Fast Algorithms for the Discrete *W* Transform and for the Discrete Fourier Transform

ZHONGDE WANG, MEMBER, IEEE

Abstract—**A systematic method of sparse matrix factorization is developed for all four versions of the discrete *W* transform, the discrete cosine transform, and the discrete sine transform, as well as for the discrete Fourier transform. The factorization leads to fast algorithms in which only real arithmetic is involved. A scheme for reducing multiplications and a convenient index system are introduced. This makes new algorithms more efficient than conventional algorithms for the discrete Fourier transform, the discrete cosine transform, and the discrete sine transform.**

I. Introduction

THE *W* transform is a real approach for the harmonic analysis [1]-[3]. It is a conventional assumption that harmonics should be multiples of a fundamental frequency. However, this assumption is not a necessity. A data sequence may be composed not only of its integer harmonics, but also of its fractional harmonics [3]. Nevertheless, integer harmonics, which are multiples of the fundamental frequency, and half-integer[1] harmonics, which are odd multiples of the fundamental frequency divided by 2, are preferred. In these two cases, the symmetry and antisymmetry of a sequence may be used and the discrete *W* transform (DWT) may always be decomposed into the discrete cosine transform (DCT) and the discrete sine transform (DST).

Unlike the case of the discrete Fourier transform (DFT), where only one version of the DFT is in use, four versions of the DWT will be introduced in this paper. These four versions of DWT correspond to two types of sampling in both original and transformed domains, i.e., sampling at integers or at half-integers.

For the decomposition of the DFT matrix and especially of the DWT matrices, four versions of both DCT and DST will be involved. The different versions of the DCT and DST were of interest mainly in the area of image coding [4]. Conventionally, the DCT and DST were implemented by the FFT algorithm [5]-[7]. Since the importance of the DCT and DST have been increasing, new algorithms were developed for some versions of the DCT and DST in the last few years [8]-[11].

A systematic method of decomposition of all discrete sinusoidal transforms, i.e., the DFT, DWT, DCT, and DST, will be given in this paper. Unlike all existing FFT algorithms, the basis of which is to factorize the complex DFT matrix into complex matrices with lower order, the new method factorizes the DFT matrix and all versions of the DWT, DCT, and DST matrices into real matrices with lower order. Consequently, only real arithmetic is involved for the new method.

Manuscript received November 15, 1982; revised January 6, 1984.

The author was on leave at the Department of Electrical Engineering, University of Arizona, Tucson, AZ. He is with the Kunming Institute of Physics, Kunming, Yunnan, China.

[1] A number is called a half-integer if it is the sum of an integer and $\frac{1}{2}$.

II. Definitions

In this paper, N is used to represent an integer that is a power of 2. Notation $[\cdot]$ denotes a matrix, the order of which is represented by a subscript inside the pair of square brackets, while the version number is represented by a superscript. With these annotations, we define the following matrices.

A. Four Versions of the DWT Matrices

$$[W_N^{\mathrm{I}}] = \sqrt{2/N}\left[\sin\left(\frac{\pi}{4} + mn\frac{2\pi}{N}\right)\right] \quad m,n = 0,1,\cdots,N-1 \tag{1}$$

$$[W_N^{\mathrm{II}}] = \sqrt{2/N}\left[\sin\left(\frac{\pi}{4} + m\left(n+\frac{1}{2}\right)\frac{2\pi}{N}\right)\right] \quad m,n = 0,1,\cdots,N-1 \tag{2}$$

$$[W_N^{\mathrm{III}}] = \sqrt{2/N}\left[\sin\left(\frac{\pi}{4} + \left(m+\frac{1}{2}\right)n\frac{2\pi}{N}\right)\right] \quad m,n = 0,1,\cdots,N-1 \tag{3}$$

and

$$[W_N^{\mathrm{IV}}] = \sqrt{2/N}\left[\sin\left(\frac{\pi}{4} + \left(m+\frac{1}{2}\right)\left(n+\frac{1}{2}\right)\frac{2\pi}{N}\right)\right] \quad m,n = 0,1,\cdots,N-1. \tag{4}$$

The following relations hold for the inverse DWT matrices:

$$[W_N^{\mathrm{I}}]^{-1} = [W_N^{\mathrm{I}}] \tag{5}$$

$$[W_N^{\mathrm{II}}]^{-1} = [W_N^{\mathrm{III}}] \tag{6}$$

and

$$[W_N^{\mathrm{IV}}]^{-1} = [W_N^{\mathrm{IV}}]. \tag{7}$$

B. Four Versions of the DCT Matrices

$$[C_{N+1}^{\mathrm{I}}] = \sqrt{2/N}\,[k_m k_n \cos(mn\pi/N)] \quad m,n = 0,1,\cdots,N \tag{8}$$

$$[C_N^{\mathrm{II}}] = \sqrt{2/N}\,[k_m \cos(m(n+\tfrac{1}{2})\pi/N)] \quad m,n = 0,1,\cdots,N-1 \tag{9}$$

$$[C_N^{\mathrm{III}}] = \sqrt{2/N}\,[k_n \cos((m+\tfrac{1}{2})n\pi/N)] \quad m,n = 0,1,\cdots,N-1 \tag{10}$$

and

$$[C_N^{\mathrm{IV}}] = \sqrt{2/N}\,[\cos((m+\tfrac{1}{2})(n+\tfrac{1}{2})\pi/N)] \quad m,n = 0,1,\cdots,N-1 \tag{11}$$

where in (8)-(10)

$$k_{j(j=m \text{ or } n)} = \begin{cases} 1 & \text{if } j \neq 0 \text{ and } j \neq N, \\ 1/\sqrt{2} & \text{if } j = 0 \text{ or } j = N. \end{cases} \tag{12}$$

0096-3518/84/0800-0803$01.00

The following relations hold for the inverse DCT matrices:

$$[C_{N+1}^{\mathrm{I}}]^{-1} = [C_{N+1}^{\mathrm{I}}] \tag{13}$$

$$[C_N^{\mathrm{II}}]^{-1} = [C_N^{\mathrm{III}}] \tag{14}$$

and

$$[C_N^{\mathrm{IV}}]^{-1} = [C_N^{\mathrm{IV}}]. \tag{15}$$

The DCT-I, which is defined for the order $N+1$, was introduced into digital signal processing by Wang and Hunt [12]. The DCT-II and its inverse the DCT-III were introduced by Ahmed *et al.* [5]. The DCT-IV was introduced by Jain [7].

C. Four Versions of the DST Matrices

$$[S_{N-1}^{\mathrm{I}}] = \sqrt{2/N}\,[\sin(mn\pi/N)]$$
$$m, n = 1, 2, \cdots, N-1 \tag{16}$$

$$[S_N^{\mathrm{II}}] = \sqrt{2/N}\,[k_m \sin(m(n-\tfrac{1}{2})\pi/N)]$$
$$m, n = 1, 2, \cdots, N \tag{17}$$

$$[S_N^{\mathrm{III}}] = \sqrt{2/N}\,[k_n \sin((m-\tfrac{1}{2})n\pi/N)]$$
$$m, n = 1, 2, \cdots, N \tag{18}$$

and

$$[S_N^{\mathrm{IV}}] = \sqrt{2/N}\,[\sin((m+\tfrac{1}{2})(n+\tfrac{1}{2})\pi/N)]$$
$$m, n = 0, 1, \cdots, N-1 \tag{19}$$

where k_m and k_n in (17) and (18) are defined by (12).

The following relations hold for the inverse DST matrices

$$[S_{N-1}^{\mathrm{I}}]^{-1} = [S_{N-1}^{\mathrm{I}}] \tag{20}$$

$$[S_N^{\mathrm{II}}]^{-1} = [S_N^{\mathrm{III}}] \tag{21}$$

and

$$[S_N^{\mathrm{IV}}]^{-1} = [S_N^{\mathrm{IV}}]. \tag{22}$$

The DST-I, which is defined for the order $N-1$, and the DST-IV were introduced into digital signal processing by Jain [13], [7]. The DST-II and its inverse the DST-III were introduced by Kekre and Solanki [6].

