Walsh-Hadamard Transform

Hadamard Matrix

The *Kronecker product* of two matrices $A = [a_{ij}]_{m\times n}$ and $B = [b_{ij}]_{k\times l}$ is defined as

$$A \otimes B \triangleq \begin{bmatrix} a_{11}B & \cdots & a_{1n}B \\ \cdots & \cdots & \cdots \\ a_{m1}B & \cdots & a_{mn}B \end{bmatrix}_{mk\times nl}$$

In general, $A \otimes B \neq B \otimes A$.

The *Hadamard Matrix* is defined recursively as below:

$$H_1 \triangleq \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

$$H_n = H_1 \otimes H_{n-1} = \begin{bmatrix} H_{n-1} & H_{n-1} \\ H_{n-1} & -H_{n-1} \end{bmatrix}$$

For example,

$$H_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} H_1 & H_1 \\ H_1 & -H_1 \end{bmatrix} = \frac{1}{\sqrt{4}} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}$$

$$H_3 = \frac{1}{\sqrt{2}} \begin{bmatrix} H_2 & H_2 \\ H_2 & -H_2 \end{bmatrix} = \frac{1}{\sqrt{8}} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix} \begin{array}{cc} 0 & 0 \\ 1 & 7 \\ 2 & 3 \\ 3 & 4 \\ 4 & 1 \\ 5 & 6 \\ 6 & 2 \\ 7 & 5 \end{array}$$

The first column following the array is the index numbers of the $N = 8$ rows, and the second column represents the *sequency* — the number of zero-crossings (sign changes) in each row.

Sequency is similar to, but different from, frequency in the sense that it measures the rate of change of non-periodical signals.

The Hadamard matrix can also be obtained by defining its element in the kth row and lth column of H $(k, m = 0, 1, \cdots, N-1)$ as

$$h(k,m) = (-1)^{\sum_{i=0}^{n-1} k_i m_i} = \prod_{i=0}^{n-1} (-1)^{k_i m_i} = h(m,k)$$

where

$$k = \sum_{i=0}^{n-1} k_i 2^i = (k_{n-1} k_{n-2} \cdots k_1 k_0)_2 \quad (k_i = 0, 1)$$

$$m = \sum_{i=0}^{n-1} m_i 2^i = (m_{n-1} m_{n-2} \cdots m_1 m_0)_2 \quad (m_i = 0, 1)$$

i.e., $(k_{n-1} k_{n-2} \cdots k_1 k_0)_2$ and $(m_{n-1} m_{n-2} \cdots m_1 m_0)_2$ are the binary representations of k and m, respectively. Obviously, $n = log_2 N$.

H is real, symmetric, and orthogonal:

$$H = H^* = H^T = H^{-1}$$

Fast Walsh-Hadamard Transform (Hadamard Ordered)

As any orthogonal (unitary) matrix can be used to define an orthogonal (unitary) transform, we define a Walsh-Hadamard transform of Hadamard order (WHT_h) as

$$\begin{cases} \overline{X} = H\overline{x} \\ \overline{x} = H\overline{X} \end{cases}$$

These are the forward and inverse WHT_h transform pair.
Here $\overline{x} = [x(0), x(1), \cdots, x(N-1)]^T$ and $\overline{X} = [X(0), X(1), \cdots, X(N-1)]^T$ are the signal and spectrum vectors, respectively. The kth element of the transform can also be written as

$$X(k) = \sum_{m=0}^{N-1} x(m) Wal_h(m,k) = \sum_{m=0}^{N-1} x(m) \prod_{i=0}^{n-1} (-1)^{m_i k_i}$$

The complexity of WHT is $O(N^2)$. Similar to FFT algorithm, we can derive a fast WHT algorithm with complexity of $O(N log_2 N)$. We will assume $n = 3$ and $N = 2^n = 8$ in the following derivation. An $N = 8$ point WHT_h of signal $x(m)$ is defined as

$$\begin{bmatrix} X(0) \\ . \\ X(3) \\ X(4) \\ . \\ X(7) \end{bmatrix} = \begin{bmatrix} H_2 & H_2 \\ H_2 & -H_2 \end{bmatrix} \begin{bmatrix} x(0) \\ . \\ x(3) \\ x(4) \\ . \\ x(7) \end{bmatrix}$$

This equation can be separated into two parts. The first half of the X vector can be obtained as

