*[1] $\mathrm{plf}(u)$ is defined such that

$$\mathrm{plf}(u)(x) = u_i, \qquad \frac{i-1}{m} \leq x < \frac{i}{m},\ 1 \leq i \leq m.$$

In words, the function $\mathrm{plf}(u)$ has the value u_1 on the interval $[0, 1/m)$, the value u_2 on $[1/m, 2/m)$, *etc.*, and the value u_m on the interval $[(m-1)/m, 1)$.

For example, the piecewise linear function associated with the vector

$$u = (2.4, 2.2, 2.15, 2.05, 6.8, 2.8, -1.1, -1.3)$$

is shown in Figure 5.4.

Then each basis vector h_k^j corresponds to the function

$$\psi_k^j = \mathrm{plf}(h_k^j).$$

[1]Piecewise constant function might be a more accurate name.

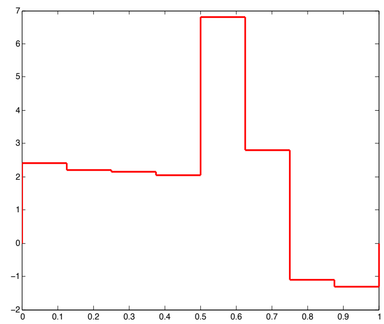

Figure 5.4: The piecewise linear function $\mathrm{plf}(u)$.

In particular, for all n, the Haar basis vectors

$$h_0^0 = w_2 = \underbrace{(1, \ldots, 1, -1, \ldots, -1)}_{2^n}$$

yield the same piecewise linear function ψ given by

$$\psi(x) = \begin{cases} 1 & \text{if} \quad 0 \leq x < 1/2 \\ -1 & \text{if} \quad 1/2 \leq x < 1 \\ 0 & \text{otherwise,} \end{cases}$$

whose graph is shown in Figure 5.5. It is easy to see that ψ_k^j is given by the simple expression

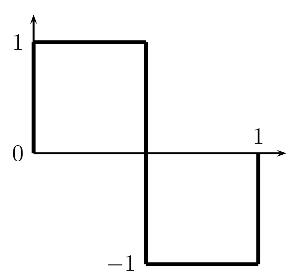

Figure 5.5: The Haar wavelet ψ.

$$\psi_k^j(x) = \psi(2^j x - k), \quad 0 \leq j \leq n-1,\ 0 \leq k \leq 2^j - 1.$$

The above formula makes it clear that ψ_k^j is obtained from ψ by scaling and shifting.

Definition 5.3. The function $\phi_0^0 = \mathrm{plf}(w_1)$ is the piecewise linear function with the constant value 1 on $[0, 1)$, and the functions $\psi_k^j = \mathrm{plf}(h_k^j)$ together with ϕ_0^0 are known as the *Haar wavelets*.

Rather than using W^{-1} to convert a vector u to a vector c of coefficients over the Haar basis, and the matrix W to reconstruct the vector u from its Haar coefficients c, we can use faster algorithms that use averaging and differencing.

If c is a vector of Haar coefficients of dimension 2^n, we compute the sequence of vectors $u^0, u^1, \ldots, u^n$ as follows:

$$\begin{aligned} u^0 &= c \\ u^{j+1} &= u^j \\ u^{j+1}(2i-1) &= u^j(i) + u^j(2^j + i) \\ u^{j+1}(2i) &= u^j(i) - u^j(2^j + i), \end{aligned}$$

for $j = 0, \ldots, n-1$ and $i = 1, \ldots, 2^j$. The reconstructed vector (signal) is $u = u^n$.

If u is a vector of dimension 2^n, we compute the sequence of vectors $c^n, c^{n-1}, \ldots, c^0$ as follows:

$$\begin{aligned} c^n &= u \\ c^j &= c^{j+1} \\ c^j(i) &= (c^{j+1}(2i-1) + c^{j+1}(2i))/2 \\ c^j(2^j + i) &= (c^{j+1}(2i-1) - c^{j+1}(2i))/2, \end{aligned}$$

for $j = n-1, \ldots, 0$ and $i = 1, \ldots, 2^j$. The vector over the Haar basis is $c = c^0$.

We leave it as an exercise to implement the above programs in `Matlab` using two variables u and c, and by building iteratively 2^j. Here is an example of the conversion of a vector to its Haar coefficients for $n = 3$.

Given the sequence $u = (31, 29, 23, 17, -6, -8, -2, -4)$, we get the sequence

$$\begin{aligned} c^3 &= (31, 29, 23, 17, -6, -8, -2, -4) \\ c^2 &= \left(\frac{31+29}{2}, \frac{23+17}{2}, \frac{-6-8}{2}, \frac{-2-4}{2}, \frac{31-29}{2}, \frac{23-17}{2}, \frac{-6-(-8)}{2}, \frac{-2-(-4)}{2}\right) \\ &= (30, 20, -7, -3, 1, 3, 1, 1) \\ c^1 &= \left(\frac{30+20}{2}, \frac{-7-3}{2}, \frac{30-20}{2}, \frac{-7-(-3)}{2}, 1, 3, 1, 1\right) \\ &= (25, -5, 5, -2, 1, 3, 1, 1) \\ c^0 &= \left(\frac{25-5}{2}, \frac{25-(-5)}{2}, 5, -2, 1, 3, 1, 1\right) = (10, 15, 5, -2, 1, 3, 1, 1) \end{aligned}$$

so $c = (10, 15, 5, -2, 1, 3, 1, 1)$. Conversely, given $c = (10, 15, 5, -2, 1, 3, 1, 1)$, we get the sequence

$$\begin{aligned}
u^0 &= (10, 15, 5, -2, 1, 3, 1, 1)\\
u^1 &= (10+15, 10-15, 5, -2, 1, 3, 1, 1) = (25, -5, 5, -2, 1, 3, 1, 1)\\
u^2 &= (25+5, 25-5, -5+(-2), -5-(-2), 1, 3, 1, 1) = (30, 20, -7, -3, 1, 3, 1, 1)\\
u^3 &= (30+1, 30-1, 20+3, 20-3, -7+1, -7-1, -3+1, -3-1)\\
&= (31, 29, 23, 17, -6, -8, -2, -4),
\end{aligned}$$

which gives back $u = (31, 29, 23, 17, -6, -8, -2, -4)$.

5.3 Kronecker Product Construction of Haar Matrices

There is another recursive method for constructing the Haar matrix W_n of dimension 2^n that makes it clearer why the columns of W_n are pairwise orthogonal, and why the above algorithms are indeed correct (which nobody seems to prove!). If we split W_n into two $2^n \times 2^{n-1}$ matrices, then the second matrix containing the last 2^{n-1} columns of W_n has a very simple structure: it consists of the vector

$$\underbrace{(1, -1, 0, \ldots, 0)}_{2^n}$$

and $2^{n-1} - 1$ shifted copies of it, as illustrated below for $n = 3$:

$$\begin{pmatrix}
1 & 0 & 0 & 0\\
-1 & 0 & 0 & 0\\
0 & 1 & 0 & 0\\
0 & -1 & 0 & 0\\
0 & 0 & 1 & 0\\
0 & 0 & -1 & 0\\
0 & 0 & 0 & 1\\
0 & 0 & 0 & -1
\end{pmatrix}.$$

Observe that this matrix can be obtained from the identity matrix $I_{2^{n-1}}$, in our example

$$I_4 = \begin{pmatrix}
1 & 0 & 0 & 0\\
0 & 1 & 0 & 0\\
0 & 0 & 1 & 0\\
0 & 0 & 0 & 1
\end{pmatrix},$$

by forming the $2^n \times 2^{n-1}$ matrix obtained by replacing each 1 by the column vector

$$\begin{pmatrix} 1\\ -1 \end{pmatrix}$$

and each zero by the column vector

$$\begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

Now the first half of W_n, that is the matrix consisting of the first 2^{n-1} columns of W_n, can be obtained from W_{n-1} by forming the $2^n \times 2^{n-1}$ matrix obtained by replacing each 1 by the column vector

$$\begin{pmatrix} 1 \\ 1 \end{pmatrix},$$

each -1 by the column vector

$$\begin{pmatrix} -1 \\ -1 \end{pmatrix},$$

and each zero by the column vector

$$\begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

For $n = 3$, the first half of W_3 is the matrix

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

which is indeed obtained from

$$W_2 = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & -1 & 0 \\ 1 & -1 & 0 & 1 \\ 1 & -1 & 0 & -1 \end{pmatrix}$$

using the process that we just described.

These matrix manipulations can be described conveniently using a product operation on matrices known as the Kronecker product.

Definition 5.4. Given a $m \times n$ matrix $A = (a_{ij})$ and a $p \times q$ matrix $B = (b_{ij})$, the *Kronecker product* (or *tensor product*) $A \otimes B$ of A and B is the $mp \times nq$ matrix

$$A \otimes B = \begin{pmatrix} a_{11}B & a_{12}B & \cdots & a_{1n}B \\ a_{21}B & a_{22}B & \cdots & a_{2n}B \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1}B & a_{m2}B & \cdots & a_{mn}B \end{pmatrix}.$$

It can be shown that $\otimes$ is associative and that

$$\begin{aligned}(A \otimes B)(C \otimes D) &= AC \otimes BD \\ (A \otimes B)^\top &= A^\top \otimes B^\top,\end{aligned}$$

whenever AC and BD are well defined. Then it is immediately verified that W_n is given by the following neat recursive equations:

$$W_n = \begin{pmatrix} W_{n-1} \otimes \begin{pmatrix} 1 \\ 1 \end{pmatrix} & I_{2^{n-1}} \otimes \begin{pmatrix} 1 \\ -1 \end{pmatrix} \end{pmatrix},$$

with $W_0 = (1)$. If we let

$$B_1 = 2 \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}$$

and for $n \geq 1$,

$$B_{n+1} = 2 \begin{pmatrix} B_n & 0 \\ 0 & I_{2^n} \end{pmatrix},$$

then it is not hard to use the Kronecker product formulation of W_n to obtain a rigorous proof of the equation

$$W_n^\top W_n = B_n, \quad \text{for all } n \geq 1.$$

The above equation offers a clean justification of the fact that the columns of W_n are pairwise orthogonal.

Observe that the right block (of size $2^n \times 2^{n-1}$) shows clearly how the detail coefficients in the second half of the vector c are added and subtracted to the entries in the first half of the partially reconstructed vector after $n - 1$ steps.

5.4 Multiresolution Signal Analysis with Haar Bases

An important and attractive feature of the Haar basis is that it provides a *multiresolution analysis* of a signal. Indeed, given a signal u, if $c = (c_1, \ldots, c_{2^n})$ is the vector of its Haar coefficients, the coefficients with low index give coarse information about u, and the coefficients with high index represent fine information. For example, if u is an audio signal corresponding to a Mozart concerto played by an orchestra, c_1 corresponds to the "background noise," c_2 to the bass, c_3 to the first cello, c_4 to the second cello, c_5, c_6, c_7, c_7 to the violas, then the violins, *etc.* This multiresolution feature of wavelets can be exploited to compress a signal, that is, to use fewer coefficients to represent it. Here is an example.

Consider the signal

$$u = (2.4, 2.2, 2.15, 2.05, 6.8, 2.8, -1.1, -1.3),$$

whose Haar transform is

$$c = (2, 0.2, 0.1, 3, 0.1, 0.05, 2, 0.1).$$

The piecewise-linear curves corresponding to u and c are shown in Figure 5.6. Since some of the coefficients in c are small (smaller than or equal to 0.2) we can compress c by replacing them by 0. We get

$$c_2 = (2, 0, 0, 3, 0, 0, 2, 0),$$

and the reconstructed signal is

$$u_2 = (2, 2, 2, 2, 7, 3, -1, -1).$$

The piecewise-linear curves corresponding to u_2 and c_2 are shown in Figure 5.7.

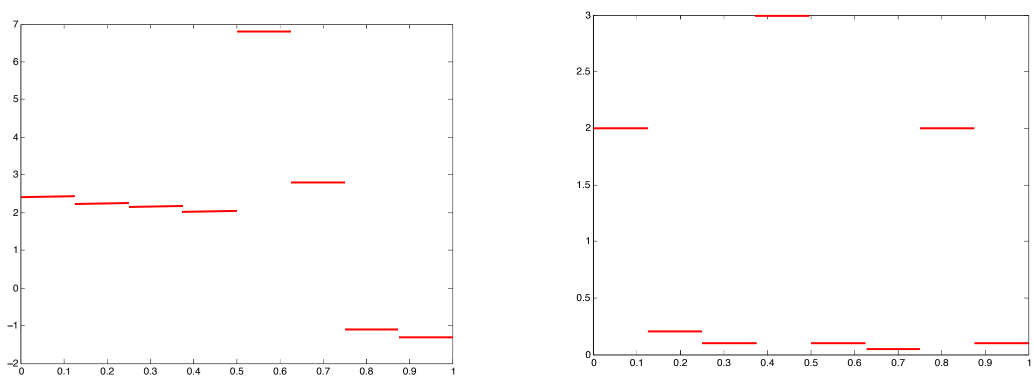

Figure 5.6: A signal and its Haar transform.

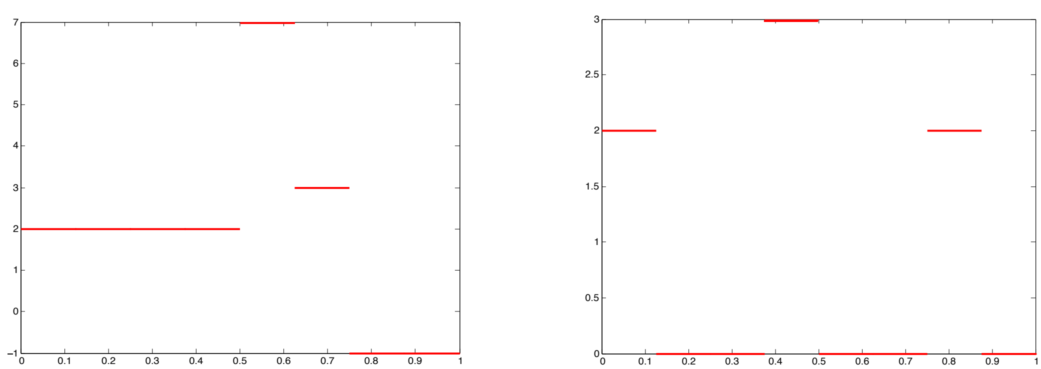

Figure 5.7: A compressed signal and its compressed Haar transform.

An interesting (and amusing) application of the Haar wavelets is to the compression of audio signals. It turns out that if your type `load handel` in `Matlab` an audio file will be loaded in a vector denoted by y, and if you type `sound(y)`, the computer will play this piece of music. You can convert y to its vector of Haar coefficients c. The length of y is 73113,

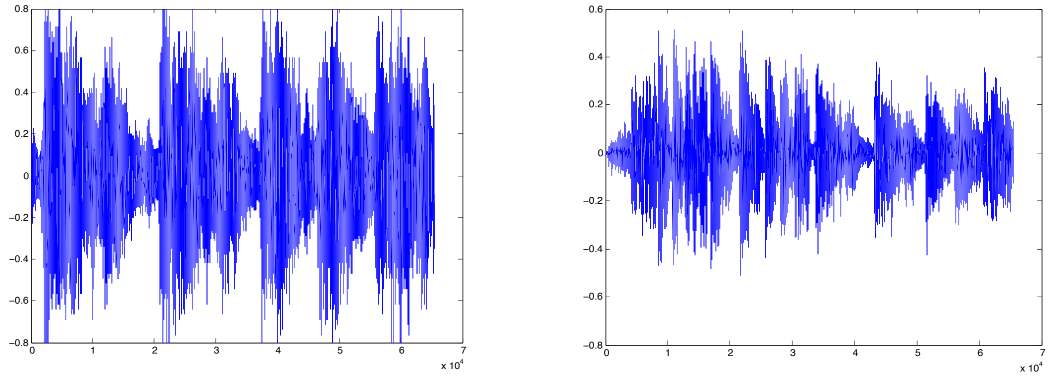

Figure 5.8: The signal “handel” and its Haar transform.

so first tuncate the tail of y to get a vector of length $65536 = 2^{16}$. A plot of the signals corresponding to y and c is shown in Figure 5.8. Then run a program that sets all coefficients of c whose absolute value is less that 0.05 to zero. This sets 37272 coefficients to 0. The resulting vector c_2 is converted to a signal y_2. A plot of the signals corresponding to y_2 and c_2 is shown in Figure 5.9. When you type `sound(y2)`, you find that the music doesn't differ

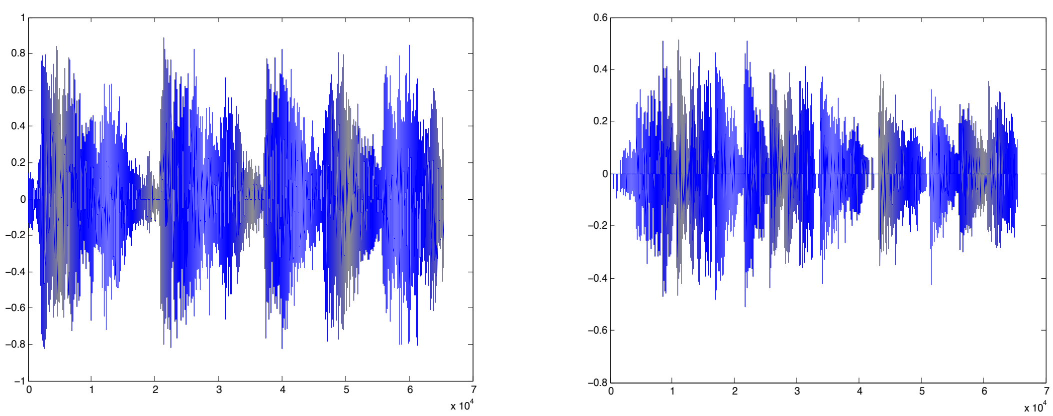

Figure 5.9: The compressed signal “handel” and its Haar transform.

much from the original, although it sounds less crisp. You should play with other numbers greater than or less than 0.05. You should hear what happens when you type `sound(c)`. It plays the music corresponding to the Haar transform c of y, and it is quite funny.

5.5 Haar Transform for Digital Images

Another neat property of the Haar transform is that it can be instantly generalized to matrices (even rectangular) without any extra effort! This allows for the compression of digital images. But first we address the issue of normalization of the Haar coefficients. As we observed earlier, the $2^n \times 2^n$ matrix W_n of Haar basis vectors has orthogonal columns, but its columns do not have unit length. As a consequence, $W_n^\top$ is not the inverse of W_n, but rather the matrix

$$W_n^{-1} = D_n W_n^\top$$

with $D_n = \mathrm{diag}\Big(2^{-n}, \underbrace{2^{-n}}_{2^0}, \underbrace{2^{-(n-1)}, 2^{-(n-1)}}_{2^1}, \underbrace{2^{-(n-2)}, \ldots, 2^{-(n-2)}}_{2^2}, \ldots, \underbrace{2^{-1}, \ldots, 2^{-1}}_{2^{n-1}}\Big)$.

Definition 5.5. The orthogonal matrix

$$H_n = W_n D_n^{\frac{1}{2}}$$

whose columns are the normalized Haar basis vectors, with

$$D_n^{\frac{1}{2}} = \mathrm{diag}\Big(2^{-\frac{n}{2}}, \underbrace{2^{-\frac{n}{2}}}_{2^0}, \underbrace{2^{-\frac{n-1}{2}}, 2^{-\frac{n-1}{2}}}_{2^1}, \underbrace{2^{-\frac{n-2}{2}}, \ldots, 2^{-\frac{n-2}{2}}}_{2^2}, \ldots, \underbrace{2^{-\frac{1}{2}}, \ldots, 2^{-\frac{1}{2}}}_{2^{n-1}}\Big)$$

is called the *normalized Haar transform matrix.* Given a vector (signal) u, we call $c = H_n^\top u$ the *normalized Haar coefficients* of u.

Because H_n is orthogonal, $H_n^{-1} = H_n^\top$.

Then a moment of reflection shows that we have to slightly modify the algorithms to compute $H_n^\top u$ and $H_n c$ as follows: When computing the sequence of u^js, use

$$\begin{aligned} u^{j+1}(2i-1) &= (u^j(i) + u^j(2^j+i))/\sqrt{2} \\ u^{j+1}(2i) &= (u^j(i) - u^j(2^j+i))/\sqrt{2}, \end{aligned}$$

and when computing the sequence of c^js, use

$$\begin{aligned} c^j(i) &= (c^{j+1}(2i-1) + c^{j+1}(2i))/\sqrt{2} \\ c^j(2^j+i) &= (c^{j+1}(2i-1) - c^{j+1}(2i))/\sqrt{2}. \end{aligned}$$

Note that things are now more symmetric, at the expense of a division by $\sqrt{2}$. However, for long vectors, it turns out that these algorithms are numerically more stable.

Remark: Some authors (for example, Stollnitz, Derose and Salesin [168]) rescale c by $1/\sqrt{2^n}$ and u by $\sqrt{2^n}$. This is because the norm of the basis functions ψ_k^j is not equal to 1 (under the inner product $\langle f, g\rangle = \int_0^1 f(t)g(t)dt)$. The normalized basis functions are the functions $\sqrt{2^j}\psi_k^j$.

Let us now explain the 2D version of the Haar transform. We describe the version using the matrix W_n, the method using H_n being identical (except that $H_n^{-1} = H_n^\top$, but this does not hold for W_n^{-1}). Given a $2^m \times 2^n$ matrix A, we can first convert the *rows* of A to their Haar coefficients using the Haar transform W_n^{-1}, obtaining a matrix B, and then convert the *columns* of B to their Haar coefficients, using the matrix W_m^{-1}. Because columns and rows are exchanged in the first step,

$$B = A(W_n^{-1})^\top,$$

and in the second step $C = W_m^{-1}B$, thus, we have

$$C = W_m^{-1}A(W_n^{-1})^\top = D_m W_m^\top A W_n\, D_n.$$

In the other direction, given a $2^m \times 2^n$ matrix C of Haar coefficients, we reconstruct the matrix A (the image) by first applying W_m to the columns of C, obtaining B, and then $W_n^\top$ to the rows of B. Therefore

$$A = W_m C W_n^\top.$$

Of course, we don't actually have to invert W_m and W_n and perform matrix multiplications. We just have to use our algorithms using averaging and differencing. Here is an example.

If the data matrix (the image) is the 8×8 matrix

$$A = \begin{pmatrix} 64 & 2 & 3 & 61 & 60 & 6 & 7 & 57 \\ 9 & 55 & 54 & 12 & 13 & 51 & 50 & 16 \\ 17 & 47 & 46 & 20 & 21 & 43 & 42 & 24 \\ 40 & 26 & 27 & 37 & 36 & 30 & 31 & 33 \\ 32 & 34 & 35 & 29 & 28 & 38 & 39 & 25 \\ 41 & 23 & 22 & 44 & 45 & 19 & 18 & 48 \\ 49 & 15 & 14 & 52 & 53 & 11 & 10 & 56 \\ 8 & 58 & 59 & 5 & 4 & 62 & 63 & 1 \end{pmatrix},$$

then applying our algorithms, we find that

$$C = \begin{pmatrix} 32.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4 & -4 & 4 & -4 \\ 0 & 0 & 0 & 0 & 4 & -4 & 4 & -4 \\ 0 & 0 & 0.5 & 0.5 & 27 & -25 & 23 & -21 \\ 0 & 0 & -0.5 & -0.5 & -11 & 9 & -7 & 5 \\ 0 & 0 & 0.5 & 0.5 & -5 & 7 & -9 & 11 \\ 0 & 0 & -0.5 & -0.5 & 21 & -23 & 25 & -27 \end{pmatrix}.$$

As we can see, C has more zero entries than A; it is a compressed version of A. We can further compress C by setting to 0 all entries of absolute value at most 0.5. Then we get

$$C_2 = \begin{pmatrix} 32.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4 & -4 & 4 & -4 \\ 0 & 0 & 0 & 0 & 4 & -4 & 4 & -4 \\ 0 & 0 & 0 & 0 & 27 & -25 & 23 & -21 \\ 0 & 0 & 0 & 0 & -11 & 9 & -7 & 5 \\ 0 & 0 & 0 & 0 & -5 & 7 & -9 & 11 \\ 0 & 0 & 0 & 0 & 21 & -23 & 25 & -27 \end{pmatrix}.$$

We find that the reconstructed image is

$$A_2 = \begin{pmatrix} 63.5 & 1.5 & 3.5 & 61.5 & 59.5 & 5.5 & 7.5 & 57.5 \\ 9.5 & 55.5 & 53.5 & 11.5 & 13.5 & 51.5 & 49.5 & 15.5 \\ 17.5 & 47.5 & 45.5 & 19.5 & 21.5 & 43.5 & 41.5 & 23.5 \\ 39.5 & 25.5 & 27.5 & 37.5 & 35.5 & 29.5 & 31.5 & 33.5 \\ 31.5 & 33.5 & 35.5 & 29.5 & 27.5 & 37.5 & 39.5 & 25.5 \\ 41.5 & 23.5 & 21.5 & 43.5 & 45.5 & 19.5 & 17.5 & 47.5 \\ 49.5 & 15.5 & 13.5 & 51.5 & 53.5 & 11.5 & 9.5 & 55.5 \\ 7.5 & 57.5 & 59.5 & 5.5 & 3.5 & 61.5 & 63.5 & 1.5 \end{pmatrix},$$

which is pretty close to the original image matrix A.

It turns out that `Matlab` has a wonderful command, `image(X)` (also `imagesc(X)`, which often does a better job), which displays the matrix X has an image in which each entry is shown as a little square whose gray level is proportional to the numerical value of that entry (lighter if the value is higher, darker if the value is closer to zero; negative values are treated as zero). The images corresponding to A and C are shown in Figure 5.10. The compressed images corresponding to A_2 and C_2 are shown in Figure 5.11. The compressed versions appear to be indistinguishable from the originals!

If we use the normalized matrices H_m and H_n, then the equations relating the image matrix A and its normalized Haar transform C are

$$\begin{aligned} C &= H_m^\top A H_n \\ A &= H_m C H_n^\top. \end{aligned}$$

The Haar transform can also be used to send large images progressively over the internet. Indeed, we can start sending the Haar coefficients of the matrix C starting from the coarsest coefficients (the first column from top down, then the second column, *etc.*), and at the receiving end we can start reconstructing the image as soon as we have received enough data.

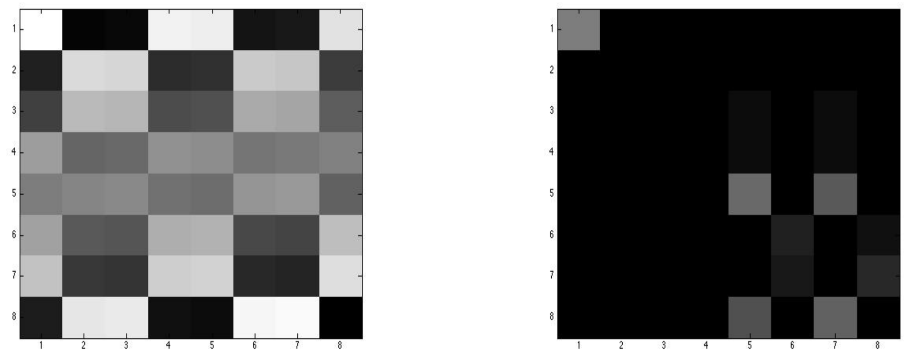

Figure 5.10: An image and its Haar transform.

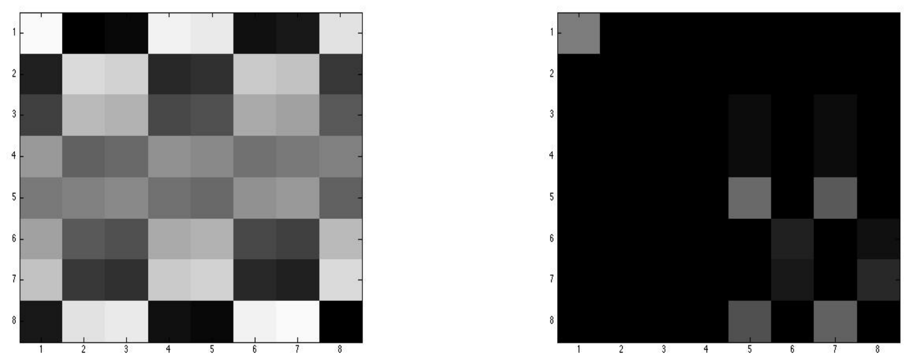

Figure 5.11: Compressed image and its Haar transform.

Observe that instead of performing all rounds of averaging and differencing on each row and each column, we can perform partial encoding (and decoding). For example, we can perform a single round of averaging and differencing for each row and each column. The result is an image consisting of four subimages, where the top left quarter is a coarser version of the original, and the rest (consisting of three pieces) contain the finest detail coefficients. We can also perform two rounds of averaging and differencing, or three rounds, *etc.* The second round of averaging and differencing is applied to the top left quarter of the image. Generally, the kth round is applied to the $2^{m+1-k} \times 2^{n+1-k}$ submatrix consisting of the first 2^{m+1-k} rows and the first 2^{n+1-k} columns $(1 \leq k \leq n)$ of the matrix obtained at the end of the previous round. This process is illustrated on the image shown in Figure 5.12. The result of performing one round, two rounds, three rounds, and nine rounds of averaging is shown in Figure 5.13. Since our images have size 512×512, nine rounds of averaging yields the Haar transform, displayed as the image on the bottom right. The original image has completely

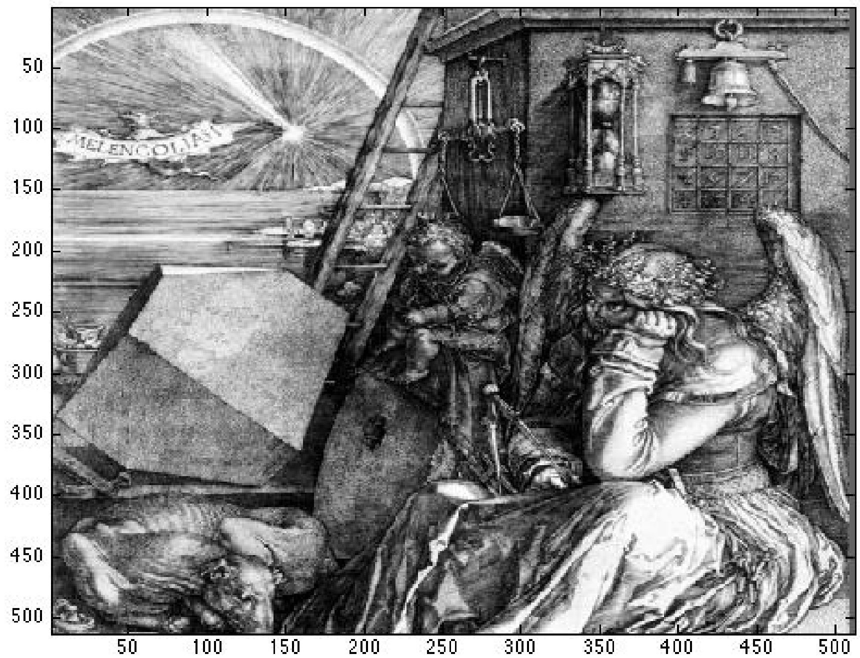

Figure 5.12: Original drawing by Durer.

disappeared! We leave it as a fun exercise to modify the algorithms involving averaging and differencing to perform k rounds of averaging/differencing. The reconstruction algorithm is a little tricky.

A nice and easily accessible account of wavelets and their uses in image processing and computer graphics can be found in Stollnitz, Derose and Salesin [168]. A very detailed account is given in Strang and and Nguyen [172], but this book assumes a fair amount of background in signal processing.

We can find easily a basis of $2^n \times 2^n = 2^{2n}$ vectors w_{ij} ($2^n \times 2^n$ matrices) for the linear map that reconstructs an image from its Haar coefficients, in the sense that for any $2^n \times 2^n$ matrix C of Haar coefficients, the image matrix A is given by

$$A = \sum_{i=1}^{2^n} \sum_{j=1}^{2^n} c_{ij} w_{ij}.$$

Indeed, the matrix w_{ij} is given by the so-called outer product

$$w_{ij} = w_i (w_j)^\top.$$

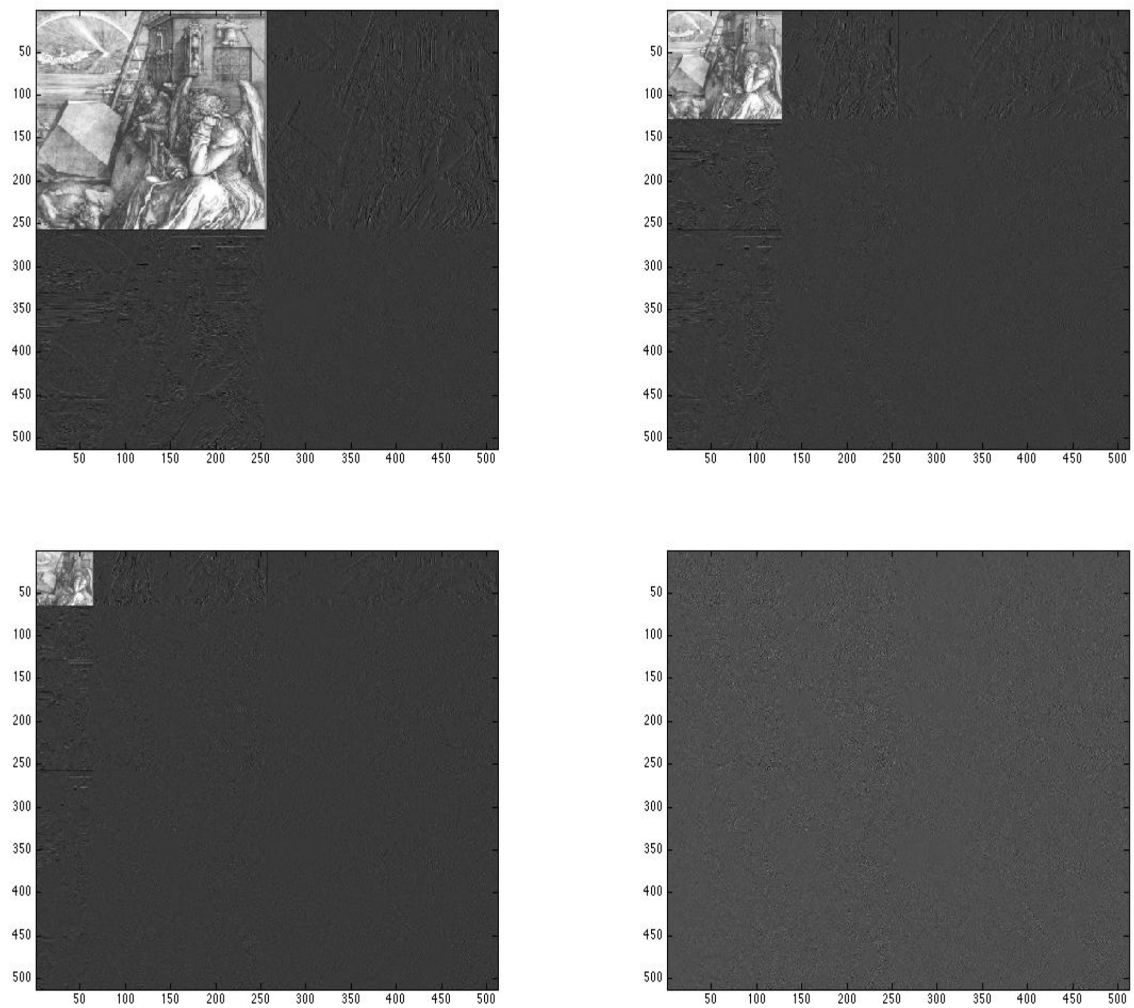

Figure 5.13: Haar tranforms after one, two, three, and nine rounds of averaging.

Similarly, there is a basis of $2^n \times 2^n = 2^{2n}$ vectors h_{ij} ($2^n \times 2^n$ matrices) for the 2D Haar transform, in the sense that for any $2^n \times 2^n$ matrix A, its matrix C of Haar coefficients is given by

$$C = \sum_{i=1}^{2^n} \sum_{j=1}^{2^n} a_{ij} h_{ij}.$$

If the columns of W^{-1} are $w'_1, \ldots, w'_{2^n}$, then

$$h_{ij} = w'_i (w'_j)^\top.$$

We leave it as exercise to compute the bases (w_{ij}) and (h_{ij}) for $n = 2$, and to display the corresponding images using the command `imagesc`.

5.6 Hadamard Matrices

There is another famous family of matrices somewhat similar to Haar matrices, but these matrices have entries $+1$ and -1 (no zero entries).

Definition 5.6. A real $n \times n$ matrix H is a *Hadamard matrix* if $h_{ij} = \pm 1$ for all i, j such that $1 \leq i, j \leq n$ and if

$$H^\top H = nI_n.$$

Thus the columns of a Hadamard matrix are pairwise orthogonal. Because H is a square matrix, the equation $H^\top H = nI_n$ shows that H is invertible, so we also have $HH^\top = nI_n$. The following matrices are example of Hadamard matrices:

$$H_2 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad H_4 = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix},$$

and

$$H_8 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{pmatrix}.$$

A natural question is to determine the positive integers n for which a Hadamard matrix of dimension n exists, but surprisingly this is an *open problem*. The *Hadamard conjecture* is that for every positive integer of the form $n = 4k$, there is a Hadamard matrix of dimension n.

What is known is a necessary condition and various sufficient conditions.

Theorem 5.1. *If H is an $n \times n$ Hadamard matrix, then either $n = 1, 2$, or $n = 4k$ for some positive integer k.*

Sylvester introduced a family of Hadamard matrices and proved that there are Hadamard matrices of dimension $n = 2^m$ for all $m \geq 1$ using the following construction.

Proposition 5.2. *(Sylvester, 1867) If H is a Hadamard matrix of dimension n, then the block matrix of dimension $2n$,*

$$\begin{pmatrix} H & H \\ H & -H \end{pmatrix},$$

is a Hadamard matrix.

If we start with

$$H_2 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix},$$

we obtain an infinite family of symmetric Hadamard matrices usually called *Sylvester–Hadamard* matrices and denoted by H_{2^m}. The Sylvester–Hadamard matrices H_2, H_4 and H_8 are shown on the previous page.

In 1893, Hadamard gave examples of Hadamard matrices for $n = 12$ and $n = 20$. At the present, Hadamard matrices are known for all $n = 4k \leq 1000$, *except for* $n = 668, 716$, *and* 892.

Hadamard matrices have various applications to error correcting codes, signal processing, and numerical linear algebra; see Seberry, Wysocki and Wysocki [154] and Tropp [177]. For example, there is a code based on H_{32} that can correct 7 errors in any 32-bit encoded block, and can detect an eighth. This code was used on a Mariner spacecraft in 1969 to transmit pictures back to the earth.

For every $m \geq 0$, the piecewise affine functions $\mathrm{plf}((H_{2^m})_i)$ associated with the 2^m rows of the Sylvester–Hadamard matrix H_{2^m} are functions on $[0, 1]$ known as the *Walsh functions*. It is customary to index these 2^m functions by the integers $0, 1, \ldots, 2^m - 1$ in such a way that the Walsh function $\mathrm{Wal}(k, t)$ is equal to the function $\mathrm{plf}((H_{2^m})_i)$ associated with the Row i of H_{2^m} that contains k changes of signs between consecutive groups of $+1$ and consecutive groups of -1. For example, the fifth row of H_8, namely

$$\begin{pmatrix} 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \end{pmatrix},$$

has five consecutive blocks of $+1$s and -1s, four sign changes between these blocks, and thus is associated with $\mathrm{Wal}(4, t)$. In particular, Walsh functions corresponding to the rows of H_8 (from top down) are:

$$\mathrm{Wal}(0, t),\ \mathrm{Wal}(7, t),\ \mathrm{Wal}(3, t),\ \mathrm{Wal}(4, t),\ \mathrm{Wal}(1, t),\ \mathrm{Wal}(6, t),\ \mathrm{Wal}(2, t),\ \mathrm{Wal}(5, t).$$

Because of the connection between Sylvester–Hadamard matrices and Walsh functions, Sylvester–Hadamard matrices are called *Walsh–Hadamard matrices* by some authors. For every m, the 2^m Walsh functions are pairwise orthogonal. The countable set of Walsh functions $\mathrm{Wal}(k,t)$ for all $m \geq 0$ and all k such that $0 \leq k \leq 2^m - 1$ can be ordered in such a way that it is an orthogonal Hilbert basis of the Hilbert space $\mathrm{L}^2([0,1)]$; see Seberry, Wysocki and Wysocki [154].

The Sylvester–Hadamard matrix H_{2^m} plays a role in various algorithms for dimension reduction and low-rank matrix approximation. There is a type of structured dimension-reduction map known as the *subsampled randomized Hadamard transform*, for short SRHT; see Tropp [177] and Halko, Martinsson and Tropp [86]. For $\ell \ll n = 2^m$, an *SRHT matrix* is an $\ell \times n$ matrix of the form

$$\Phi = \sqrt{\frac{n}{\ell}} RHD,$$

where

1. D is a random $n \times n$ diagonal matrix whose entries are independent random signs.

2. $H = n^{-1/2} H_n$, a normalized Sylvester–Hadamard matrix of dimension n.

3. R is a random $\ell \times n$ matrix that restricts an n-dimensional vector to ℓ coordinates, chosen uniformly at random.

It is explained in Tropp [177] that for any input x such that $\|x\|_2 = 1$, the probability that $|(HDx)_i| \geq \sqrt{n^{-1}\log(n)}$ for any i is quite small. Thus HD has the effect of "flattening" the input x. The main result about the SRHT is that it preserves the geometry of an entire subspace of vectors; see Tropp [177] (Theorem 1.3).

5.7 Summary

The main concepts and results of this chapter are listed below:

- Haar basis vectors and a glimpse at *Haar wavelets*.

- *Kronecker product* (or *tensor product*) of matrices.

- Hadamard and Sylvester–Hadamard matrices.

- Walsh functions.

5.8 Problems

Problem 5.1. (Haar extravaganza) Consider the matrix

$$W_{3,3} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 \end{pmatrix}.$$

(1) Show that given any vector $c = (c_1, c_2, c_3, c_4, c_5, c_6, c_7, c_8)$, the result $W_{3,3}c$ of applying $W_{3,3}$ to c is

$$W_{3,3}c = (c_1 + c_5, c_1 - c_5, c_2 + c_6, c_2 - c_6, c_3 + c_7, c_3 - c_7, c_4 + c_8, c_4 - c_8),$$

the last step in reconstructing a vector from its Haar coefficients.

(2) Prove that the inverse of $W_{3,3}$ is $(1/2)W_{3,3}^{\top}$. Prove that the columns and the rows of $W_{3,3}$ are orthogonal.

(3) Let $W_{3,2}$ and $W_{3,1}$ be the following matrices:

$$W_{3,2} = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \quad W_{3,1} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

Show that given any vector $c = (c_1, c_2, c_3, c_4, c_5, c_6, c_7, c_8)$, the result $W_{3,2}c$ of applying $W_{3,2}$ to c is

$$W_{3,2}c = (c_1 + c_3, c_1 - c_3, c_2 + c_4, c_2 - c_4, c_5, c_6, c_7, c_8),$$

the second step in reconstructing a vector from its Haar coefficients, and the result $W_{3,1}c$ of applying $W_{3,1}$ to c is

$$W_{3,1}c = (c_1 + c_2, c_1 - c_2, c_3, c_4, c_5, c_6, c_7, c_8),$$

the first step in reconstructing a vector from its Haar coefficients.

Conclude that

$$W_{3,3}W_{3,2}W_{3,1} = W_3,$$