D. Even-Odd Transform Matrices

$$[A_N^{\mathrm{I}}] = \frac{1}{\sqrt{2}} \begin{bmatrix} \sqrt{2} & & \\ & I_{N/2-1} & \bar{I}_{N/2-1} \\ & & \sqrt{2} \\ & \bar{I}_{N/2-1} & -I_{N/2-1} \end{bmatrix} \tag{23}$$

$$[A_N^{\mathrm{II}}] = \frac{1}{\sqrt{2}} \begin{bmatrix} I_{N/2} & \bar{I}_{N/2} \\ \bar{I}_{N/2} & -I_{N/2} \end{bmatrix} \tag{24}$$

$$[A_N^{\mathrm{III}}] = \frac{1}{\sqrt{2}} \begin{bmatrix} \sqrt{2} & & \\ & I_{N/2-1} & -\bar{I}_{N/2-1} \\ & & \sqrt{2} \\ & \bar{I}_{N/2-1} & I_{N/2-1} \end{bmatrix} \tag{25}$$

$$[A_N^{\mathrm{IV}}] = \frac{1}{\sqrt{2}} \begin{bmatrix} I_{N/2} & -\bar{I}_{N/2} \\ \bar{I}_{N/2} & I_{N/2} \end{bmatrix} \tag{26}$$

and

$$[A_J^{\mathrm{II}}] = \frac{1}{\sqrt{2}} \begin{bmatrix} I_{(J-1)/2} & & \bar{I}_{(J-1)/2} \\ & \sqrt{2} & \\ \bar{I}_{(J-1)/2} & & -I_{(J-1)/2} \end{bmatrix} \quad \text{for } J \text{ odd} \tag{27}$$

where in (23), (25), and (27) as well as in later matrix representations, the blank submatrices or blanks in a matrix or submatrix indicate that the elements in those regions are zeros, and

$$\bar{I} = \begin{bmatrix} & & 1 \\ & \cdot & \\ 1 & & \end{bmatrix} \tag{28}$$

is the opposite identity matrix.

The even-odd transform matrices convert sequences into their even and odd parts.

E. The Normalized DFT Matrix

$$[F_N] = \frac{1}{\sqrt{N}}\,[\exp(-imn2\pi/N)],$$
$$m, n = 0, 1, \cdots, N-1 \tag{29}$$

while $i = \sqrt{-1}$.

F. Permutation Matrices

$$[P_J] = \begin{bmatrix} 1 & 0 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 \\ 0 & 1 & 0 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & \cdot & \cdot & 1 & 0 & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & \cdot & \cdot & \cdot & 0 & 1 & \cdot & 0 & \cdot & \cdot \\ 0 & \cdot & \cdot & \cdot & 1 & 0 & \cdot & \cdot & \cdot & \cdot \end{bmatrix} \quad \text{for } J \text{ odd} \tag{30}$$

and

$$[P_J] = \begin{bmatrix} 1 & 0 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 \\ 0 & 1 & \cdot & 0 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & \cdot & \cdot & \cdot & \cdot & 1 & 0 & \cdot & \cdot & \cdot \\ 0 & \cdot & \cdot & 1 & 0 & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & \cdot & \cdot & \cdot & 1 & 0 & \cdot & \cdot & \cdot & \cdot \end{bmatrix} \quad \text{for } J \text{ even.} \tag{31}$$

$[P_J]$ reorders the components of a vector such that the first half of the components of the vector are set to be even numbered components of the reordered vector, while the last half of the components of the vector are set to be odd numbered components of the reordered vector but in a reversed order. Note that the number is counted from zero.

G. Matrix with Reversed Order

Double bars on the top of a matrix indicate that the matrix is in a reversed order for both its rows and columns

$$[\bar{\bar{\cdot}}] = [\bar{I}_\cdot][\cdot][\bar{I}_\cdot]. \tag{32}$$

Notice that all matrices defined in this section are orthonormal.

III. Decomposition of DWT and DFT Matrices

The DWT matrices may be decomposed into the DCT and DST matrices with lower order

$$[W_N^{\mathrm{I}}] = [A_N^{\mathrm{I}}] \begin{bmatrix} C_{N/2+1}^{\mathrm{I}} & \\ & \bar{S}_{N/2-1}^{\mathrm{I}} \end{bmatrix} [A_N^{\mathrm{I}}] \tag{33}$$

$$[W_N^{\mathrm{II}}] = [A_N^{\mathrm{III}}]^T \begin{bmatrix} C_{N/2}^{\mathrm{II}} & \\ & \bar{S}_{N/2}^{\mathrm{II}} \end{bmatrix} [A_N^{\mathrm{II}}] \tag{34}$$

$$[W_N^{\mathrm{III}}] = [A_N^{\mathrm{II}}] \begin{bmatrix} C_{N/2}^{\mathrm{III}} & \\ & \bar{S}_{N/2}^{\mathrm{III}} \end{bmatrix} [A_N^{\mathrm{III}}] \tag{35}$$

and

$$[W_N^{\mathrm{IV}}] = [A_N^{\mathrm{IV}}]^T \begin{bmatrix} C_{N/2}^{\mathrm{IV}} & \\ & \bar{S}_{N/2}^{\mathrm{IV}} \end{bmatrix} [A_N^{\mathrm{IV}}] \tag{36}$$

where T in (34) and (36) represents transposition.

On the other hand, since the Fourier transform is closely related to the W transform the DFT may also be decomposed into the DCT-I and DST-I with lower order as

$$[F_N] = [G_N] \begin{bmatrix} C_{N/2+1}^{\mathrm{I}} & \\ & \bar{S}_{N/2-1}^{\mathrm{I}} \end{bmatrix} [A_N^{\mathrm{I}}] \tag{37}$$

where

$$[G_N] = \frac{1}{\sqrt{2}} \begin{bmatrix} \sqrt{2} & & & \\ & I_{N/2-1} & & -i\bar{I}_{N/2-1} \\ & & \sqrt{2} & \\ & \bar{I}_{N/2-1} & & iI_{N/2-1} \end{bmatrix}. \tag{38}$$

The proof of (37) will be shown as an example.

Let

$$[C] = \left[\cos \frac{2mn\pi}{N}\right]; \quad m, n = 1, 2, \cdots, N/2 - 1 \tag{39}$$

$$[S] = \left[\sin \frac{2mn\pi}{N}\right]; \quad m, n = 1, 2, \cdots, N/2 - 1 \tag{40}$$

$$[I] = [I_{N/2-1}] \tag{41}$$

and

$$[\bar{I}] = [\bar{I}_{N/2-1}]. \tag{42}$$

The order of $[C]$, $[S]$, $[I]$, and $[\bar{I}]$ is $N/2 - 1$, which is omitted for simplicity. $[C]$ and $[S]$ related to $[C_{N/2+1}^{\mathrm{I}}]$ and $[S_{N/2-1}^{\mathrm{I}}]$ by

$$[C_{N/2+1}^{\mathrm{I}}] = \sqrt{\frac{4}{N}} \left[\begin{array}{c|ccc|c} \frac{1}{2} & \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & \cdots & \frac{1}{2} \\ \hline \frac{\sqrt{2}}{2} & & & & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & & C & & \frac{\sqrt{2}}{2} \\ \vdots & & & & \vdots \\ \hline \frac{1}{2} & -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & \cdots & \frac{1}{2} \end{array}\right] \tag{43}$$

and

$$[S_{N/2-1}^{\mathrm{I}}] = \sqrt{\frac{4}{N}}\,[S]. \tag{44}$$

Then

$$[G_N] \begin{bmatrix} C_{N/2+1}^{\mathrm{I}} & \\ & \bar{S}_{N/2-1}^{\mathrm{I}} \end{bmatrix} [A_N^{\mathrm{I}}]$$

$$= \sqrt{\frac{2}{N}}\,[G_N] \left[\begin{array}{c|ccc|c|c} \frac{1}{2} & \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & \cdots & \frac{1}{2} & \\ \hline \frac{\sqrt{2}}{2} & & & & \frac{\sqrt{2}}{2} & \\ \frac{\sqrt{2}}{2} & & C & & \frac{\sqrt{2}}{2} & \\ \vdots & & & & \vdots & \\ \hline \frac{1}{2} & -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & \cdots & \frac{1}{2} & \\ \hline & & & & & \bar{I}S\bar{I} \end{array}\right]$$

$$\cdot \begin{bmatrix} \sqrt{2} & & & \\ & I & & \bar{I} \\ & & \sqrt{2} & \\ & \bar{I} & & -I \end{bmatrix}$$

$$= \sqrt{\frac{1}{N}} \begin{bmatrix} \sqrt{2} & & & \\ & I & & -i\bar{I} \\ & & \sqrt{2} & \\ & I & & iI \end{bmatrix}$$

$$\begin{bmatrix} \frac{\sqrt{2}}{2} & 1\ 1\cdots & \frac{\sqrt{2}}{2} & 1\ 1\cdots \\ 1 & & -1 & \\ 1 & C & 1 & C\bar{I} \\ \vdots & & \vdots & \\ \frac{\sqrt{2}}{2} & -1\ 1\cdots & \frac{\sqrt{2}}{2} & -1\ 1\cdots \\ 0 & & 0 & \\ \vdots & & \vdots & \\ & \bar{I}S & & -\bar{I}S\bar{I} \end{bmatrix}$$

$$= \sqrt{\frac{1}{N}} \begin{bmatrix} 1 & 1\ 1\cdots & 1 & 1\ 1\cdots \\ 1 & & -1 & \\ 1 & C - iS & 1 & (C+iS)\bar{I} \\ \vdots & & \vdots & \\ 1 & -1\ 1\cdots & 1 & -1\ 1\cdots \\ 1 & & -1 & \\ 1 & & 1 & \\ \vdots & \bar{I}(C+iS) & \vdots & \bar{I}(C-iS)\bar{I} \end{bmatrix} \quad (45)$$

On the other hand, since

$$\exp(-imn \cdot 2\pi/N) = \cos(2mn\pi/N) - i\sin(2mn\pi/N) \quad (46)$$

writing (29) explicitly, one may obtain

$$[F_N] = \sqrt{\frac{1}{N}} \begin{bmatrix} 1 & 1\ 1\cdots & 1 & 1\ 1\cdots \\ 1 & & -1 & \\ 1 & C - iS & 1 & (C+iS)\bar{I} \\ \vdots & & \vdots & \\ 1 & -1\ 1\cdots & 1 & -1\ 1\cdots \\ 1 & & -1 & \\ 1 & & 1 & \\ \vdots & \bar{I}(C+iS) & \vdots & \bar{I}(C-iS)\bar{I} \end{bmatrix} \quad (47)$$

which coincides with (45). Therefore, (37) is proven.

The procedure of the proof of (33)-(36), as well as the decomposition of the DCT and DST matrices in the next section, is the same as above, and will be omitted for brevity.

The further decomposition of the DWT and DFT matrices relies on the decomposition of the DCT and DST matrices. In the past few years, many contributions have been made for the decomposition of the DCT and DST. Among them, perhaps the most important one is Chen's algorithm for the DCT-II [8], which includes the fast algorithm for the DCT-IV and is a model of the decomposition for the DCT and DST. Following this model, Yip and Rao found an algorithm for the DST-I [9]. Wang reported an algorithm for the DST-II, which is implemented by the fast algorithm for the DCT-II [10]. Since the DCT-III and DST-II are the inverse versions of the DCT-II and DST-II, respectively, fast algorithms for the DCT-III and DST-III are the same as fast algorithms for the DCT-II and DST-II but in a reversed order. Therefore, the remaining unsolved problem is just to find algorithms for the DCT-I and DST-IV.

IV. Decomposition of DCT and DST

Although many contributions have been made for the decomposition of $[C_N^j]$ and $[S_N^j]$, j = I, II, III, or IV [8]-[11], they are separate and unsystematic. We will now give a systematic matrix factorization for all versions of DCT and DST matrices with the order related to a power of 2. We will start from version I.

If $N \geqslant 4$, $[C_{N+1}^{\mathrm{I}}]$ and $[S_{N-1}^{\mathrm{I}}]$ may be decomposed into the following recursive forms:

$$[C_{N+1}^{\mathrm{I}}] = [P_{N+1}] \begin{bmatrix} C_{N/2+1}^{\mathrm{I}} & \\ & \bar{C}_{N/2}^{\mathrm{III}} \end{bmatrix} [A_{N+1}^{\mathrm{II}}] \quad (48)$$

and

$$[S_{N-1}^{\mathrm{I}}] = [P_{N-1}] \begin{bmatrix} S_{N/2}^{\mathrm{III}} & \\ & \bar{S}_{N/2-1}^{\mathrm{I}} \end{bmatrix} [A_{N-1}^{\mathrm{II}}], \quad (49)$$

where $[P_{N+1}]$ and $[P_{N-1}]$ are defined by (30), $[A_{N+1}^{\mathrm{II}}]$ and $[A_{N-1}^{\mathrm{II}}]$ are defined by (27).

The decomposition of $[C_N^{\mathrm{II}}]$ was solved by Chen *et al.* [8]. In a slightly different form, $[C_N^{\mathrm{II}}]$ may be factorized as

$$[C_N^{\mathrm{II}}] = [P_N] \begin{bmatrix} C_{N/2}^{\mathrm{II}} & \\ & \bar{C}_{N/2}^{\mathrm{IV}} \end{bmatrix} [A_N^{\mathrm{II}}], \quad (N \geqslant 4) \quad (50)$$

where $[P_N]$ is given by (31) and $[A_N^{\mathrm{II}}]$ is given by (24). $[S_N^{\mathrm{II}}]$ may be decomposed in the same manner

$$[S_N^{\mathrm{II}}] = [P_N] \begin{bmatrix} S_{N/2}^{\mathrm{IV}} & \\ & \bar{S}_{N/2}^{\mathrm{II}} \end{bmatrix} [A_N^{\mathrm{II}}] \quad (N \geqslant 4). \quad (51)$$

The decomposition of $[C_N^{\mathrm{III}}]$ may be obtained from (14) and (50).

$$\begin{aligned} [C_N^{\mathrm{III}}] &= [C_N^{\mathrm{II}}]^{-1} \\ &= [C_N^{\mathrm{II}}]^T \\ &= [A_N^{\mathrm{II}}]^T \begin{bmatrix} C_{N/2}^{\mathrm{II}} & \\ & \bar{C}_{N/2}^{\mathrm{IV}} \end{bmatrix}^T [P_N]^T \\ &= [A_N^{\mathrm{II}}] \begin{bmatrix} C_{N/2}^{\mathrm{III}} & \\ & \bar{C}_{N/2}^{\mathrm{IV}} \end{bmatrix} [P_N]^T \quad (N \geqslant 4). \end{aligned} \quad (52)$$

Similarly

$$[S_N^{III}] = [A_N^{II}] \begin{bmatrix} S_{N/2}^{IV} & \\ & \bar{S}_{N/2}^{III} \end{bmatrix} [P_N]^T \quad (N \geqslant 4). \tag{53}$$

On the other hand, $[S_N^{II}]$ relates to $[C_N^{II}]$ by [10]

$$[S_N^{II}] = [\bar{I}_N] [C_N^{II}] [D_N] \tag{54}$$

where

$$[D_N] = \begin{bmatrix} 1 & & & & \\ & -1 & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & -1 \end{bmatrix}. \tag{55}$$

The same relation holds between $[S_N^{IV}]$ and $[C_N^{IV}]$

$$[S_N^{IV}] = [\bar{I}_N] [C_N^{IV}] [D_N]. \tag{56}$$

Therefore, the matrix factorization of all versions of the DCT and DST matrices and, hence, of all versions of the DWT matrices, as well as of the DFT matrix, depends only on the decomposition of $[C_N^{IV}]$.

V. Decomposition of $[C_N^{IV}]$

Chen *et al.* were the first to describe the decomposition of $[C_N^{IV}]$ [8]. But the correct results were given by Wang [11]. However, the index system used in [11] is rather complicated. A more convenient index system will be given below.

Let

$$J = \log_2 N. \tag{57}$$

$[C_N^{IV}]$ then may be written into a product of $2J + 1$ sparse matrices

$$[C_N^{IV}] = [Q_N] [V_N(J)] [U_N(J-1)] \cdot [V_N(J-1)] \cdots [U_N(1)] [V_N(1)] [H_N]. \tag{58}$$

These matrices are of five distinct types.