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = H_2 \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix} + H_2 \begin{bmatrix} x(4) \\ x(5) \\ x(6) \\ x(7) \end{bmatrix} = H_2 \begin{bmatrix} x_1(0) \\ x_1(1) \\ x_1(2) \\ x_1(3) \end{bmatrix} \tag{1}$$

where

$$x_1(i) \triangleq x(i) + x(i+4) \quad (i = 0, \cdots, 3) \tag{2}$$

The second half of the X vector can be obtained as

$$\begin{bmatrix} X(4) \\ X(5) \\ X(6) \\ X(7) \end{bmatrix} = H_2 \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix} - H_2 \begin{bmatrix} x(4) \\ x(5) \\ x(6) \\ x(7) \end{bmatrix} = H_2 \begin{bmatrix} x_1(4) \\ x_1(5) \\ x_1(6) \\ x_1(7) \end{bmatrix} \tag{3}$$

where

$$x_1(i+4) \stackrel{\triangle}{=} x(i) - x(i+4) \quad (i = 0, \cdots, 3) \tag{4}$$

What we have done is converting a WHT of size $N = 8$ into two $WHTs$ of size $N/2 = 4$. Continuing this process recursively, we can rewrite Eq. (1) as the following (similar process for Eq. (3))

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} H_1 & H_1 \\ H_1 & -H_1 \end{bmatrix} \begin{bmatrix} x_1(0) \\ x_1(1) \\ x_1(2) \\ x_1(3) \end{bmatrix}$$

This equation can again be separated into two halves. The first half is

$$\begin{bmatrix} X(0) \\ X(1) \end{bmatrix} = H_1 \begin{bmatrix} x_1(0) \\ x_1(1) \end{bmatrix} + H_1 \begin{bmatrix} x_1(2) \\ x_1(3) \end{bmatrix} = H_1 \begin{bmatrix} x_2(0) \\ x_2(1) \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} x_2(0) \\ x_2(1) \end{bmatrix} \tag{5}$$

where

$$x_2(i) \stackrel{\triangle}{=} x_1(i) + x_1(i+2) \quad (i = 0, 1) \tag{6}$$

The second half is

$$\begin{bmatrix} X(2) \\ X(3) \end{bmatrix} = H_1 \begin{bmatrix} x_1(0) \\ x_1(1) \end{bmatrix} - H_1 \begin{bmatrix} x_1(2) \\ x_1(3) \end{bmatrix} = H_1 \begin{bmatrix} x_2(2) \\ x_2(3) \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} x_2(2) \\ x_2(3) \end{bmatrix} \tag{7}$$

where

$$x_2(i+2) \stackrel{\triangle}{=} x_1(i) - x_1(i+2) \quad (i = 0, 1) \tag{8}$$

Finally, from Eq. (5) we get (similar process for Eq. (7))

$$X(0) = x_2(0) + x_2(1) \tag{9}$$

and

$$X(1) = x_2(0) - x_2(1) \tag{10}$$

Summarizing the above steps of Equations (2), (4), (6), (8), (9) and (10), we get the Fast WHT algorithm as illustrated below.

Fast Walsh-Hadamard Transform (Sequency Ordered)

In order for the elements in the spectrum $\overline{X} = [X(0), X(1), \cdots, X(N-1)]^T$ to represent different sequency components contained in the signal in a low-to-high order, we can re-order the rows (or columns) of the Hadamard matrix H according to their sequencies.

To convert a given sequency number s into the corresponding index number k in Hadamard order, we need to

1. represent s in binary form:

$$s = (s_{n-1}s_{n-2}\cdots s_1 s_0)_2 = \sum_{i=0}^{n-1} s_i 2^i$$

2. convert this binary form to Gray code:

$$g_i = s_i \oplus s_{i+1} \quad (i = 0, \cdots, n-1)$$

   where $\oplus$ represents exclusive or and $s_n \stackrel{\triangle}{=} 0$.

3. bit-reverse g_i's to get k_i's:

$$k_i = g_{n-1-i} = s_{n-1-i} \oplus s_{n-i}$$

Now k can be found as

$$k = (k_{n-1}\, k_{n-2} \cdots k_1\, k_0)_2 = \sum_{i=0}^{n-1} s_{n-1-i} \oplus s_{n-i} 2^i = \sum_{j=0}^{n-1} s_j \oplus s_{j+1} 2^{n-1-j}$$

where $j = n-1-i$ is the index of the summation, which is just a symbol and can be replace by, say, i.