Type 1: The first matrix is a permutation matrix that changes the odd-numbered components of the vector to be transformed into a reversed order.

$$[Q_N] = \begin{bmatrix} 1 & 0 & \cdots & & & & & & & 0 \\ 0 & \cdots & & & & & & & 0 & 1 \\ 0 & 0 & 1 & 0 & \cdots & & & & & 0 \\ \cdots & & & & & & & & & \cdots \\ \cdots & & & & & & & & & \cdots \\ 0 & 0 & 0 & 1 & 0 & \cdots & & & & 0 \\ 0 & \cdots & & & & & & 0 & 1 & 0 \\ 0 & 1 & 0 & \cdots & & & & & & 0 \end{bmatrix}. \tag{59}$$

Type 2: The last matrix $[H_N]$ is also a permutation matrix that changes an increasing index into a Hadamard index. $[H_N]$ may be formed by

$$[H_N] = [P_N] \begin{bmatrix} P_{N/2} & \\ & \bar{P}_{N/2} \end{bmatrix} \begin{bmatrix} P_{N/4} & & & \\ & \bar{P}_{N/4} & & \\ & & P_{N/4} & \\ & & & \bar{P}_{N/4} \end{bmatrix} \cdots \cdot \begin{bmatrix} P_4 & & & & \\ & \bar{P}_4 & & & \\ & & \ddots & & \\ & & & P_4 & \\ & & & & \bar{P}_4 \end{bmatrix} \quad \text{for } N \geqslant 4 \tag{60}$$

where $[P_4] \cdots [P_N]$ are defined by (31).

Type 3: Matrices $U_N(j), j = 1, 2, \cdots, J-1$ are block diagonal binary matrices

$$[U_N(j)] = \frac{1}{\sqrt{2}} \text{ block diag } \{B(j), B(j), \cdots, B(j)\} \tag{61}$$

where

$$B(j) = \begin{bmatrix} I_{2^j} & I_{2^j} \\ I_{2^j} & -I_{2^j} \end{bmatrix}. \tag{62}$$

Type 4: The second matrix $V_M(J)$ is a block diagonal matrix formed by

$$[V_N(J)] = \text{block diag } \{T_{1/4N}, T_{5/4N}, \cdots, T_{(2N-3)/4N}\} \tag{63}$$

where

$$T_r = \begin{bmatrix} \cos r\pi & \sin r\pi \\ \sin r\pi & -\cos r\pi \end{bmatrix}. \tag{64}$$

Type 5: All remaining matrices $V_N(j), j = 1, 2, \cdots, J-1$, are block diagonal matrices. It is formed by alternating submatrices $[I_{2^j}]$ and $[E(j)]$ from the upper left to the lower right on the main diagonal

$$[V_M(j)] = \text{block diag } \{I_{2^j}, E(j), I_{2^j}, \cdots, E(j)\} \tag{65}$$

where $E(j)$ is defined by

$$E(j) = \text{block diag } \{T_{1/2^{j+1}}, T_{5/2^{j+1}}, \cdots, T_{(2^{j+1}-3)/2^{j+1}}\} \tag{66}$$

and T_r is defined by (64).

It should be noted that all those $2J + 1$ matrices in (58) are symmetric orthonormal matrices. Since $[C_N^{IV}]$ is symmetric, the order of the product in (58) may be reversed

$$[C_N^{IV}] = [H_N] [V_N(1)] [U_N(1)] \cdots [V_N(J-1)] \cdot [U_N(J-1)] [V_N(J)] [Q_N] \tag{67}$$

$$
\begin{aligned}
[C_8^{\mathrm{IV}}] = \sqrt{\frac{2}{8}}
&\begin{bmatrix}
1&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&1\\
0&0&1&0&0&0&0&0\\
0&0&0&0&0&1&0&0\\
0&0&0&0&1&0&0&0\\
0&0&0&1&0&0&0&0\\
0&0&0&0&0&0&1&0\\
0&1&0&0&0&0&0&0
\end{bmatrix}
\begin{bmatrix}
C_{1/32} & S_{1/32} & & & & & & \\
S_{1/32} & -C_{1/32} & & & & & & \\
 & & C_{5/32} & S_{5/32} & & & & \\
 & & S_{5/32} & -C_{5/32} & & & & \\
 & & & & C_{9/32} & S_{9/32} & & \\
 & & & & S_{9/32} & -C_{9/32} & & \\
 & & & & & & C_{13/32} & S_{13/32} \\
 & & & & & & S_{13/32} & -C_{13/32}
\end{bmatrix}\\
&\cdot\begin{bmatrix} I_4 & I_4 \\ I_4 & -I_4 \end{bmatrix}
\begin{bmatrix}
I_4 & & & \\
 & C_{1/8} & S_{1/8} & \\
 & S_{1/8} & -C_{1/8} & \\
 & & & \begin{matrix} C_{5/8} & S_{5/8} \\ S_{5/8} & -C_{5/8} \end{matrix}
\end{bmatrix}
\begin{bmatrix}
I_2 & I_2 & & \\
I_2 & -I_2 & & \\
 & & I_2 & I_2 \\
 & & I_2 & -I_2
\end{bmatrix}\\
&\cdot\begin{bmatrix}
I_2 & & & & & \\
 & C_{1/4} & S_{1/4} & & & \\
 & S_{1/4} & -C_{1/4} & & & \\
 & & & I_2 & & \\
 & & & & C_{1/4} & S_{1/4} \\
 & & & & S_{1/4} & -C_{1/4}
\end{bmatrix}
\begin{bmatrix}
1&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&1\\
0&0&0&1&0&0&0&0\\
0&0&0&0&1&0&0&0\\
0&1&0&0&0&0&0&0\\
0&0&0&0&0&0&1&0\\
0&0&1&0&0&0&0&0\\
0&0&0&0&0&1&0&0
\end{bmatrix}.
\end{aligned}
\tag{68}
$$

The decomposition of $[C_8^{\mathrm{IV}}]$ is shown in (68) as an example, where $C_r = \cos r\pi$ and $S_r = \sin r\pi$.

The signal flow graph of $[C_{32}^{\mathrm{IV}}]$ is shown in Fig. 1, where the constant $\sqrt{2/32}$ does not appear. Comparing it with the graph given in [11], one may find that the index system in Fig. 1 is more convenient.

It is interesting to notice that if $[Q_N]$ and all $[V_N(j)]$ matrices except $[V_N(J)]$ in (58) are taken off, and if we let all $[T_r]$ in $[V_N(J)]$ be $[T_{1/4}]$, then the remaining matrix product accomplishes a Walsh matrix. In other words

$$
\begin{aligned}
[WA_N] = {} & [U_N(0)]\,[U_N(J-1)] \\
& \cdot [U_N(J-2)] \cdots [U_N(1)]\,[H_N]
\end{aligned}
\tag{69}
$$

where $[WA_N]$ is the Walsh matrix of order N,

$$
[U_N(0)] = \frac{1}{\sqrt{2}}\ \text{block diag}\ \{B(0), B(0), \cdots, B(0)\}, \tag{70}
$$

$[H_N]$, $[U_N(j)]$, $j = 1, 2, \cdots, J-1$, and $B(0)$ are obtained by (60), (61), and (62), respectively.

VI. Considerations for Reducing the Number of Multiplications

Neglecting the factor $1/\sqrt{2}$ in all $[U]$ matrices of (61), which may be moved to the end of the procedure, all multiplications in (58) are concentrated in matrices $[V_N(j)]$, $j = 1, 2, \cdots, J$. The basis of $[V_N(j)]$ is $[T_r]$ of (64), which is a 2×2 matrix. Normally, it requires four multiplications and two additions for $[T_r]$. However, $[T_r]$ may be further factorized into three sparse matrices

$$
\begin{aligned}
[T_r] &= \begin{bmatrix} \cos r\pi & \sin r\pi \\ \sin r\pi & -\cos r\pi \end{bmatrix} \\
&= \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & 1 \end{bmatrix}
\cdot \begin{bmatrix} \cos r\pi + \sin r\pi & 0 & 0 \\ 0 & -\cos r\pi + \sin r\pi & 0 \\ 0 & 0 & \sin r\pi \end{bmatrix} \\
&\quad \cdot \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & -1 \end{bmatrix}.
\end{aligned}
\tag{71}
$$

Coefficients $\cos r\pi + \sin r\pi$ and $-\cos r\pi + \sin r\pi$ may be precalculated, and thus only three multiplications together with three additions are required.

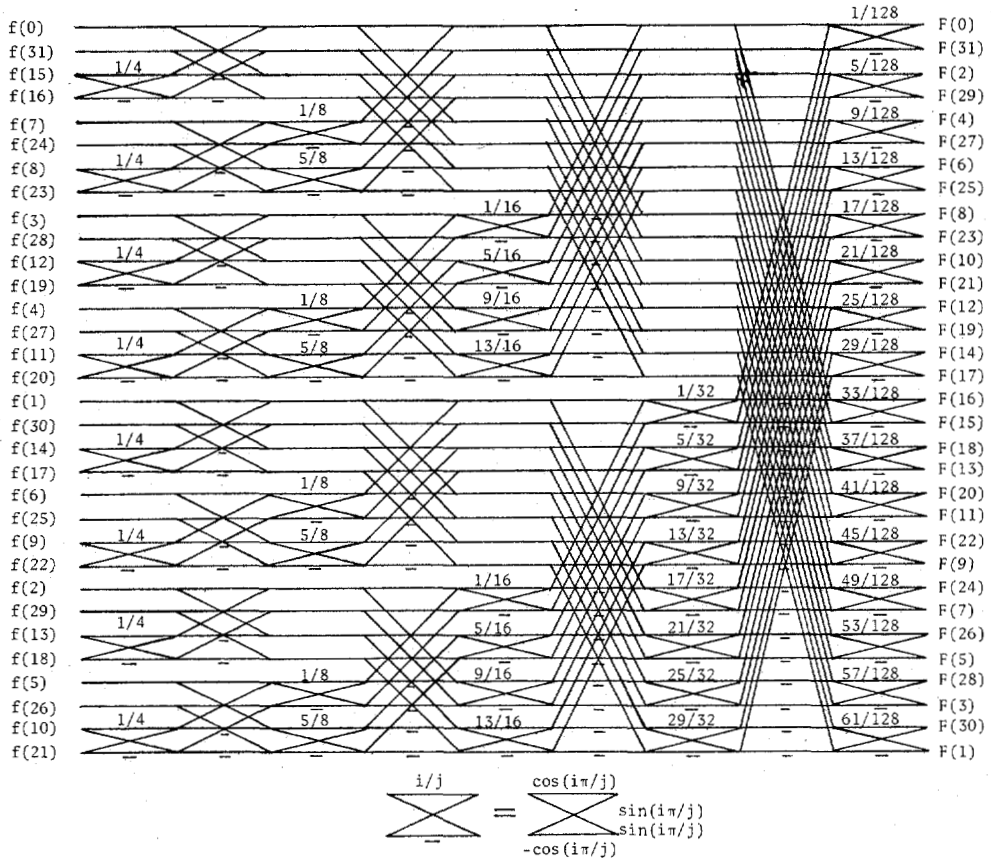

Fig. 1. Signal flow graph for $[C_{32}^{IV}]$.

If $r = 1/4$, since $\cos \pi/4 = \sin \pi/4 = 1/\sqrt{2}$,

$$[T_{1/4}] = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}. \tag{72}$$

The arithmetic operations for $[T_{1/4}]$ reduce to two multiplications and two additions. This has already been counted by Chen *et al.* [8].

VII. Arithmetic Operations

Let $\mu(L)$ and $\alpha(L)$ be the number of multiplications and additions for matrix $[L]$ respectively; e.g., $\mu(C_{N/2}^{IV})$ is the number of multiplications for $[C_{N/2}^{IV}]$; $\alpha(V_N(j))$ is the number of additions for $[V_N(j)]$, etc. Then $\mu(C_N^{IV})$ and $\alpha(C_N^{IV})$ may be evaluated from (58).

In (58), $[Q_N]$ and $[H_N]$ require neither multiplications nor additions. Neglecting the factor $1/\sqrt{2}$ in (61), which may be moved to the end of the procedure, each of the $[U_N(j)]$, $j = 1, 2, \cdots, J-1$, requires only N additions but no multiplications. $[V_N(j)]$, $j = 1, 2, \cdots, J$, consists of $[T_r]$ of (71), which requires the same amount of multiplications and additions. $[V_N(1)]$ contains $N/4$ of $[T_{1/4}]$; $[V_N(J)]$ contains $N/2$ of $[T_r]$, where $r \neq 1/4$; each of the other $[V_N(j)]$, $j = 2, 3, \cdots, J-1$, contains $N/4$ of $[T_r]$, where $r \neq 1/4$. Hence

$$\mu(C_N^{IV}) = \mu(V_N(1)) + \mu(V_N(J)) + \sum_{j=2}^{J-1} \mu(V_N(j))$$

$$= \frac{N}{4}\mu(T_{1/4}) + \frac{N}{2}\mu(T_r) + \frac{N}{4}(J-2)\mu(T_r)$$

$$= \frac{N}{4}[J\mu(T_r) + \mu(T_{1/4})] \qquad r \neq \frac{1}{4}$$

$$= \frac{N}{4}(3J+2); \tag{73}$$

$$\alpha(C_{IV}^N) = \mu(C_{IV}^N) + \sum_{j=1}^{J-1} \alpha(U_N(j))$$

$$= \frac{N}{4}(3J+2) + N(J-1)$$

$$= \frac{N}{4}(7J-2) \tag{74}$$

where $\mu(T_{1/4}) = 2$ and $\mu(T_r) = 3$ for $r \neq 1/4$ are obtained from the last section.

The arithmetic operations required for all other matrices may be derived from $\mu(C_N^{IV})$ and $\alpha(C_N^{IV})$ by the following relations:

$$\mu(C_N^{II}) = \mu(C_{N/2}^{II}) + \mu(C_{N/2}^{IV}) \tag{75}$$

$$\alpha(C_N^{II}) = \alpha(C_{N/2}^{II}) + \alpha(C_{N/2}^{IV}) + N \tag{76}$$

$$\mu(C_N^{III}) = \mu(S_N^{III}) = \mu(S_N^{II}) = \mu(C_N^{II}) \tag{77}$$

$$\alpha(C_N^{III}) = \alpha(S_N^{III}) = \alpha(S_N^{II}) = \alpha(C_N^{II}) \tag{78}$$

$$\mu(C_{N+1}^{I}) = \mu(C_{N/2+1}^{I}) + \mu(C_{N/2}^{II}) - 1 \tag{79}$$

$$\alpha(C_{N+1}^{I}) = \alpha(C_{N/2+1}^{I}) + \alpha(C_{N/2}^{II}) + N \tag{80}$$

$$\mu(S^{I}_{N-1}) = \mu(S^{I}_{N/2-1}) + \mu(S^{II}_{N/2}) - 1 \quad (81)$$

$$\alpha(S^{I}_{N-1}) = \alpha(S^{I}_{N/2-1}) + \alpha(S^{II}_{N/2}) + N - 2 \quad (82)$$

$$\mu(S^{IV}_{N}) = \mu(C^{IV}_{N}) \quad (83)$$

$$\alpha(S^{IV}_{N}) = \alpha(C^{IV}_{N}) \quad (84)$$

$$\mu(W^{I}_{N}) = \mu(C^{I}_{N/2+1}) + \mu(S^{I}_{N/2-1}) - 4 \quad (85)$$

$$\alpha(W^{I}_{N}) = \alpha(C^{I}_{N/2+1}) + \alpha(S^{I}_{N/2-1}) + 2N - 4 \quad (86)$$

$$\mu(W^{II}_{N}) = \mu(W^{III}_{N}) = 2\mu(C^{II}_{N/2}) - 2 \quad (87)$$

$$\alpha(W^{II}_{N}) = \alpha(W^{III}_{N}) = 2\alpha(C^{II}_{N/2}) + 2N - 2 \quad (88)$$

$$\mu(W^{IV}_{N}) = 2\mu(C^{IV}_{N/2}) \quad (89)$$

and

$$\alpha(W^{IV}_{N}) = 2\alpha(C^{IV}_{N/2}) + 2N. \quad (90)$$

These relations are from the equations for matrix decomposition, i.e., (33)-(37) and (48)-(56). The factor $1/\sqrt{2}$ in all $[A]$ matrices is not counted into the number of multiplications because it may be moved to the end of the calculation. Multiplying by -1 is not counted as a multiplication. In addition, the element $\sqrt{2}$ in all $[A]$ matrices has been cancelled by k_j of (12) in the DCT and DST matrices. This also contributes to the reduction of the number of multiplications. For example, from (33) one might think that the multiplications required for $[W^{I}_{N}]$ should be

$$\mu(W^{I}_{N}) = \mu(C^{I}_{N/2+1}) + \mu(S^{I}_{N/2-1}) + 4$$

where the last four multiplications were contributed by four $\sqrt{2}$ elements in two $[A^{I}_{N}]$. Since these $\sqrt{2}$ elements are cancelled by $\sqrt{2}/2$ of k_j in $[C^{I}_{N/2+1}]$ when $j = 0$ or $j = N/2$, the correct relation is given by (85).

After some fundamental mathematic manipulation, the multiplications and additions may be represented in terms of N. The results are listed in Table I.

Table II compares the arithmetic operations of the present method to the conventional radix-2 FFT [14]-[16] and to Preuss' very fast Fourier transform algorithm [17]. Table III compares the present method to Chen's algorithm for the DCT-II. Tables II and III show that the present method is more efficient than the conventional radix-2 FFT and Chen's algorithm for DCT-II, but is less efficient than Preuss' algorithm.

As an illustration, a Fortran program for DWT-I and DFT is given in the Appendix.

VIII. Comparison of DWT-I with DFT

Except for the difference shown by (33) and (37), the new algorithm for the DFT is the same as for the DWT-I. The difference between them is only in the last step. Combining the even and odd parts together in the last step, one obtains the DWT-I coefficients. Keeping the even parts as the real parts and the odd parts as the imaginary parts of the complex coefficients, one obtains the DFT coefficients.

Both the DWT-I and DFT are implementations for the harmonic analysis of data sequences. In the real world, data sequences are usually real. For a real input data sequence, the outputs of the DWT-I are real, but the outputs of the DFT are complex. A complex data sequence may be treated as two real data sequences. The computation for either DWT-I or DFT of a complex sequence is twice as much as the computation of the DWT-I or DFT of a real sequence. The harmonic analysis usually includes the forward transform, some processing or filtering in the frequency domain and the inverse transform to

TABLE I
Arithmetic Operations for DCT, DST, DWT, and DFT

Transformation	Multiplications	Additions
DWT-I, DFT for N>8	$(N/4)(3\log_2 N-13)+4\log_2 N-2$	$(N/4)(7\log_2 N-13)+4\log_2 N-2$
DCT-I for N+1>5	$(N/4)(3\log_2 N-10)+2\log_2 N+5$	$(7/4)N(\log_2 N-2)+3\log_2 N+4$
DST-I for N-1>3	$(N/4)(3\log_2 N-10)+2\log_2 N+1$	$(7/4)N(\log_2 N-2)+\log_2 N+2$
DWT_{II}, DWT_{III} for N>8	$(3/4)N(\log_2 N-2)+4$	$(7/4)N(\log_2 N-1)+4$
DCT-II, DST-II		
DCT-III, DST-III for N>4	$N((3/4)\log_2 N-1)+3$	$N((7/4)\log_2 N-2)+3$
DWT-IV for N>8	$N(3\log_2 N-1)/4$	$N(7\log_2 N-1)/4$
DCT-IV, DST-IV for N>4	$(3\log_2 N+2)/4$	$N(7\log_2 N-2)/4$

TABLE II
Comparison of Different Radix-2 DFT Algorithms with Complex Inputs

	Present Method		References [14], [15], [16]		Reference [17]	
N	Multiplies	Adds	Multiplies	Adds	Multiplies	Adds
8	4	52	6	54	4	64
16	20	148	30	158	20	172
32	68	388	102	422	68	448
64	204	972	294	1062	196	1124
128	564	2356	774	2566	516	2728
256	1468	5564	1926	6022	1284	6444
512	3652	12868	4614	13830	3076	14896
1024	8780	29260	10758	31238	7172	33844
2048	20564	65620	24582	69638	16388	75832
4096	47196	145500	55302	153606	36868	167996
8192	106596	319588	122886	335878	81924	368704

TABLE III
Comparison of DCT-II Algorithms

	Present Method		Reference [8]	
N	Multiplies	Adds	Multiplies	Adds
4	5	9	6	8
8	13	29	16	26
16	35	83	44	74
32	91	219	116	194
64	227	547	292	482
128	547	1315	708	1154
256	1283	3075	1668	2690
512	2947	7043	3884	6146
1024	6659	15875	8708	13826
2048	14851	35331	19460	30722
4096	32771	77828	43012	67586

reconstruct the desired sequence. Therefore, the computation of the harmonic analysis via the DFT for doing both the forward and the inverse transformation is 50 percent more than that via the DWT-I, because the inputs of the inverse discrete Fourier transform are complex, but the inputs of the inverse DWT-I are real. From this point of view, the DWT-I is superior to the DFT.

Usually, it requires $2N$ memory storages to store the DFT coefficients of an N data sequence, since the DFT coefficients are complex. However, only N memory storages are required to store the DWT coefficients of an N data sequence. Therefore, the DWT is also superior to the DFT in this aspect.

The estimation of the frequency coefficients of a convolution is another important aspect of the harmonic analysis. The DFT of the convolution of two sequences is the complex product of the DFT's of these two sequences, which is equivalent to four real multiplications together with two additions. On the other hand, the DWT of the convolution of two sequences is the mirror product [18] of the DWT's of these two sequences, which is equivalent to two multiplications together with three additions. This is less than the arithmetic operations required for the DFT of the convolution.

From the above arguments one may conclude that the DWT-I outperforms the DFT in all three aspects; i.e., the computation for doing the transformations, the memory storage requirement, and the computation for obtaining the frequency coefficients of a convolution.

IX. Conclusion

This paper has presented algorithms for all four versions of the DWT, DCT, and DST as well as for the DFT together with a Fortran subroutine for DWT-I and DFT. Although the present method for DFT is less efficient than Preuss' algorithm, the algorithms presented in this paper indicate the close relationship among different versions of the DWT, DCT, and DST as well as their relation with the DFT. Since the DFT matrix consists of imaginary exponential power elements, all existing algorithms for the DFT are based on the usage of the properties of the imaginary exponential power, while the approach in this paper is based on the real matrix factorization. This new approach might be applied to mixed radix FFT algorithms or to the development of mixed radix DWT algorithms.

Since the DWT coefficients of a real sequence are real, while the DFT coefficients are complex, harmonic analysis via the DWT is more efficient than harmonic analysis via the DFT.

Appendix

```
C*************************************************************************
C
C       PROGRAMMER: ZHONGDE WANG
C       DATE: OCTOBER 1982
C
C       THIS PROGRAM DEMOSTRATES THE FAST ALGORITHM FOR RADIX-2 DISCRETE W
C       AND FOURIER TRANSFORMS. IT IS AN IN-PLACE ALGORITHM AND  USES REAL
C       ARITHMETIC ONLY.
C
C       FOR THE W TRANSFORM, BOTH INPUT AND OUTPUT SEQUENCES ARE IN
C       SEQUENCE ORDER. FOR THE FOURIER TRANSFORM OF A REAL SEQUENCE,
C       THE REAL PARTS OF THE FREQUENCE COMPONENTS ARE STORED IN THE
C       FIRST N/2+1 UNITS OF THE ARRAY WITH A SEQUENCE ORDER; THE
C       IMAGINARY PARTS OF THE FREQUENCE COMPONENTS ARE STORED IN THE LAST
C       N/2-1 UNITS OF THE ARRAY BUT IN A REVERSED ORDER. LET F1(I) BE THE
C       (I-1)TH FOURIER FREQUENCE COMPONENT. THEN FOR I=2 TO N/2,
C       F1(I)=CMPLX(F(I),-F(N+2-I));
C       F1(N+2-I)=CMPLX(F(I),F(N+2-I))
C       F1(1)=CMPLX(F(1),0.) AND F1(N/2+1)=CMPLX(F(N/2+1),0.);
C       WHERE F IS THE OUTPUT OF THIS PROGRAM.
C
C       PARAMETER EXPLANATION:
C       F:    INPUT AND OUTPUT DATA SET. THE DIMENSION OF F IS N.
C       G:    STORAGE FOR PERMUTATION, THE DIMENSION OF WHICH SHOULD BE
C             EQUAL OR GREATER THAN N/4.
C       A,B,C:   COEFFICIENTS OBTAINED BY CALLING SUBROUTINE COEF(A,B,C,N).
C             THE DIMENSION OF EACH OF THEM IS 3*N/16. FOR REPEATING
C             TRANSFORMATION, THE SUBROUTINE COEF(A,B,C,N) ONLY HAVE TO BE
C             CALLED ONCE.
C       N:    NUMBER OF DATA, THE DIMENSION OF F, WHICH MUST BE A POWER OF
C             2 AND EQUAL OR GREATER THAN 8.
C       NP:   PARAMETER FOR DETERMINING THE REQUIRED TRANSFORM.
C          NP.GE.0:  FORWARD TRANSFORM.
C          NP.LT.0:  INVERSE TRANSFORM.
C          ABS(NP)=0:  NORMALIZED W TRANSFORM.
C          ABS(NP)=1:  UNNORMALIZED W TRANSFORM.
C          ABS(NP)=2:  FOURIER TRANSFORM.
C          ABS(NP)=3:  FOURIER AND UNNORMALIZED W TRANSFORM FOR A SYMETRIC
C                      DATA SET.
C          ABS(NP)=4:  FOURIER AND UNNORMALIZED W TRANSFORM FOR A ANTI-SYMETRIC
C                      DATA SET.
C
C*************************************************************************
```

```
C-----------------------------------------------------------------------
C       MAIN SUBROUTINE
C-----------------------------------------------------------------------
        SUBROUTINE FWFT(F,G,A,B,C,N,NP)
        DIMENSION F(N),G(1),C(1),A(1),B(1)
C
        NM=ABS(NP)
        NO=N/2
        N9=1
C-----------------------------------------------------------------------
C       TO DECIDE IF FORWARD OR INVERSE TRANSFORM IS REQUIRED
C-----------------------------------------------------------------------
        IF(NP.GT.-2) GOTO 20
C-----------------------------------------------------------------------
C       THE REQUIREMENT OF THE INVERSE TRANSFORM
C-----------------------------------------------------------------------
        DO 11 I=2,N
11      F(I)=F(I)+F(I)
        F(NO+1)=.5*F(NO+1)
        GOTO 30
C-----------------------------------------------------------------------
C       TO SAPERATE THE EVEN AND ODD PARTS OF THE DATA
C-----------------------------------------------------------------------
20      N2=N+2
        DO 21 I=2,NO
        K=N2-I
        W=F(I)
        F(I)=W+F(K)
        F(K)=W-F(K)
21      CONTINUE
        IF(N9.EQ.0) RETURN
        IF(NP+1.LT.0) RETURN
C-----------------------------------------------------------------------
C       TO PERFORM THE COSINE TRANSFORM DEFINED BY ZHONGDE WANG
C-----------------------------------------------------------------------
30      CONTINUE
        N2=N+1
        N4=N/4
31      N1=2*N4+2
        IF(NM.EQ.4) GOTO 40
        DO 32 I=1,N4
        W=F(N1-I)
        F(N1-I)=F(I)-W
32      F(I)=F(I)+W
        CALL EDCT3(F(N4+2),G,A,B,C,N4,1)
        IF(NM.EQ.3) GOTO 42
C-----------------------------------------------------------------------
C       TO PERFORM THE SINE TRANSFORM DEFINED BY A. K. JAIN
C-----------------------------------------------------------------------
40      N5=N2-2*N4
        N3=N4-1
        DO 41 I=1,N3
        I2=I+N5
        W=F(N2-I)
        F(N2-I)=W-F(I2)
41      F(I2)=W+F(I2)
        CALL EDCT3(F(N5+1),G,A,B,C,N4,-1)
42      N4=N4/2
        IF(N4.GT.1) GOTO 31
        W=F(1)+F(3)
        F(3)=F(1)-F(3)
        F(1)=W+F(2)
        F(2)=W-F(2)
C-----------------------------------------------------------------------
C       CHANGE SIGN AS REQUIRED BY THE SINE TRANSFORM
C-----------------------------------------------------------------------
        N1=NO+2
        N2=N-2
        DO 51 I=N1,N2,2
51      F(I)=-F(I)
C-----------------------------------------------------------------------
C       DOING PERMUTATION
```

```
C-----------------------------------------------------------------------
      N1=N0/2
      N2=N1+1
      IF(NM.EQ.4) GOTO 70
C-----------------------------------------------------------------------
C     TO PERMUTE THE EVEN PART OF THE TRANSFORMED DATA INTO SEQUENCE
C     ORDER
C-----------------------------------------------------------------------
      DO 60 I=1,N1
60    G(I)=F(I+1)
      DO 61 I=1,N1
61    F(2*I)=F(N2+I)
      N1=N/8
      N2=1
62    N2=N2*2
      DO 63 I=1,N1
63    F(N2*(2*I-1)+1)=G(N1+I)
      N1=N1/2
      IF(N1.GE.1) GOTO 62
      F(N0+1)=G(1)
C-----------------------------------------------------------------------
C     TO PERMUTE THE ODD PART OF THE TRANSFORMED DATA INTO SEQUENCE
C     ORDER
C-----------------------------------------------------------------------
      IF (NM.EQ.3) GOTO 80
70    N1=N0/2
      N2=N1-1
      N4=N-N2
      N5=N+2
      N6=N5-N1
      DO 72 I=1,N2
72    G(I)=F(N4+I)
      DO 73 I=1,N2
73    F(N5-2*I)=F(N6-I)
      N2=1
      N3=-N1
74    N1=N1/2
      N3=N3+N1*2
      N2=N2*2
      DO 75 I=1,N1
75    F(N0+1+N2*(2*I-1))=G(N3+I)
      IF(N1.GT.1) GOTO 74
C-----------------------------------------------------------------------
C     MULTIPLYING A FACTOR FOR NORMALIZATION
C-----------------------------------------------------------------------
80    CONTINUE
      IF(NP.LT.0) GOTO 82
      W=1./N
      IF(NP.EQ.0) W=SQRT(W)
      DO 81 I=1,N
81    F(I)=W*F(I)
82    IF(NM.EQ.0.OR.NM.EQ.1) N9=0
      IF(NP.LE.1) GOTO 20
      RETURN
      END


C***********************************************************************
C     THIS SUBROUTINE IS TO OBTAIN THE EVEN COSINE-III TRANSFORM
C     THE INPUT DATA IS IN A REVERSED ORDER BUT THE TRANSFORMED DATA
C     IS IN SEQUENCE ORDER
C***********************************************************************
      SUBROUTINE EDCT3(F,G,A,B,C,N,NP)
      DIMENSION F(N),G(N),A(1),B(1),C(1)
      IF(N.EQ.2) GOTO 11
C-----------------------------------------------------------------------
C     TO PERMUTE THE INPUT DATA
C-----------------------------------------------------------------------
      N2=N/2
      N0=0
      M=2
1     DO 2 I=1,N2
2     G(N0+I)=F(N+M/2-M*I)
```

```
      NO=NO+N2
      N2=N2/2
      M=2*M
      IF(N2.GE.1) GOTO 1
      N1=N-1
      DO 3 I=1,N1
3     F(I)=G(I)
      N1=1
      N2=N/2
C-----------------------------------------------------------------------
C     DOING TRANSFORM
C-----------------------------------------------------------------------
10    CONTINUE
      CALL EDCT4(F(N1),G,A,B,C,N2)
      N1=N1+N2
      N2=N2/2
      IF(N2.GE.2) GOTO 10
11    F(N-1)=C(1)*F(N-1)
      N1=1
      NO=N+1
      IF(N.EQ.2) GOTO 20
12    N2=2*N1
      DO 13 I=1,N1
      I1=NO-I
      I2=N-N2+I
      W=F(I1)
      F(I1)=W+F(I2)
13    F(I2)=W-F(I2)
      N1=2*N1
      IF(N2.LT.N/2) GOTO 12
C-----------------------------------------------------------------------
C     DIFFERENT TREATMENT FOR EDCT3 OR EDST3 TRANSFORM
C-----------------------------------------------------------------------
20    DO 22 I=1,N1
      W=F(NO-I)
      IF(NP.EQ.1) GOTO 21
      F(NO-I)=W+F(I)
      F(I)=W-F(I)
      GOTO 22
21    F(NO-I)=W-F(I)
      F(I)=W+F(I)
22    CONTINUE
      RETURN
      END

C*********************************************************************
C
C     THIS IS A SUBROUTINE FOR EVEN COSINE-IV TRANSFORM
C     BOTH INPUT AND OUTPUT DATA ARE IN SEQUENCE ORDER
C
C*********************************************************************
      SUBROUTINE EDCT4(F,G,A,B,C,N)
      DIMENSION F(N),A(1),B(1),C(1),G(1)
      IF(N.EQ.2) GOTO 30
      NO=N/2
      N1=NO
      N2=1
C-----------------------------------------------------------------
C     TO PERMUTE THE INPUT DATA INTO A HADMARD ORDER
C-----------------------------------------------------------------
10    DO 11 I=1,N1
      I1=N-I*N2
      I2=(I-1)*N2
      DO 11 J=1,N2
11    G(I2+J)=F(I1+J)
      DO 12 I=1,N1
      I1=N-2*I*N2
      I2=NO-I*N2
      DO 12 J=1,N2
12    F(I1+J)=F(I2+J)
      DO 13 I=1,N1
      I1=(2*I-1)*N2
```

```
      I2=(I-1)*N2
      DO 13 J=1,N2
13    F(I1+J)=G(I2+J)
      N2=2*N2
      N1=N1/2
      IF(N1.GE.2)GOTO 10
C-----------------------------------------------------------------
C     TO PERFORM THE EVEN COSINE-IV TRANSFORM
C-----------------------------------------------------------------
      M1=2
20    M2=2*M1
      M3=M1/2
      N1=N/M2
      DO 25 J1=1,N1
      N8=(J1-1)*M2
      N7=N8+M1+1
      IF(M3.EQ.1) GOTO 22
      DO 21 K=1,M3
      N2=N7+2*(K-1)
      N3=N2+1
      J=M3-1+K
      W1=A(J)*F(N2)
      W2=B(J)*F(N3)
      W3=C(J)*(F(N2)+F(N3))
      F(N2)=W2+W3
21    F(N3)=W1-W3
      GOTO 24
22    N3=N7+1
      W=F(N7)
      F(N7)=C(1)*(W+F(N3))
      F(N3)=C(1)*(W-F(N3))
24    DO 25 K=1,M1
      N2=N8+K
      N3=N2+M1
      W=F(N2)
      F(N2)=W+F(N3)
25    F(N3)=W-F(N3)
      M1=2*M1
      IF(M1.LT.N) GOTO 20
      N1=N/4
      DO 26 I=1,N1
      J=2*(I-1)+1
      K=N-1+I
      K1=N*3/2-I
      N2=N-J
      N3=N2+1
      N4=J+1
      W1=A(K)*F(J)
      W2=B(K)*F(N4)
      W3=C(K)*(F(J)+F(N4))
      W4=A(K1)*F(N2)
      W5=B(K1)*F(N3)
      W6=C(K1)*(F(N2)+F(N3))
      F(J)=W2+W3
      F(N3)=W1-W3
      F(N2)=W5+W6
26    F(N4)=W4-W6
      GOTO 40
30    W1=A(2)*F(1)
      W2=B(2)*F(2)
      W3=C(2)*(F(1)+F(2))
      F(1)=W2+W3
      F(2)=W1-W3
40    RETURN
      END


C*****************************************************************************
C
C     SUBROUTINE FOR INITIATION OF COEFFICIENTS
C
C*****************************************************************************
```

```
      SUBROUTINE COEF(A,B,C,N)
      DIMENSION A(3*N/16),B(3*N/16),C(3*N/16)
      M=2
      N2=3*N/16
      P=4.*ATAN(1.)
      L=1
1     M=2*M
      K=1
2     CONTINUE
      C(L)=COS(K*P/M)
      B(L)=SIN(K*P/M)
      A(L)=B(L)+C(L)
      B(L)=B(L)-C(L)
      L=L+1
      K=K+4
      IF(L.EQ.N2) RETURN
      IF(K.LT.M) GOTO 2
      IF(L.LT.N2) GOTO 1
      RETURN
      END
```

ACKNOWLEDGMENT

The author wishes to acknowledge the Digital Analysis Laboratory of the University of Arizona directed by Prof. B. R. Hunt for providing the facilities. The encouragement given by Prof. Hunt was an important factor in the completion of this paper.

REFERENCES

[1] Z. Wang, "Harmonic analysis with a real frequency function, I: Aperiodic case," *Appl. Math. Comput.*, vol. 9, pp. 53–73, 1981.
[2] —, "Harmonic analysis with a real frequency function, II: Period and bounded cases," *Appl. Math. Comput.*, vol. 9, pp. 153–163, 1981.
[3] —, "Harmonic analysis with a real frequency function, III: Data sequence," *Appl. Math. Comput.*, vol. 9, pp. 245–255, 1981.
[4] W. K. Pratt, *Digital Image Processing.* New York: Wiley, 1978, ch. 10, 23.
[5] H. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transform," *IEEE Trans. Comput.*, vol. C-23, pp. 90–93, 1974.
[6] H. B. Kekre and J. K. Solanki, "Comparative performance of various trigonometric unitary transforms for transform image coding," *Int. J. Electron.*, vol. 44, pp. 305–315, 1978.
[7] A. K. Jain, "A sinusoidal family of unitary transform," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-1, pp. 356–365, 1979.
[8] W. H. Chen, C. H. Smith, and S. C. Fralick, "A fast computational algorithm for the discrete cosine transform," *IEEE Trans. Commun.*, vol. COM-25, pp. 1004–1009, 1977.
[9] P. Yip and K. R. Rao, "A fast computational algorithm for the discrete sine transform," *IEEE Trans. Commun.*, vol. COM-28, pp. 304–307, 1980.
[10] Z. Wang, "A fast algorithm for the discrete sine transform implemented by the fast cosine transform," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-30, pp. 814–815, 1982.
[11] —, "Reconsideration of 'A fast computational algorithm for the discrete cosine transform,'" *IEEE Trans. Commun.*, vol. COM-31, pp. 121–123, 1983.
[12] Z. Wang and B. R. Hunt, "The discrete cosine transform—A new version," in *Proc. Int. Conf. Acoust., Speech, Signal Processing*, 1983.
[13] A. K. Jain, "A fast Karhunen-Loeve transform for a class of stochastic processes," *IEEE Trans. Commun.*, vol. COM-24, pp. 1023–1029, 1976.
[14] L. R. Rabiner and B. Gold, *Theory and Application of Digital Signal Processing.* Englewood Cliffs, NJ: Prentice-Hall, 1975.
[15] D. P. Kolba and T. W. Parks, "A prime factor FFT algorithm using high-speed convolution," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-25, pp. 281–294, 1977.
[16] R. C. Agarwal, "Comments on 'A prime factor FFT algorithm using high-speed convolution,'" *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-26, p. 254, 1978.
[17] R. D. Preuss, "Very fast computation of the radix-2 discrete Fourier transform," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-30, pp. 595–607, 1982.
[18] Z. Wang and B. R. Hunt, "The discrete *W* transform," to be published.

Zhongde Wang (M'83) was born in Nanjing, Jiangsu, China on March 19, 1937. He graduated in physics from Yunnan University, China, in 1960.

Since 1960, he has been with the Kunming Institute of Physics, Kunming, China, as a Research Assistant and Research Engineer. From May 1980 to February 1983 he worked at the Digital Image Analysis Laboratory of the University of Arizona, Tucson, as a Visiting Scholar and as a Research Associate. During that period, he was awarded a grant by the U.S. National Science Foundation. He returned to Kunming Institute of Physics in 1983, continuing his research there. His main research interests are algorithms, signal processing, and image processing.

Mr. Wang is a member of the Chinese Physical Society, the Chinese Electronical Society, and the American Mathematical Society.