For example, $n = log_2 N = log_2 8 = 3$, we have

s	0	1	2	3	4	5	6	7
binary	000	001	010	011	100	101	110	111
Gray code	000	001	011	010	110	111	101	100
bit-reverse	000	100	110	010	011	111	101	001
k	0	4	6	2	3	7	5	1

The sequency ordered Walsh-Hadamard transform (WHT_w, also called Walsh ordered WHT) can be obtained by first carrying out the fast WHT_h and then reordering the components of $\overline{X}$ as shown above. Alternatively, we can use the following fast WHT_w directly with better efficiency.

The sequency ordered WHT of $x(m)$ can also be defined as

$$X(k) = \sum_{m=0}^{N-1} x(m) Wal_w(m,k) = \sum_{m=0}^{N-1} x(m) \prod_{i=0}^{n-1} (-1)^{(k_i+k_{i+1})m_{n-1-i}}$$

where $N = 2^n$, $k_n \stackrel{\triangle}{=} 0$, and the exponent of -1 represents the conversion from sequency ordering to Hadamard ordering (binary-to-Gray code conversion and bit-reversal conversion).

In the following, we assume $n = 3$, $\ N = 2^3 = 8$, and we represent m and k in binary form as, respectively, $(m_2m_1m_0)_2$ and $(k_2k_1k_0)_2$, i.e.,

$$m = \sum_{i=0}^{n-1} m_i 2^i = 4m_2 + 2m_1 + m_0 \qquad (m_i = 0, 1)$$

$$k = \sum_{i=0}^{n-1} k_i 2^i = 4k_2 + 2k_1 + k_0 \qquad (k_i = 0, 1)$$

As the first step of the algorithm, we rearrange the order of the samples $x(m)$ by bit-reversal to get

$$x_0(4m_0 + 2m_1 + m_2) \stackrel{\triangle}{=} x(4m_2 + 2m_1 + m_0) \qquad m = 0, 1, \cdots 7$$

We also define $l_i = m_{n-1-i}$. Now the WHT_w can be written as

$$\begin{aligned} X(k) &= \sum_{m_2=0}^{1} \sum_{m_1=0}^{1} \sum_{m_0=0}^{1} x_0(4m_0 + 2m_1 + m_0) \prod_{i=0}^{2} (-1)^{(k_i+k_{i+1})m_{n-1-i}} \\ &= \sum_{l_0=0}^{1} \sum_{l_1=0}^{1} \sum_{l_2=0}^{1} x_0(4l_2 + 2l_1 + l_0) \prod_{i=0}^{2} (-1)^{(k_i+k_{i+1})l_i} \end{aligned}$$

Expanding the 3rd summation into two terms, we get

$$\begin{aligned} X(k) &= \sum_{l_0=0}^{1} \sum_{l_1=0}^{1} \prod_{i=0}^{1} (-1)^{(k_i+k_{i+1})l_i} [x_0(2l_1 + l_0) + (-1)^{k_2+k_3} x_0(4 + 2l_1 + l_0)] \\ &= \sum_{l_0=0}^{1} \sum_{l_1=0}^{1} \prod_{i=0}^{1} (-1)^{(k_i+k_{i+1})l_i} x_1(4k_2 + 2l_1 + l_0) \end{aligned}$$

where $k_3 \triangleq 0$ and x_1 is defined as

$$x_1(4k_2 + 2l_1 + l_0) \triangleq x_0(2l_1 + l_0) + (-1)^{k_2+k_3} x_0(4 + 2l_1 + l_0) \tag{11}$$

Expanding the 2nd summation into two terms, we get

$$\begin{aligned} X(k) &= \sum_{l_0=0}^{1} (-1)^{(k_i+k_{i+1})l_0} [x_1(4k_2 + l_0) + (-1)^{k_1+k_2} x_1(4k_2 + 2 + l_0)] \\ &= \sum_{l_0=0}^{1} (-1)^{(k_i+k_{i+1})l_0} x_2(4k_2 + 2k_1 + m_0) \end{aligned}$$

where x_2 is defined as

$$x_2(4k_2 + 2k_1 + l_0) \triangleq x_1(4k_2 + l_0) + (-1)^{k_1+k_2} x_1(4k_2 + 2 + l_0) \tag{12}$$

Finally, expanding the 1st summation into two terms, we have

$$X(k) = x_2(4k_2 + 2k_1) + (-1)^{k_0+k_1} x_2(4k_2 + 2k_1 + 1) \tag{13}$$

Summarizing the above steps, we get the fast WHT_w algorithm composed of the bit-reversal and the three equations (11), (12), and (13), as illustrated below: