Symmetric Convolution and the Discrete Sine and Cosine Transforms

Stephen A. Martucci, *Member, IEEE*

Abstract—**This paper discusses the use of *symmetric convolution* and the discrete sine and cosine transforms (DST's & DCT's) for general digital signal processing. The operation of symmetric convolution is a formalized approach to convolving symmetrically extended sequences. The result is the same as that obtained by taking an inverse discrete trigonometric transform (DTT) of the product of the forward DTT's of those two sequences. There are 16 members in the family of DTT's. Each provides a representation for a corresponding distinct type of *symmetric-periodic sequence*. In this paper, we define symmetric convolution, relate the DST's and DCT's to symmetric-periodic sequences, and then use these principles to develop simple but powerful convolution-multiplication properties for the entire family of DST's and DCT's. Symmetric convolution can be used for discrete linear filtering when the filter is symmetric or antisymmetric. The filtering will be efficient because fast algorithms exist for all versions of the DTT's. Conventional linear convolution is possible if we first zero-pad the input data. Symmetric convolution and its fast implementation using DTT's are now an alternative to circular convolution and the DFT.**

I. Introduction

THE discrete sine and cosine transforms (DST's & DCT's) have received considerable attention lately because of their predominant use in transform coding. The type-2 DCT is at the core of emerging standards for image and video compression, *e.g.*, JPEG, H.261, and MPEG [1]–[3]. The type-1 DST constitutes the basis of a technique called *Recursive Block Coding* [4]. The type-4 DCT and type-4 DST are both found in a fast implementation of *Lapped Orthogonal Transforms* for coding [5].

It is well-known that the discrete Fourier transform (DFT) possesses a convolution-multiplication property whereby a circular convolution of two finite sequences can be computed efficiently by means of a transform-domain multiplication. Although the DST's and DCT's are related to the DFT, a general way to use these transforms for performing convolution has only recently been shown [6], [7]. Some earlier attempts have been reported, but those results were awkward to use or incomplete [8]–[10].

In this paper we show that simple but powerful convolution-multiplication properties do exist for the entire family of discrete trigonometric transforms (DTT's). The convolution performed is a special type, to which we give the name *symmetric convolution*. The operation of symmetric convolution is a formalized approach to convolving symmetrically extended sequences. With proper zero-padding of one of the sequences, symmetric convolution can be used to perform linear convolution in much the same way that circular convolution can. As we shall see, the theory that relates symmetric convolution to the DTT's parallels that which relates circular convolution to the DFT.

We restrict the discussion in this paper to one-dimensional sequences. Concepts also apply to higher-dimensional sequences except that points of symmetry would instead be lines, planes, or hyperplanes of symmetry. We can use symmetric convolution to implement both separable and nonseparable multidimensional FIR filters; the only requirement is that there must be symmetry in every dimension. Other FIR filters can be accommodated if they are first decomposed into their symmetric and antisymmetric parts.

The paper is organized as follows. In Section II, we reintroduce the generalized discrete Fourier transform and discuss some of its properties. Next, we define in Section III the 16 members of the family of discrete trigonometric transforms. We look at symmetric-periodic sequences and the DTT's in Section IV. We examine the subject of convolution-multiplication properties for discrete transforms, introduce symmetric convolution, and present the convolution-multiplication properties for the DTT's in Section V. Then, in Section VI, we explain how to use symmetric convolution to perform a type of discrete filtering. In Section VII, we show how a symmetric convolution can be turned into a linear convolution. In Section VIII, we review the earlier efforts to find convolution-multiplication properties for the DTT's. We mention some applications for symmetric convolution and the DTT's in Section IX, and then give the conclusion in Section X.

II. The Generalized Discrete Fourier Transform and Generalized-Periodic Sequences

Throughout this paper we use calligraphic letters to denote transform and other operators and some special sequences. We define all transforms as matrices that left-multiply the input sequence $x(n)$ represented as a column vector $\mathbf{x}$. If the support of $x(n)$ is $n = 0, 1, \cdots, M-1$, then

$$\mathbf{x} = [x(0)\ x(1)\ \cdots\ x(M-1)]^T. \tag{1}$$

The discrete Fourier transform can be generalized to allow shifts in either or both indices. The resulting invertible

Manuscript received May 10, 1992; revised June 2, 1993. The associate editor coordinating the review of this paper and approving it for publication was Prof. James Cooley. This work was supported in part by the Joint Services Electronics Program under Contract DAAL-03-90-C-0004.

The author was with the School of Electrical Engineering, Georgia Institute of Technology, Atlanta, GA , USA 30332. He is now with the David Sarnoff Research Center, CN5300, Princeton, NJ, USA 08543-5300.

IEEE Log Number 9216665.

transform is called the *generalized discrete Fourier transform* (GDFT) [11]. The entry at row m and column n of the GDFT matrix of order M is given by

$$[\mathcal{G}_{a,b}]_{mn} = \exp\left(-j\frac{2\pi}{M}(m+a)(n+b)\right) \qquad m,n = 0,1,\cdots,M-1 \quad (2)$$

where a and b are real numbers.[1] The inverse GDFT matrix is the scaled Hermitian transpose of the forward matrix.

For convenience in notation and interpretation, we assume for the remainder of this section that inputs to forward transforms are in the time domain and outputs are in the frequency domain. Of interest to us are four special forms of the GDFT that arise when a and b take on the values 0 or $\frac{1}{2}$. With $a = 0$ and $b = 0$ the GDFT reduces to the normal DFT. Setting $a = \frac{1}{2}$ and $b = 0$ defines a DFT with a 1/2-sample advance in the frequency domain, a version of the DFT previously reported as the Odd DFT (ODFT) [12], [13], but which we prefer to call the Odd Frequency DFT (OFDFT). A 1/2-sample delay in the time domain occurs when $a = 0$ and $b = \frac{1}{2}$ giving a version of the DFT we call the Odd Time DFT (OTDFT). Both $a = \frac{1}{2}$ and $b = \frac{1}{2}$ produces a form called the Odd-time Odd-frequency DFT (O^2DFT) [14]. These are the only forms of the GDFT that we use in this paper.

The name Odd Frequency DFT comes from the following observation: If the sequence $X(m)$ is the length-M OFDFT of the sequence $x(n)$, then the samples of $X(m)$ are the same as the odd-indexed samples of the length-$2M$ DFT of $x'(n)$ where $x'(n)$ is $x(n)$ zero-padded on the right to length $2M$. A similar comment holds for the Inverse Odd Time DFT (IOTDFT): If the sequence $x(n)$ is the length-M IOTDFT of the sequence $X(m)$, then the samples of $x(n)$ are equal to twice the odd-indexed samples of the length-$2M$ IDFT of $X'(m)$ where $X'(m)$ is $X(m)$ zero-padded on the right to length $2M$.

The inverse matrices of these special forms of the GDFT are related to the forward matrices in the following ways

$$\left[\mathcal{G}_{0,0}\right]^{-1} = \frac{1}{M}\left[\mathcal{G}_{0,0}\right]^{H} = \frac{1}{M}\left[\mathcal{G}_{0,0}\right]^{*} \quad (3)$$

$$\left[\mathcal{G}_{\frac{1}{2},0}\right]^{-1} = \frac{1}{M}\left[\mathcal{G}_{\frac{1}{2},0}\right]^{H} = \frac{1}{M}\left[\mathcal{G}_{0,\frac{1}{2}}\right]^{*} \quad (4)$$

$$\left[\mathcal{G}_{0,\frac{1}{2}}\right]^{-1} = \frac{1}{M}\left[\mathcal{G}_{0,\frac{1}{2}}\right]^{H} = \frac{1}{M}\left[\mathcal{G}_{\frac{1}{2},0}\right]^{*} \quad (5)$$

$$\left[\mathcal{G}_{\frac{1}{2},\frac{1}{2}}\right]^{-1} = \frac{1}{M}\left[\mathcal{G}_{\frac{1}{2},\frac{1}{2}}\right]^{H} = \frac{1}{M}\left[\mathcal{G}_{\frac{1}{2},\frac{1}{2}}\right]^{*} \quad (6)$$

where ^{H} denotes Hermitian transpose and * denotes complex conjugation.

Each of the above four special forms of the length-M GDFT defines representations for a length-M sequence as infinite sequences that are periodic with period M both before and after transformation. Here we are using the term *periodic* in a general sense to describe sequences that are either *strictly periodic* or *antiperiodic*. An antiperiodic sequence with generalized period M satisfies

$$x_a(n) = -x_a(n+M) \qquad \text{for all } n. \quad (7)$$

[1] This definition for the GDFT swaps the parameters a and b as compared to the definition in [11].

Notice that $x_a(n)$ is antiperiodic with period M and strictly periodic with period $2M$. The four possible combinations of implicit periodicities before and after transformation and the corresponding GDFT's are summarized in Table I. These four special forms of the GDFT can be better understood by looking at the periodic and antiperiodic sequences they represent. Some important properties that we will need later are:

- The product of two strictly periodic sequences, both with period M, is strictly periodic with period M.
- The product of two antiperiodic sequences, both with period M, is strictly periodic with period M.
- The product of a strictly periodic sequence and an antiperiodic sequence, both with period M, is antiperiodic with period M.
- The result of the periodic convolution of two strictly periodic sequences, both with period M, is a strictly periodic sequence with period M. This property follows directly from the definition for periodic convolution

$$\tilde{x}(n) \circledast \tilde{y}(n) = \sum_{k=0}^{M-1} \tilde{x}(k)\tilde{y}(n-k) = \sum_{k=0}^{M-1} \tilde{y}(k)\tilde{x}(n-k) \quad (8)$$

  where $\tilde{x}(n)$ and $\tilde{y}(n)$ are both strictly periodic with period M.
- The result of the periodic convolution of two antiperiodic sequences, both with period M, is an antiperiodic sequence with period M. This property also follows from (8) except that in this case $\tilde{x}(n)$ and $\tilde{y}(n)$ are both antiperiodic with period M.
- The periodic convolution of a strictly periodic sequence and an antiperiodic sequence, both with period M, cannot be computed according to (8). However, because both sequences are also strictly periodic with period $2M$, we can compute their periodic convolution by performing the summation over the period $2M$. The result is always a sequence of zeros.
- The circular convolution of two length-M sequences that have been defined over the same interval is equivalent to the corresponding period of a periodic convolution of strictly periodic sequences with period M. We can compute the circular convolution of $x(n)$ and $y(n)$, $n = 0,1,\cdots,M-1$, from

$$x(n)\textcircled{c}y(n) = \sum_{k=0}^{n} x(k)y(n-k) + \sum_{k=n+1}^{M-1} x(k)y(n-k+M). \quad (9)$$

- The skew-circular convolution of two length-M sequences that have been defined over the same interval is equivalent to the corresponding period of a periodic convolution of antiperiodic sequences with period M. We can compute the skew-circular convolution of $x(n)$

TABLE I
PERIODICITY PROPERTIES OF GDFT'S

$\mathcal{G}_{a,b}^{-1}$	$\mathcal{G}_{a,b}$	*Periodicity before transform*	*Periodicity after transform*
$\mathcal{G}_{0,0}^{-1}$ (IDFT)	$\mathcal{G}_{0,0}$ (DFT)	periodic	periodic
$\mathcal{G}_{\frac{1}{2},0}^{-1}$ (IOFDFT)	$\mathcal{G}_{0,\frac{1}{2}}$ (OTDFT)	periodic	antiperiodic
$\mathcal{G}_{0,\frac{1}{2}}^{-1}$ (IOTDFT)	$\mathcal{G}_{\frac{1}{2},0}$ (OFDFT)	antiperiodic	periodic
$\mathcal{G}_{\frac{1}{2},\frac{1}{2}}^{-1}$ (IO^{2}DFT)	$\mathcal{G}_{\frac{1}{2},\frac{1}{2}}$ (O^{2}DFT)	antiperiodic	antiperiodic

and $y(n)$, $n = 0, 1, \cdots, M-1$, from

$$x(n) \circledS y(n) = \sum_{k=0}^{n} x(k)y(n-k) - \sum_{k=n+1}^{M-1} x(k)y(n-k+M). \quad (10)$$

III. THE DISCRETE TRIGONOMETRIC TRANSFORMS

The family of discrete trigonometric transforms comprises eight versions of the discrete cosine transform and eight versions of the discrete sine transform. These have been categorized by Wang and are tabulated in [15]–[17]. Each transform is identified as cosine or sine, even or odd, and type 1, 2, 3, or 4. The origins of these transforms can be found in [18]–[23].

Every Wang matrix is orthogonal with entry at row m and column n of the general form

$$[\mathcal{T}_{\mathrm{w}}]_{mn} = A\, w_1(m)\, w_2(n) t(m,n). \quad (11)$$

The term $t(m,n)$ is the transform kernel. The term $w_1(m)$ is a weighting function needed in some of the transforms to make the column vectors orthogonal to one another. The weighting function $w_2(n)$ is needed in some of the transforms to make the row vectors orthogonal to one another. The scalar A is a final multiplier that normalizes the rows and columns to produce an orthogonal matrix.

We need to modify the Wang matrices to put them in a form that better suits our needs. First, we recast (11) as a product of nonsingular square matrices

$$[\mathcal{T}_{\mathrm{w}}] = A[W_1][T][W_2] \quad (12)$$

where $[W_1]$ and $[W_2]$ are diagonal matrices that left- and right-multiply the kernel $[T]$, respectively. Next, because of orthogonality, we can write

$$A^2[W_1][T][W_2][W_2][T]^T[W_1] = [I] \quad (13)$$

where $[I]$ is the identity matrix. We factor A^2 into $A^2 = A_f A_i$, where A_f is not necessarily equal to A_i, and associate A_f with the forward transform and A_i with the inverse. Then, we use the property that a nonsingular matrix commutes with its inverse and rewrite (13) as

$$A_f[T][W_2][W_2]A_i[T]^T[W_1][W_1] = [I] \quad (14)$$

from which we derive the new forward and inverse transforms

$$[T] = A_f[T][W_2]^2 \quad (15)$$

$$[T^{-1}] = A_i[T]^T[W_1]^2 \quad (16)$$

The entries at row m and column n of these new transforms are

$$[T]_{mn} = A_f\, w_2^2(n) t(m,n) \quad (17)$$

$$[T^{-1}]_{mn} = A_i\, w_1^2(n) t(n,m). \quad (18)$$

We call these new formulations for the discrete sine and cosine transforms the *convolution form* of the transforms. They can all be found in the appendix. When in convolution form, the transform matrices may no longer be orthogonal. However, the relations that existed between each inverse transform matrix and its own or another forward transform matrix remain, but with a scaling factor. These relations are also given in the appendix.

Our main reasons for redefining the trigonometric transforms as we have done are to make the connection to the GDFT, as defined in Section II, more direct and so that the convolution-multiplication properties will not require any extra scaling factors or weighting functions. These points will be brought out in later sections. Also, note that the weighting functions for both the forward and inverse transforms now only right-multiply the matrix kernels. Therefore, the weights can instead left-multiply the input data vector to each transform, and the actual transform can be implemented without any weights. Many of the available algorithms for computing the DST and DCT also assume such a separation of the weighting functions from the kernel of the transforms. For this reason, fast algorithms that exist for computing the Wang forms of the DTT's can be used for these new forms as well. Notable algorithms for the DTT's can be found in [5], [7], [15], [17], [24]–[33].

Because the kernels $t(m,n)$ in some of the transforms in the appendix evaluate to zero for some values of the indices m and n, we have specified the ranges for m and n to avoid those values. Such an *exclusive index range* is therefore one of the following: (a) $0, 1, \cdots, N$; (b) $0, 1, \cdots, N-1$; (c) $0, 1, \cdots, N-2$; (d) $1, 2, \cdots, N$; or (e) $1, 2, \cdots, N-1$. For consistency we use the term *exclusive index range* even for those transforms that do not have terms that evaluate to zero. Over the exclusive index ranges for its rows and columns each transform matrix is square and invertible.

It is not necessary for the exclusive ranges for both m and n for a particular transform to be the same. In the new definitions for the sine transforms $S_{2e}, S_{3e}, S_{2o}, S_{3o}$, and their inverses, the exclusive ranges for m and n differ. To keep the input and output index ranges the same, Wang modified the kernels of these particular transforms. Specifically, the definitions for $S_N^{\mathrm{II}E}, S_N^{\mathrm{III}E}, S_{N-1}^{\mathrm{II}O}$, and $S_{N-1}^{\mathrm{III}O}$ given in [15], [16] contain the term $(m - \frac{1}{2})$ or $(n - \frac{1}{2})$ instead of the $(m + \frac{1}{2})$ or $(n + \frac{1}{2})$ found in the other transforms. In the transform definitions in the appendix, we have undone this modification; instead, we allow different ranges for m and n. With this new formulation, there is a direct link between all DTT's and the GDFT. The result is a consistent approach to symmetric convolution and its implementation using DTT's.

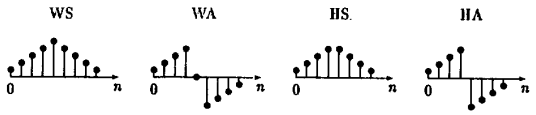


Fig. 1. Examples of symmetry types in a sequence.

IV. Symmetric-Periodic Sequences and the Discrete Trigonometric Transforms

A. Characteristics

A *symmetric-periodic sequence* (SPS) is an infinite sequence that is both symmetric and periodic. Just as the special forms of the GDFT provide representations for generalized-periodic sequences, the various forms of the DTT's represent symmetric-periodic sequences. Any SPS can be written as a linear combination of either only sine or only cosine sequences since sine and cosine sequences are themselves symmetric-periodic sequences. Therefore, to better understand the DTT's, we examine more closely the characteristics of SPS's.

We can classify SPS's according to the types of the symmetries found within the sequences. Symmetry within any sequence must be of one of four types: *whole-sample symmetry* (WS), *whole-sample antisymmetry* (WA), *half-sample symmetry* (HS), or *half-sample antisymmetry* (HA) [34]. The meaning of the terms *symmetry* and *antisymmetry* should be clear. The designations *whole-sample* and *half-sample* are new terms that describe where that point of symmetry lies, either coincident with one of the samples of the sequence or at a theoretical half-sample half-way between two samples. The symmetry in an odd-length symmetric sequence is either WS or WA; in an even-length symmetric sequence it is either HS or HA. We give an example finite sequence of each type in Fig. 1.

If an infinite sequence has more than one point of symmetry, then it has an infinite number of points of symmetry, it is periodic, and it is an SPS. The entire sequence is symmetric or antisymmetric about each of these points of symmetry. The points of symmetry are either all of the same type or of two different types. If the types are different, then they alternate along the length of the SPS.

A finite sequence $x(n)$ can be converted into an infinite SPS by symmetrically extending at each end in one of the above four possible ways and continuing that extension indefinitely. The half-sample extensions are straightforward; the whole-sample extensions require further discussion. When the extension is WS, the new point of symmetry being created is the corresponding endpoint of $x(n)$. This sample is not repeated in the extension. For the case of WA extension, however, the point of symmetry must have the value zero. In the most general case, neither of the endpoints of the sequence $x(n)$ is zero. Therefore, we assume that the point of symmetry is one sample beyond the endpoint of $x(n)$ and putting the value zero there is part of the extension process. If the sequence $x(n)$ already has the value zero at an endpoint and we do not want to repeat that point, then we must remove that point before we perform the extension.

Four choices for extension at each of two endpoints means that a total of 16 distinct SPS's can be created. There is a one-to-one correspondence between each type of SPS and each type of DTT. However, if these 16 sequences are derived from the same finite sequence, not all of the corresponding transforms are members of the same family. We will have more to say about this later.

Within an SPS there is a finite subsequence of samples that can be used to generate all other samples. We call these the *representative samples*. For an SPS with K representative samples, it is not true that any arbitrary subsequence of K consecutive samples of the SPS contains all K representative samples. We therefore find it necessary to make a precise definition of the *base period* of an SPS and of where the representative samples lie within the base period. We call this the *standard form* of the symmetric-periodic sequence.

The base period of an SPS in standard form is the interval $n = 0, 1, \cdots, M-1$, where M is the generalized period. We assume that this M is the smallest possible period for the sequence. There are three possibilities for the number of representative samples: $N-1, N$, or $N+1$, where the relation of M to N is either $M = 2N$ or $M = 2N-1$. The index range of the representative samples is one of the following: (a) $0, 1, \cdots, N$; (b) $0, 1, \cdots, N-1$; (c) $0, 1, \cdots, N-2$; (d) $1, 2, \cdots, N$; or (e) $1, 2, \cdots, N-1$. There are two points of symmetry (POS's) associated with the base period, a left point of symmetry (LPOS) and a right point of symmetry (RPOS), and between them lie the representative samples. At each POS we find one of the four previously defined types of symmetry: WS, WA, HS, or HA. By definition, only when the symmetry is WS is the sample at a point of symmetry included as one of the representative samples.

A pair of symmetry types in conjunction with the representative samples is sufficient to fully specify an SPS. We can therefore name each SPS as the concatenation of the mnemonic for the LPOS with that for the RPOS, *e.g.*, WSWS, HAHA, WAHS, etc. Equivalently, we can classify the SPS's by the type of the generating DST or DCT (or inverse) for each.

We can think of an SPS as the infinite-length representation of its representative samples, or we can say that the representative samples together with the type of the LPOS and the RPOS are a finite-length representation of the infinite SPS. As is true of any infinite sequence, an SPS has a *representative length* equal to the minimum number of samples needed to represent the entire sequence; those samples are the representative samples. The representative samples of an SPS are the same as the samples of the finite sequence from which the SPS could have been created.

In Table II we give a number of important characteristics of the 16 types of SPS's in standard form and their corresponding DTT's. These data is repeated in Table III, but grouped differently. A dominant element in the tables is the parameter N. A common value of N defines a complete family of 16 SPS's and thus a complete family of 16 DTT's. The periodicity M of the SPS's is derived from this N; either $M = 2N$ or $M = 2N-1$. The two possibilities for M divides the family of DTT's into two groups: the eight even DTT's and the eight odd DTT's. Each DTT is fully specified by its N, M, trigonometric

TABLE II
CHARACTERISTICS OF DTT'S AND SPS'S, CONVOLUTION DOMAIN

$\mathcal{T}^{-1}$	SPS	length	index range	LPOS	RPOS	P or A	$\mathcal{G}_{a,b}^{-1}$
$\mathcal{C}_{1e}^{-1}$	WSWS	$N+1$	$0 \to N$	0	N	P	$\mathcal{G}_{0,0}^{-1}$
$\mathcal{S}_{1e}^{-1}$	WAWA	$N-1$	$1 \to N-1$	0*	N*	P	$-j\mathcal{G}_{0,0}^{-1}$
$\mathcal{C}_{2e}^{-1}$	HSHS	N	$0 \to N-1$	$-\frac{1}{2}$	$N-\frac{1}{2}$	P	$\mathcal{G}_{0,\frac{1}{2}}^{-1}$
$\mathcal{S}_{2e}^{-1}$	HAHA	N	$0 \to N-1$	$-\frac{1}{2}$	$N-\frac{1}{2}$	P	$-j\mathcal{G}_{0,\frac{1}{2}}^{-1}$
$\mathcal{C}_{3e}^{-1}$	WSWA	N	$0 \to N-1$	0	N*	A	$\mathcal{G}_{\frac{1}{2},0}^{-1}$
$\mathcal{S}_{3e}^{-1}$	WAWS	N	$1 \to N$	0*	N	A	$-j\mathcal{G}_{\frac{1}{2},0}^{-1}$
$\mathcal{C}_{4e}^{-1}$	HSHA	N	$0 \to N-1$	$-\frac{1}{2}$	$N-\frac{1}{2}$	A	$\mathcal{G}_{\frac{1}{2},\frac{1}{2}}^{-1}$
$\mathcal{S}_{4e}^{-1}$	HAHS	N	$0 \to N-1$	$-\frac{1}{2}$	$N-\frac{1}{2}$	A	$-j\mathcal{G}_{\frac{1}{2},\frac{1}{2}}^{-1}$
$\mathcal{C}_{1o}^{-1}$	WSHS	N	$0 \to N-1$	0	$N-\frac{1}{2}$	P	$\mathcal{G}_{0,0}^{-1}$
$\mathcal{S}_{1o}^{-1}$	WAHA	$N-1$	$1 \to N-1$	0*	$N-\frac{1}{2}$	P	$-j\mathcal{G}_{0,0}^{-1}$
$\mathcal{C}_{2o}^{-1}$	HSWS	N	$0 \to N-1$	$-\frac{1}{2}$	$N-1$	P	$\mathcal{G}_{0,\frac{1}{2}}^{-1}$
$\mathcal{S}_{2o}^{-1}$	HAWA	$N-1$	$0 \to N-2$	$-\frac{1}{2}$	$N-1$*	P	$-j\mathcal{G}_{0,\frac{1}{2}}^{-1}$
$\mathcal{C}_{3o}^{-1}$	WSHA	N	$0 \to N-1$	0	$N-\frac{1}{2}$	A	$\mathcal{G}_{\frac{1}{2},0}^{-1}$
$\mathcal{S}_{3o}^{-1}$	WAHS	$N-1$	$1 \to N-1$	0*	$N-\frac{1}{2}$	A	$-j\mathcal{G}_{\frac{1}{2},0}^{-1}$
$\mathcal{C}_{4o}^{-1}$	HSWA	$N-1$	$0 \to N-2$	$-\frac{1}{2}$	$N-1$*	A	$\mathcal{G}_{\frac{1}{2},\frac{1}{2}}^{-1}$
$\mathcal{S}_{4o}^{-1}$	HAWS	N	$0 \to N-1$	$-\frac{1}{2}$	$N-1$	A	$-j\mathcal{G}_{\frac{1}{2},\frac{1}{2}}^{-1}$

TABLE III
CHARACTERISTICS OF DTT'S AND SPS'S, MULTIPLICATION DOMAIN

$\mathcal{T}$	SPS	length	index range	LPOS	RPOS	P or A	$\mathcal{G}_{a,b}$
$\mathcal{C}_{1e}$	WSWS	$N+1$	$0 \to N$	0	N	P	$\mathcal{G}_{0,0}$
$\mathcal{S}_{1e}$	WAWA	$N-1$	$1 \to N-1$	0*	N*	P	$j\mathcal{G}_{0,0}$
$\mathcal{C}_{2e}$	WSWA	N	$0 \to N-1$	0	N*	A	$\mathcal{G}_{0,\frac{1}{2}}$
$\mathcal{S}_{2e}$	WAWS	N	$1 \to N$	0*	N	A	$j\mathcal{G}_{0,\frac{1}{2}}$
$\mathcal{C}_{3e}$	HSHS	N	$0 \to N-1$	$-\frac{1}{2}$	$N-\frac{1}{2}$	P	$\mathcal{G}_{\frac{1}{2},0}$
$\mathcal{S}_{3e}$	HAHA	N	$0 \to N-1$	$-\frac{1}{2}$	$N-\frac{1}{2}$	P	$j\mathcal{G}_{\frac{1}{2},0}$
$\mathcal{C}_{4e}$	HSHA	N	$0 \to N-1$	$-\frac{1}{2}$	$N-\frac{1}{2}$	A	$\mathcal{G}_{\frac{1}{2},\frac{1}{2}}$
$\mathcal{S}_{4e}$	HAHS	N	$0 \to N-1$	$-\frac{1}{2}$	$N-\frac{1}{2}$	A	$j\mathcal{G}_{\frac{1}{2},\frac{1}{2}}$
$\mathcal{C}_{1o}$	WSHS	N	$0 \to N-1$	0	$N-\frac{1}{2}$	P	$\mathcal{G}_{0,0}$
$\mathcal{S}_{1o}$	WAHA	$N-1$	$1 \to N-1$	0*	$N-\frac{1}{2}$	P	$j\mathcal{G}_{0,0}$
$\mathcal{C}_{2o}$	WSHA	N	$0 \to N-1$	0	$N-\frac{1}{2}$	A	$\mathcal{G}_{0,\frac{1}{2}}$
$\mathcal{S}_{2o}$	WAHS	$N-1$	$1 \to N-1$	0*	$N-\frac{1}{2}$	A	$j\mathcal{G}_{0,\frac{1}{2}}$
$\mathcal{C}_{3o}$	HSWS	N	$0 \to N-1$	$-\frac{1}{2}$	$N-1$	P	$\mathcal{G}_{\frac{1}{2},0}$
$\mathcal{S}_{3o}$	HAWA	$N-1$	$0 \to N-2$	$-\frac{1}{2}$	$N-1$*	P	$j\mathcal{G}_{\frac{1}{2},0}$
$\mathcal{C}_{4o}$	HSWA	$N-1$	$0 \to N-2$	$-\frac{1}{2}$	$N-1$*	A	$\mathcal{G}_{\frac{1}{2},\frac{1}{2}}$
$\mathcal{S}_{4o}$	HAWS	N	$0 \to N-1$	$-\frac{1}{2}$	$N-1$	A	$j\mathcal{G}_{\frac{1}{2},\frac{1}{2}}$

function, and type number. The type number is 1, 2, 3, or 4. The actual value of this parameter is not significant; its purpose is just to classify the transforms. However, there are some similarities between transforms sharing the same type number.

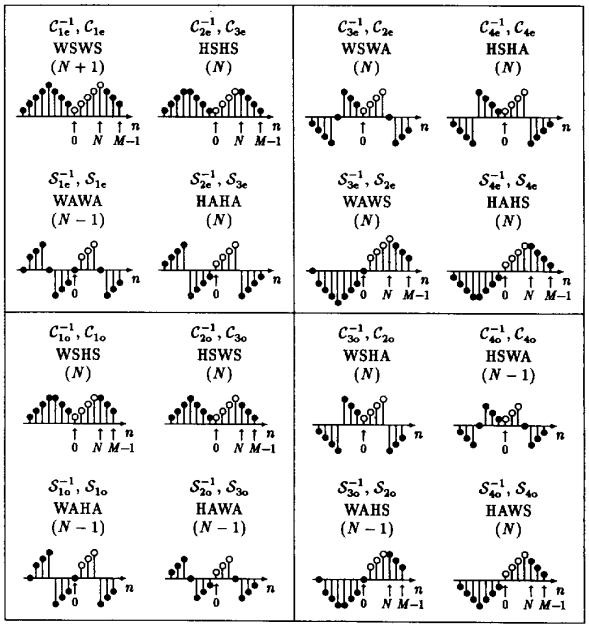


Fig. 2. Example plots showing symmetries of SPS's and DTT's. All SPS's are in standard form with $N = 4$. The representative samples for each SPS are marked by open circles and the representative length is given in parentheses. The SPS's are grouped into convolution classes appropriate for the convolution domain; symmetric convolution is possible only between SPS's within the same class.

Example plots for each of the 16 SPS's and corresponding DTT's are in Fig. 2. These sequences are in standard form and share a common value of $N = 4$. All plots cover the index range $n = -M, \cdots, M-1$. The SPS's are all periodic with period $2M$ and either periodic or antiperiodic with generalized period M where $M = 8$ for the upper eight plots and $M = 7$ for the lower eight. The representative samples for each SPS are marked by open circles and the representative length given in parentheses.

For each entry in Tables II and III we list a DTT or its inverse, an SPS, a length, and an index range. The length is the representative length. The range is the index range of the representative samples of the SPS in standard form and is the same as the exclusive output index range of the corresponding DTT. We also give the index of the left point of symmetry (LPOS) and of the right point of symmetry (RPOS). An asterisk indicates that the sample at that point of symmetry must have the value zero because the symmetry there is WA.

A symmetric-periodic sequence is generalized-periodic with period M. Therefore, we also show in the tables whether each SPS is periodic or antiperiodic. Knowledge of the periodicity type enables us to use another method for converting a finite sequence into an SPS. Instead of symmetrically extending the finite sequence at both ends we can extend it symmetrically on the right just far enough to specify all values of the base period. In a few cases we may also have to append the value zero at the left. Then we either periodically or antiperiodically extend this base period according to the periodicity type of the SPS.

B. Relationship to Generalized Discrete Fourier Transform

Because they are symmetric and periodic, all SPS's have a representation in terms of the DTT's. Because they are generalized-periodic, all SPS's have an alternative representation in terms of the GDFT. It is not difficult to show that the identical SPS can be generated either from a DTT of a given finite sequence or from the corresponding GDFT of that sequence after it has been symmetrically extended to a base period. This extension is done in accordance with the symmetries associated with the inverse of the DTT.

As an example, given the finite sequence $x(n), n = 0, 1, \cdots, N-1$, we can produce an SPS that is WSWA with period $M = 2N$ by computing $\mathcal{C}_{2e}$ of $x(n)$ for all integer values of its output index. Alternatively, we can use the symmetric extension operator $\mathcal{E}_{\text{HSHS}}$ of Table IV to do an HS extension on the right of $x(n)$ creating $\hat{x}(n), n = 0, 1, \cdots, 2N$. We then compute $\mathcal{G}_{0,\frac{1}{2}}$ of $\hat{x}(n)$ with an infinite output index range. These two SPS's are identical. As a matrix equation, from $\mathbf{x}$ we can generate any L terms of the WSWA-SPS $\mathbf{v}$ by

$$\mathbf{v} = \left[\mathcal{G}_{0,\frac{1}{2}}\right]\left[\mathcal{E}_{\text{HSHS}}\right]\mathbf{x} = \left[\mathcal{C}_{2e}\right]\mathbf{x} \tag{19}$$

where $\mathbf{v}$ is $L \times 1$, $\left[\mathcal{G}_{0,\frac{1}{2}}\right]$ is $L \times 2N$, $\left[\mathcal{E}_{\text{HSHS}}\right]$ is $2N \times N$, $[\mathcal{C}_{2e}]$ is $L \times N$, $\mathbf{x}$ is $N \times 1$, and L is any positive integer. We assume that all index ranges are compatible.

Equation (19) follows from the relationship between $\mathcal{C}_{2e}$ and $\mathcal{G}_{0,\frac{1}{2}}$ that can be derived by manipulating the matrices as follows

$$\begin{aligned}\left[\mathcal{G}_{0,\frac{1}{2}}\right][\mathcal{E}_{\text{HSHS}}] &= \left[\exp\left(-j\frac{\pi m(n+\frac{1}{2})}{N}\right)\right]\begin{bmatrix} 1 & & & \\ & \ddots & & \\ & & & 1 \\ & & & 1 \\ & \cdots & & \\ 1 & & & \end{bmatrix} \\ &= \left[\exp\left(-j\frac{\pi m(n+\frac{1}{2})}{N}\right)\right. \\ &\quad \left.+\exp\left(-j\frac{\pi m(2N-1-n+\frac{1}{2})}{N}\right)\right] \\ &= \left[2\cos\left(\frac{\pi m(n+\frac{1}{2})}{N}\right)\right] \\ &= [\mathcal{C}_{2e}] \qquad m, n = 0, 1, \cdots, N-1.\end{aligned}$$

Following this pattern, we can prove that all DST's and DCT's can be defined directly in terms of their corresponding GDFT as given in the last column of Tables II and III.

V. Convolution-Multiplication Properties

Discrete convolution is a fundamental operation for digital signal processing. Let $\mathcal{T}$ be an invertible transform from one domain to another. Convolution is performed in one domain, referred to as the *convolution domain*; element-by-element multiplication in the other, the *multiplication domain*. We assume that the *forward transform* takes us from the convolution domain to the multiplication domain, and the *inverse transform* takes us back. The essence of a *convolution-multiplication property* is that the inverse transform after element-by-element multiplication gives the same result as the convolution of the original sequences. As an equation

$$x(n) * y(n) = \mathcal{T}_c^{-1}\Big\{\mathcal{T}_a\{x(n)\} \times \mathcal{T}_b\{y(n)\}\Big\} \tag{20}$$

where '$*$' is the convolution operator and '$\times$' denotes element-by-element multiplication of its operands. The notation $\mathcal{T}\{x(n)\}$ tells us to take the specified transform of the sequence $x(n)$. It is, of course, possible to swap the usage of the forward and inverse transforms; in that case, an extra scaling factor may be required in the above equation.

For a convolution-multiplication property to hold we must specify the proper definitions of the convolution operator and of the transforms $\mathcal{T}_a$, $\mathcal{T}_b$, and $\mathcal{T}_c^{-1}$. For example, (20) holds when the convolution performed is circular, $\mathcal{T}_a$ and $\mathcal{T}_b$ are the DFT, and $\mathcal{T}_c^{-1}$ is the IDFT. Although in this example all transforms are the same, it is not mandatory that they be so. We can also satisfy (20) by specifying the appropriate mix of GDFT's and either circular or skew-circular convolution. To establish convolution-multiplication properties for the DTT's, we must define a special type of convolution, which we call *symmetric convolution*. We also must determine the proper mix of specific types of DST's and DCT's.

A. Circular and Skew-Circular Convolution

First we look at the GDFT. Let $u(n) = x(n) \textcircled{c} y(n)$ and $w(n) = x(n) \textcircled{s} y(n)$ as defined by (9) and (10), respectively. Then, the following convolution-multiplication properties hold:

$$u(n) = \mathcal{G}_{0,0}^{-1}\Big\{\mathcal{G}_{0,0}\{x(n)\} \times \mathcal{G}_{0,0}\{y(n)\}\Big\} \tag{21}$$

$$u(n) = \mathcal{G}_{0,\frac{1}{2}}^{-1}\Big\{\mathcal{G}_{0,\frac{1}{2}}\{x(n)\} \times \mathcal{G}_{0,0}\{y(n)\}\Big\} \tag{22}$$

$$u(n-1) = \mathcal{G}_{0,0}^{-1}\Big\{\mathcal{G}_{0,\frac{1}{2}}\{x(n)\} \times \mathcal{G}_{0,\frac{1}{2}}\{y(n)\}\Big\} \tag{23}$$

$$w(n) = \mathcal{G}_{\frac{1}{2},0}^{-1}\Big\{\mathcal{G}_{\frac{1}{2},0}\{x(n)\} \times \mathcal{G}_{\frac{1}{2},0}\{y(n)\}\Big\} \tag{24}$$

$$w(n) = \mathcal{G}_{\frac{1}{2},\frac{1}{2}}^{-1}\Big\{\mathcal{G}_{\frac{1}{2},\frac{1}{2}}\{x(n)\} \times \mathcal{G}_{\frac{1}{2},0}\{y(n)\}\Big\} \tag{25}$$

$$w(n-1) = \mathcal{G}_{\frac{1}{2},0}^{-1}\Big\{\mathcal{G}_{\frac{1}{2},\frac{1}{2}}\{x(n)\} \times \mathcal{G}_{\frac{1}{2},\frac{1}{2}}\{y(n)\}\Big\} \tag{26}$$

where the implied periodicity of $u(n)$ is periodic and that of $w(n)$ is antiperiodic. We use these implied periodicities to evaluate the delayed sequences $u(n-1)$ and $w(n-1)$ at $n = 0$. Notice the mix of transform types in some of the expressions. Equation (21) is well-known. Equation (24) was first reported in [12]. The remaining equations are new. These properties follow in a straightforward way from the definitions for the GDFT and the properties given in Section II.

B. Symmetric Convolution

In order to present convolution-multiplication properties for the DTT's, we must first give a precise definition of symmetric convolution. Symmetric convolution is both the convolution of SPS's and the convolution of the finite sequences those

SPS's represent. The finite inputs to symmetric convolution are the representative samples of SPS's in standard form. The finite output from symmetric convolution is the sequence of representative samples of an SPS either in standard form or in standard form advanced by one sample depending on the type of the symmetric convolution.[2] All these SPS's share the same M and N, but their representative lengths may differ because those vary according to SPS type.

To express the generic symmetric convolution of the finite sequences $x(n)$ and $y(n)$ we introduce the notation

$$w(n) = x(n) <^{\mathbf{S}}_{\mathbf{C}}> y(n). \tag{27}$$

The notation can be made more specific by including within the operator a designation for the type of symmetric convolution being performed.

We can define symmetric convolution in terms of a conventional convolution sum that has been suitably modified to incorporate the implied symmetric extensions to both operands. An alternative but equivalent formulation is as the windowed circular or skew-circular convolution of the inputs $x(n)$ and $y(n)$ after their symmetric extension to a base period. A third equivalent means is as the inverse trigonometric transform of the element-by-element product of the forward trigonometric transforms of the inputs $x(n)$ and $y(n)$.

If we want to define symmetric convolution as a direct convolution sum, we must give a different equation for each type. Each equation would contain four summation terms. We can avoid this complexity by defining symmetric convolution in terms of the well-understood circular and skew-circular forms of convolution. This approach leads to the following general expression for computing (27) as a convolution

$$w(n) = \Big(\mathcal{E}_a\{x(n)\} \circledast \mathcal{E}_b\{y(n)\}\Big)\mathcal{R}_K(n). \tag{28}$$

We can also compute $w(n)$ using transforms according to

$$w(n - n_0) = \mathcal{T}_c^{-1}\Big\{\mathcal{T}_a\{x(n)\} \times \mathcal{T}_b\{y(n)\}\Big\}. \tag{29}$$

For each type of symmetric convolution we must precisely define the meaning of the operators and symbols in (28) and (29). The convolution operator '$\circledast$' denotes a length-M circular (Ⓒ) or skew-circular (Ⓢ) convolution. $\mathcal{E}_a$ and $\mathcal{E}_b$ are the symmetric extension operators applied to $x(n)$ and $y(n)$, respectively. Definitions for these operators are given in Tables IV and V. Each operator converts a finite sequence into a length-M standard form base period of the specified SPS type. $\mathcal{T}_a$ and $\mathcal{T}_b$ are the corresponding DTT's we take of $x(n)$ and $y(n)$, respectively. $\mathcal{T}_c^{-1}$ is the appropriate inverse transform. These DTT's are as defined in the appendix.

The result of symmetric convolution is a sequence of representative samples of an SPS. The purpose of the length-K rectangular window $\mathcal{R}_K(n)$ is to extract those representative samples out of the base period. There are two possibilities for the n_0 in (29). When the symmetric convolution is a type that produces an SPS in standard form, $n_0 = 0$. For those types of symmetric convolution whose output is an SPS in standard form advanced by one sample, $\mathcal{R}_K(n)$ extracts those one-sample-advanced representative samples, and $n_0 = 1$. From the inverse transform we get an index range that has not been shifted; this output is equal to $w(n-1)$, the delayed result of the symmetric convolution.

Symmetric convolution is only possible between pairs of SPS's for which the convolution in (28) can be computed. The two SPS operands must have the same period and both

[2] This is a slight change from an earlier definition given in [6].

TABLE IV
SYMMETRIC EXTENSION OPERATORS AND IMPLIED PERIODICITIES, $M = 2N$

$\mathcal{E}$	$\hat{x}(n) = \mathcal{E}\{x(n)\}$	P or A
WSWS	$\hat{x}(n) = \begin{cases} x(n) & n = 0,1,\ldots,N \\ x(M-n) & n = N+1,\ldots,M-1 \end{cases}$	P
WAWA	$\hat{x}(n) = \begin{cases} 0 & n = 0 \\ x(n) & n = 1,2,\ldots,N-1 \\ 0 & n = N \\ -x(M-n) & n = N+1,\ldots,M-1 \end{cases}$	P
HSHS	$\hat{x}(n) = \begin{cases} x(n) & n = 0,1,\ldots,N-1 \\ x(M-1-n) & n = N,\ldots,M-1 \end{cases}$	P
HAHA	$\hat{x}(n) = \begin{cases} x(n) & n = 0,1,\ldots,N-1 \\ -x(M-1-n) & n = N,\ldots,M-1 \end{cases}$	P
WSWA	$\hat{x}(n) = \begin{cases} x(n) & n = 0,1,\ldots,N-1 \\ 0 & n = N \\ -x(M-n) & n = N+1,\ldots,M-1 \end{cases}$	A
WAWS	$\hat{x}(n) = \begin{cases} 0 & n = 0 \\ x(n) & n = 1,2,\ldots,N \\ x(M-n) & n = N+1,\ldots,M-1 \end{cases}$	A
HSHA	$\hat{x}(n) = \begin{cases} x(n) & n = 0,1,\ldots,N-1 \\ -x(M-1-n) & n = N,\ldots,M-1 \end{cases}$	A
HAHS	$\hat{x}(n) = \begin{cases} x(n) & n = 0,1,\ldots,N-1 \\ x(M-1-n) & n = N,\ldots,M-1 \end{cases}$	A

TABLE V
SYMMETRIC EXTENSION OPERATORS AND IMPLIED PERIODICITIES, $M = 2N-1$

$\mathcal{E}$	$\hat{x}(n) = \mathcal{E}\{x(n)\}$	P or A
WSHS	$\hat{x}(n) = \begin{cases} x(n) & n = 0,1,\ldots,N-1 \\ x(M-n) & n = N,\ldots,M-1 \end{cases}$	P
WAHA	$\hat{x}(n) = \begin{cases} 0 & n = 0 \\ x(n) & n = 1,2,\ldots,N-1 \\ -x(M-n) & n = N,\ldots,M-1 \end{cases}$	P
HSWS	$\hat{x}(n) = \begin{cases} x(n) & n = 0,1,\ldots,N-1 \\ x(M-1-n) & n = N,\ldots,M-1 \end{cases}$	P
HAWA	$\hat{x}(n) = \begin{cases} x(n) & n = 0,1,\ldots,N-2 \\ 0 & n = N-1 \\ -x(M-1-n) & n = N,\ldots,M-1 \end{cases}$	P
WSHA	$\hat{x}(n) = \begin{cases} x(n) & n = 0,1,\ldots,N-1 \\ -x(M-n) & n = N,\ldots,M-1 \end{cases}$	A
WAHS	$\hat{x}(n) = \begin{cases} 0 & n = 0 \\ x(n) & n = 1,2,\ldots,N-1 \\ x(M-n) & n = N,\ldots,M-1 \end{cases}$	A
HSWA	$\hat{x}(n) = \begin{cases} x(n) & n = 0,1,\ldots,N-2 \\ 0 & n = N-1 \\ -x(M-1-n) & n = N,\ldots,M-1 \end{cases}$	A
HAWS	$\hat{x}(n) = \begin{cases} x(n) & n = 0,1,\ldots,N-1 \\ x(M-1-n) & n = N,\ldots,M-1 \end{cases}$	A

TABLE VI
20 OF THE 40 TYPES OF SYMMETRIC CONVOLUTION, $M = 2N$

$<\varepsilon_a \overset{S}{C} \varepsilon_b>$	output SPS	⊛	input index ranges $x(n)$	input index ranges $y(n)$	output index range	n_0	$<\tau_a \overset{S}{C} \tau_b>$	$\mathcal{T}_c^{-1}$
<WSWS $\overset{S}{C}$ WSWS>	WSWS	Ⓒ	$0 \to N$	$0 \to N$	$0 \to N$	0	<C1e $\overset{S}{C}$ C1e>	$\mathcal{C}_{1e}^{-1}$
<WSWS $\overset{S}{C}$ WAWA>	WAWA	Ⓒ	$0 \to N$	$1 \to N-1$	$1 \to N-1$	0	<C1e $\overset{S}{C}$ S1e>	$\mathcal{S}_{1e}^{-1}$
<WAWA $\overset{S}{C}$ WAWA>	WSWS	Ⓒ	$1 \to N-1$	$1 \to N-1$	$0 \to N$	0	<S1e $\overset{S}{C}$ S1e>	$-\mathcal{C}_{1e}^{-1}$
<HSHS $\overset{S}{C}$ WSWS>	HSHS	Ⓒ	$0 \to N-1$	$0 \to N$	$0 \to N-1$	0	<C2e $\overset{S}{C}$ C1e>	$\mathcal{C}_{2e}^{-1}$
<HSHS $\overset{S}{C}$ WAWA>	HAHA	Ⓒ	$0 \to N-1$	$1 \to N-1$	$0 \to N-1$	0	<C2e $\overset{S}{C}$ S1e>	$\mathcal{S}_{2e}^{-1}$
<HAHA $\overset{S}{C}$ WSWS>	HAHA	Ⓒ	$0 \to N-1$	$0 \to N$	$0 \to N-1$	0	<S2e $\overset{S}{C}$ C1e>	$\mathcal{S}_{2e}^{-1}$
<HAHA $\overset{S}{C}$ WAWA>	HSHS	Ⓒ	$0 \to N-1$	$1 \to N-1$	$0 \to N-1$	0	<S2e $\overset{S}{C}$ S1e>	$-\mathcal{C}_{2e}^{-1}$
<HSHS $\overset{S}{C}$ HSHS>	WSWS	Ⓒ	$0 \to N-1$	$0 \to N-1$	$-1 \to N-1$	1	<C2e $\overset{S}{C}$ C2e>	$\mathcal{C}_{1e}^{-1}$
<HSHS $\overset{S}{C}$ HAHA>	WAWA	Ⓒ	$0 \to N-1$	$0 \to N-1$	$0 \to N-2$	1	<C2e $\overset{S}{C}$ S2e>	$\mathcal{S}_{1e}^{-1}$
<HAHA $\overset{S}{C}$ HAHA>	WSWS	Ⓒ	$0 \to N-1$	$0 \to N-1$	$-1 \to N-1$	1	<S2e $\overset{S}{C}$ S2e>	$-\mathcal{C}_{1e}^{-1}$
<WSWA $\overset{S}{C}$ WSWA>	WSWA	Ⓢ	$0 \to N-1$	$0 \to N-1$	$0 \to N-1$	0	<C3e $\overset{S}{C}$ C3e>	$\mathcal{C}_{3e}^{-1}$
<WSWA $\overset{S}{C}$ WAWS>	WAWS	Ⓢ	$0 \to N-1$	$1 \to N$	$1 \to N$	0	<C3e $\overset{S}{C}$ S3e>	$\mathcal{S}_{3e}^{-1}$
<WAWS $\overset{S}{C}$ WAWS>	WSWA	Ⓢ	$1 \to N$	$1 \to N$	$0 \to N-1$	0	<S3e $\overset{S}{C}$ S3e>	$-\mathcal{C}_{3e}^{-1}$
<HSHA $\overset{S}{C}$ WSWA>	HSHA	Ⓢ	$0 \to N-1$	$0 \to N-1$	$0 \to N-1$	0	<C4e $\overset{S}{C}$ C3e>	$\mathcal{C}_{4e}^{-1}$
<HSHA $\overset{S}{C}$ WAWS>	HAHS	Ⓢ	$0 \to N-1$	$1 \to N$	$0 \to N-1$	0	<C4e $\overset{S}{C}$ S3e>	$\mathcal{S}_{4e}^{-1}$
<HAHS $\overset{S}{C}$ WSWA>	HAHS	Ⓢ	$0 \to N-1$	$0 \to N-1$	$0 \to N-1$	0	<S4e $\overset{S}{C}$ C3e>	$\mathcal{S}_{4e}^{-1}$
<HAHS $\overset{S}{C}$ WAWS>	HSHA	Ⓢ	$0 \to N-1$	$1 \to N$	$0 \to N-1$	0	<S4e $\overset{S}{C}$ S3e>	$-\mathcal{C}_{4e}^{-1}$
<HSHA $\overset{S}{C}$ HSHA>	WSWA	Ⓢ	$0 \to N-1$	$0 \to N-1$	$-1 \to N-2$	1	<C4e $\overset{S}{C}$ C4e>	$\mathcal{C}_{3e}^{-1}$
<HSHA $\overset{S}{C}$ HAHS>	WAWS	Ⓢ	$0 \to N-1$	$0 \to N-1$	$0 \to N-1$	1	<C4e $\overset{S}{C}$ S4e>	$\mathcal{S}_{3e}^{-1}$
<HAHS $\overset{S}{C}$ HAHS>	WSWA	Ⓢ	$0 \to N-1$	$0 \to N-1$	$-1 \to N-2$	1	<S4e $\overset{S}{C}$ S4e>	$-\mathcal{C}_{3e}^{-1}$

of them must be periodic or both antiperiodic. Accordingly, we have grouped the SPS's in Table II and Fig. 2 into four classes, called *convolution classes*, of four SPS's each. We can convolve any pair of SPS's within a class but not between classes. The SPS result is also a member of that same class. The type of the SPS result can be ascertained from the symmetries of the operands or from the equivalent GDFT's and their convolution-multiplication properties. We discuss this further in Section VI.

There are 64 possible convolutions, 16 per class. Symmetric convolution is an ordered operation; we can reverse the operator if we swap the operands. For example, the symmetric convolution $x(n)$ <HSHS $\overset{S}{C}$ WAWA> $y(n)$ is the same as $y(n)$ < WAWA $\overset{S}{C}$ HSHS> $x(n)$. After removing such redundancies, we are left with 40 distinct types of symmetric convolution. Tables VI and VII list all 40 types of symmetric convolution along with the corresponding parameters for (28) and (29). The name for each type of symmetric convolution identifies the implied symmetric extensions applied to its operands. An alternative but equivalent name identifies the types of the DTT's to take. The tables also show the SPS type of the result, all index ranges, and the appropriate inverse DTT. The output index ranges listed are those that result from performing symmetric convolution according to (28). If (29) is used, then for those entries where $n_0 = 1$, the index ranges for the outputs from the inverse DTT's differ by one sample from those listed. Finally, it is important to notice that the lengths of the two sequences being convolved and of the resulting sequence may not all be the same.

C. Symmetric Convolution Using Discrete Trigonometric Transforms

Symmetric convolution is the convolution mode of the DTT's. We can efficiently compute symmetric convolution by taking an inverse DTT of the element-by-element product of the forward DTT's of the inputs. We must use the appropriate DST or DCT for each type of symmetric convolution. The DTT whose *inverse* corresponds to the desired implied symmetric extension of a symmetric convolution operand is the DTT whose *forward* version we take of that operand.

Table III is organized around the forward transforms. We can use that table to ascertain which DTT's can be multiplied together and which inverse transform to use. A property of SPS's that we use here is that the product of two SPS's with a common period is also an SPS with this same period if and only if they all have the same LPOS and the same RPOS. Referring to Table III this means that a necessary condition for (29) to hold is that the three forward transforms $\mathcal{T}_a$, $\mathcal{T}_b$, and $\mathcal{T}_c$ share the same LPOS and the same RPOS. Accordingly,

TABLE VII
20 OF THE 40 TYPES OF SYMMETRIC CONVOLUTION, $M = 2N - 1$

$<\mathcal{E}_a \circledS_c \mathcal{E}_b>$	output SPS	⊛	input index ranges $x(n)$	input index ranges $y(n)$	output index range	n_0	$<\mathcal{T}_a \circledS_c \mathcal{T}_b>$	$\mathcal{T}_c^{-1}$
$<\text{WSHS} \circledS_c \text{WSHS}>$	WSHS	Ⓒ	$0 \to N-1$	$0 \to N-1$	$0 \to N-1$	0	$<\text{C1e} \circledS_c \text{C1e}>$	$\mathcal{C}_{1e}^{-1}$
$<\text{WSHS} \circledS_c \text{WAHA}>$	WAHA	Ⓒ	$0 \to N-1$	$1 \to N-1$	$1 \to N-1$	0	$<\text{C1e} \circledS_c \text{S1e}>$	$\mathcal{S}_{1e}^{-1}$
$<\text{WAHA} \circledS_c \text{WAHA}>$	WSHS	Ⓒ	$1 \to N-1$	$1 \to N-1$	$0 \to N-1$	0	$<\text{S1e} \circledS_c \text{S1e}>$	$-\mathcal{C}_{1e}^{-1}$
$<\text{HSWS} \circledS_c \text{WSHS}>$	HSWS	Ⓒ	$0 \to N-1$	$0 \to N-1$	$0 \to N-1$	0	$<\text{C2e} \circledS_c \text{C1e}>$	$\mathcal{C}_{2e}^{-1}$
$<\text{HSWS} \circledS_c \text{WAHA}>$	HAWA	Ⓒ	$0 \to N-1$	$1 \to N-1$	$0 \to N-2$	0	$<\text{C2e} \circledS_c \text{S1e}>$	$\mathcal{S}_{2e}^{-1}$
$<\text{HAWA} \circledS_c \text{WSHS}>$	HAWA	Ⓒ	$0 \to N-2$	$0 \to N-1$	$0 \to N-2$	0	$<\text{S2e} \circledS_c \text{C1e}>$	$\mathcal{S}_{2e}^{-1}$
$<\text{HAWA} \circledS_c \text{WAHA}>$	HSWS	Ⓒ	$0 \to N-2$	$1 \to N-1$	$0 \to N-1$	0	$<\text{S2e} \circledS_c \text{S1e}>$	$-\mathcal{C}_{2e}^{-1}$
$<\text{HSWS} \circledS_c \text{HSWS}>$	WSHS	Ⓒ	$0 \to N-1$	$0 \to N-1$	$-1 \to N-2$	1	$<\text{C2e} \circledS_c \text{C2e}>$	$\mathcal{C}_{1e}^{-1}$
$<\text{HSWS} \circledS_c \text{HAWA}>$	WAHA	Ⓒ	$0 \to N-1$	$0 \to N-2$	$0 \to N-2$	1	$<\text{C2e} \circledS_c \text{S2e}>$	$\mathcal{S}_{1e}^{-1}$
$<\text{HAWA} \circledS_c \text{HAWA}>$	WSHS	Ⓒ	$0 \to N-2$	$0 \to N-2$	$-1 \to N-2$	1	$<\text{S2e} \circledS_c \text{S2e}>$	$-\mathcal{C}_{1e}^{-1}$
$<\text{WSHA} \circledS_c \text{WSHA}>$	WSHA	Ⓢ	$0 \to N-1$	$0 \to N-1$	$0 \to N-1$	0	$<\text{C3e} \circledS_c \text{C3e}>$	$\mathcal{C}_{3e}^{-1}$
$<\text{WSHA} \circledS_c \text{WAHS}>$	WAHS	Ⓢ	$0 \to N-1$	$1 \to N-1$	$1 \to N-1$	0	$<\text{C3e} \circledS_c \text{S3e}>$	$\mathcal{S}_{3e}^{-1}$
$<\text{WAHS} \circledS_c \text{WAHS}>$	WSHA	Ⓢ	$1 \to N-1$	$1 \to N-1$	$0 \to N-1$	0	$<\text{S3e} \circledS_c \text{S3e}>$	$-\mathcal{C}_{3e}^{-1}$
$<\text{HSWA} \circledS_c \text{WSHA}>$	HSWA	Ⓢ	$0 \to N-2$	$0 \to N-1$	$0 \to N-2$	0	$<\text{C4e} \circledS_c \text{C3e}>$	$\mathcal{C}_{4e}^{-1}$
$<\text{HSWA} \circledS_c \text{WAHS}>$	HAWS	Ⓢ	$0 \to N-2$	$1 \to N-1$	$0 \to N-1$	0	$<\text{C4e} \circledS_c \text{S3e}>$	$\mathcal{S}_{4e}^{-1}$
$<\text{HAWS} \circledS_c \text{WSHA}>$	HAWS	Ⓢ	$0 \to N-1$	$0 \to N-1$	$0 \to N-1$	0	$<\text{S4e} \circledS_c \text{C3e}>$	$\mathcal{S}_{4e}^{-1}$
$<\text{HAWS} \circledS_c \text{WAHS}>$	HSWA	Ⓢ	$0 \to N-1$	$1 \to N-1$	$0 \to N-2$	0	$<\text{S4e} \circledS_c \text{S3e}>$	$-\mathcal{C}_{4e}^{-1}$
$<\text{HSWA} \circledS_c \text{HSWA}>$	WSHA	Ⓢ	$0 \to N-2$	$0 \to N-2$	$-1 \to N-2$	1	$<\text{C4e} \circledS_c \text{C4e}>$	$\mathcal{C}_{3e}^{-1}$
$<\text{HSWA} \circledS_c \text{HAWS}>$	WAHS	Ⓢ	$0 \to N-2$	$0 \to N-1$	$0 \to N-2$	1	$<\text{C4e} \circledS_c \text{S4e}>$	$\mathcal{S}_{3e}^{-1}$
$<\text{HAWS} \circledS_c \text{HAWS}>$	WSHA	Ⓢ	$0 \to N-1$	$0 \to N-1$	$-1 \to N-2$	1	$<\text{S4e} \circledS_c \text{S4e}>$	$-\mathcal{C}_{3e}^{-1}$

we have grouped the entries in Table III by LPOS and RPOS to again form four convolution classes. We can multiply any pair of transforms within a class but not between classes. The appropriate inverse transform is that whose forward transform is a member of that same class. Note that we must be careful about the index ranges when using the DTT's to perform symmetric convolution. In particular, we must align the indices correctly and supply any missing zeros for the multiplication and for the inverse transforms, if needed.

Because of the equivalence between each of the DTT's and its corresponding special form of the GDFT, the convolution-multiplication properties of the DTT's must follow those of the GDFT's. The computation of each type of symmetric convolution using DTT's can be easily proven by making direct reference to the convolution-multiplication properties for the GDFT's we give in (21)–(26) and to the definition of symmetric convolution in terms of a circular or skew-circular convolution of symmetrically extended inputs.

As an example, if we multiply the output from $\mathcal{C}_{2e}$ by the output from another $\mathcal{C}_{2e}$, this is equivalent to multiplying the output from one $\mathcal{G}_{0,\frac{1}{2}}$ with that from another, where we are assuming that the input to each $\mathcal{G}_{0,\frac{1}{2}}$ is a symmetrically extended version of the input to each $\mathcal{C}_{2e}$. According to (23) the inverse GDFT is $\mathcal{G}_{0,0}^{-1}$. Therefore, the correct inverse DTT must be $\mathcal{C}_{1e}^{-1}$. As another example, if we multiply the output from $\mathcal{C}_{2e}$ by that from $\mathcal{S}_{1e}$ we use (22) to determine that in this case the correct inverse transform must be $\mathcal{S}_{2e}^{-1}$.

VI. DISCRETE FILTERING USING SYMMETRIC CONVOLUTION

Symmetric convolution provides a systematic way to convolve symmetric FIR filters with symmetrically extended data. The symmetric convolution of two finite sequences is equivalent to symmetrically extending one sequence on the left in one of four ways, extending the other sequence symmetrically at both ends in one of 16 ways, and then linearly convolving them to produce an appropriate finite output sequence. The different types of symmetric convolution imply different types of symmetric extension for those sequences.

How we choose to interpret symmetric convolution as a filtering operation depends on which of its two operands we consider to be filtering the other. We can view either of the operands to symmetric convolution as a *filter-right-half* that, after implicit left-sided symmetric extension, becomes one of the four types of linear phase FIR filters: WS, WA, HS, or HA, where the WS and WA filters are implemented as zero-phase and the HS and HA filters are implemented as 1/2-sample advance. In order for this interpretation of the one operand as a linear phase FIR filter to be valid, the length L of the effective filter must satisfy $L \leq M$, where M is the period of the SPS's being convolved. The other operand to symmetric

TABLE VIII
CONVOLUTION OF SYMMETRIC SEQUENCES

Symmetry before filtering	*Symmetry after filtering with:* WS	WA	HS	HA
$\alpha\beta$	$\alpha\beta$	$\alpha\beta'$	$\alpha'\beta$	$\alpha'\beta'$

convolution, which is implicitly extended symmetrically at both ends, is considered to be the data being filtered. The two-sided extension of the data provides smooth boundary values for filtering near its original endpoints. From the length of the data sequence we derive the parameter N and the period M for the symmetric convolution operation. There is also an implicit right-sided extension associated with the filter-right-half but its only purpose is for compatibility with the extensions for the data.

As stated earlier, we have grouped the plots of Fig. 2 and the entries in Table II into classes each containing four SPS's that are compatible for symmetric convolution. Each of those SPS's corresponds to one of the four types of linear phase FIR filters as determined by the LPOS of the SPS. There is exactly one occurrence of each type of filter in each class. Thus, we can always find in each class any one of the four linear phase FIR filters that we may need. In order to filter a sequence using symmetric convolution, we first choose an SPS representation for the data as appropriate for how we want to handle the boundary conditions. Within the convolution class for that SPS we then find the SPS that corresponds to the filter. The type of symmetric convolution we perform is the one that is appropriate for those two SPS types. The result of the convolution is also in that same class.

The SPS type of the result of a symmetric convolution is a consequence of the filter symmetry and the implied extensions for the data. Filtering an SPS with a symmetric filter produces another SPS, but each point of symmetry changes as shown in Table VIII. In the table, α stands for either 'W' (whole) or 'H' (half), and α' is the opposite; β stands for either 'S' (symmetric) or 'A' (antisymmetric) and β' is the opposite. As examples, filtering an HSHS-SPS with an HS filter produces a WSWS-SPS, and filtering a WSHA-SPS with a WA filter yields a WAHS-SPS. We can use the table to determine for each type of symmetric convolution what would be the output SPS type as we have listed in Tables VI and VII.

We give some filtering examples in Figs. 3 and 4, and discuss them here. Our first example uses an even-length symmetric filter. Let $h(n)$ be an L-tap HS filter spanning the interval $n = -L/2, \cdots, L/2-1$ through which we pass the data sequence $x(n)$, $n = 0, \cdots, N-1$. Suppose we want HS extensions at both ends of $x(n)$. The output from this filter can be computed from

$$w(n) = \sum_{k=-L/2}^{L/2-1} h(k)\tilde{x}(n-k) \qquad n = -1, 0, \cdots, N-1 \quad (30)$$

where $\tilde{x}(n)$ is $x(n)$ after being HS extended at both ends to the extent needed for performing the summation. The output sequence $w(n)$ is one sample longer than the input $x(n)$

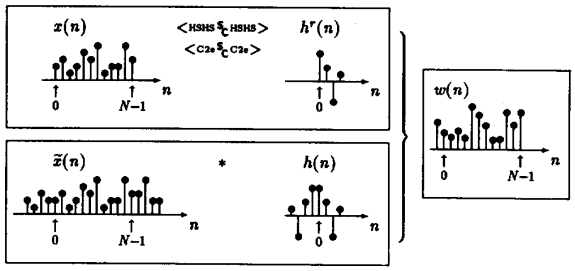


Fig. 3. Example showing how the same $w(n)$ results from a symmetric convolution as from a linear convolution of symmetrically extended inputs. For this type of symmetric convolution, $\tilde{x}(n)$ and $h(n)$ are subsequences of HSHS SPS's, and the samples of $w(n)$ are the one-sample-advanced representative samples of a WSWS-SPS.

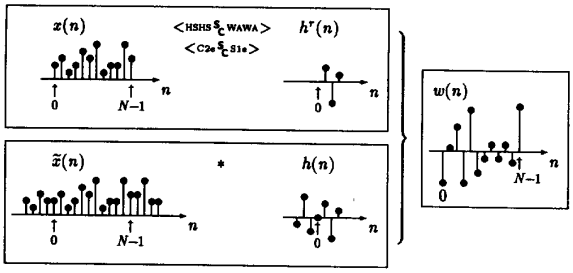


Fig. 4. Example showing how the same $w(n)$ results from a symmetric convolution as from a linear convolution of symmetrically extended inputs. For this type of symmetric convolution, $\tilde{x}(n)$ and $h(n)$ are subsequences of HSHS and WAWA SPS's, respectively, and the samples of $w(n)$ are the representative samples of a HAHA-SPS.

because of the HS extensions to $x(n)$ and the 1/2-sample advance of the filter $h(n)$.

To use symmetric convolution, we first define the filter-right-half by

$$h^r(n) = \begin{cases} h(n) & n = 0, 1, \cdots, L/2-1 \\ 0 & n = L/2, \cdots, N-1 \end{cases} \quad (31)$$

where $L/2 \leq N$. The appropriate type of symmetric convolution is

$$\begin{aligned} w(n) &= x(n) \,\langle \text{HSHS} \overset{\mathbf{S}}{\mathbf{C}} \text{HSHS} \rangle\, h^r(n) \\ &= x(n) \langle \text{C2e} \overset{\mathbf{S}}{\mathbf{C}} \text{C2e} \rangle h^r(n). \end{aligned} \quad (32)$$

We can perform the above operation by using circular convolution and (28) or by using DTT's and (29). With the correct choice of transforms, the latter equation becomes

$$w(n-1) = \mathbf{C}_{1e}^{-1}\Big\{\mathcal{C}_{2e}\{x(n)\} \times \mathcal{C}_{2e}\{h^r(n)\}\Big\}. \quad (33)$$

When using the DTT's we must pay special attention to the indices of the transforms in order for the above equation to be valid. To be specific, each $\mathcal{C}_{2e}$ is N-point with input and output index ranges $0, 1, \cdots, N-1$. We multiply their outputs over that same index range. We put a zero at index N and take an $(N+1)$-point $\mathcal{C}_{1e}^{-1}$ with input and output index ranges $0, 1, \cdots, N$. The extra value needed at index N for the input to the $\mathcal{C}_{1e}^{-1}$ is set to zero because we know that that is what the $\mathcal{C}_{2e}$ would have generated there.

In our second example we let L be odd and $h(n)$ be an L-tap WA filter spanning the interval $n = -(L-1)/2, \cdots, (L-1)/2$. We use the same $x(n)$ as in the previous example. The convolution we want is given by

$$w(n) = \sum_{k=-(L-1)/2}^{(L-1)/2} h(k)\tilde{x}(n-k) \qquad n = 0, 1, \cdots, N-1 \quad (34)$$

where $\tilde{x}(n)$ is also as before. Here the filter is zero-phase so the output $w(n)$ is the same length as the input $x(n)$.

Because $h(n)$ is WA, we omit the zero at $h(0)$ and make the assignment

$$h^r(n) = \begin{cases} h(n) & n = 1, 2, \cdots, (L-1)/2 \\ 0 & n = (L+1)/2, \cdots, N-1 \end{cases} \quad (35)$$

where $(L-1)/2 \leq (N-1)$. The symmetric convolution is

$$\begin{aligned} w(n) &= x(n) \,{<}\text{HSHS}\overset{\mathbf{S}}{\mathbf{C}}\text{WAWA}{>}\, h^r(n) \\ &= x(n){<}\text{C2e}\overset{\mathbf{S}}{\mathbf{C}}\text{S1e}{>}h^r(n). \end{aligned} \quad (36)$$

We can perform this convolution efficiently using DTT's according to

$$w(n) = \mathcal{S}_{2e}^{-1}\Big\{\mathcal{C}_{2e}\{x(n)\} \times \mathcal{S}_{1e}\{h^r(n)\}\Big\}. \quad (37)$$

In this example we must be particularly careful about the index ranges. The input and output index ranges for the N-point $\mathcal{C}_{2e}$ are both $0, 1, \cdots, N-1$, and for the $(N-1)$-point $\mathcal{S}_{1e}$ they are both $1, 2, \cdots, N-1$. The N-point $\mathcal{S}_{2e}^{-1}$ has an input index range of $1, 2, \cdots, N$. We find those samples by performing the multiplication over the index range $1, 2, \cdots, N-1$ and inserting what we know to be zero at N. The output index range from the $\mathcal{S}_{2e}^{-1}$ is the correct $0, 1, \cdots, N-1$.

VII. Using Symmetric Convolution to Perform Linear Convolution

A symmetric convolution of two finite sequences gives a result that is the same as a linear convolution of one with a left-sided extended version of the other followed by "folded aliasing" at both ends of the output. Symmetric convolution thus provides a way to filter a finite sequence and produce a same-size or near-same-size result. This folded aliasing may be preferable in some applications to the wrap-around aliasing of a circular convolution/DFT approach. Note that it is the parameters N and M that remain unchanged by symmetric convolution; input and output sequence lengths may be the same or may differ by one or two samples.

It is also possible to use symmetric convolution and DTT's to perform a true linear convolution. We merely need to pad the input data with a sufficient number of zeros *at both ends* prior to symmetric convolution to avoid the effects of the folded aliasing. The amount of zero-padding required at each end is a function of the symmetry of the filter and the implied symmetric extension done to that end of the data. If the effective FIR filter has L taps, then the minimum number of zeros we must add to each end of the data is as given in Table IX.

TABLE IX
Zero-Padding Required to Compute a Linear Convolution

Symmetry of filter	*Amount of zero-padding needed before extending as:* WS	WA	HS	HA
WS	$(L+1)/2$*	$(L-1)/2$	$(L-1)/2$	$(L-1)/2$
WA	$(L+1)/2$	$(L-1)/2$*	$(L-1)/2$	$(L-1)/2$
HS	$L/2$	$L/2-1$	$L/2$*	$L/2$
HA	$L/2$	$L/2-1$	$L/2$	$L/2$*

Any one of 16 types of symmetric convolution can be used to perform the same linear convolution because there are 16 ways to represent the data as an SPS. For each there is a corresponding compatible SPS representation of the filter. The type of symmetric convolution that we choose determines which specific transforms to use. The types also may differ in the amount of zero-padding necessary, and whether the output has the correct length or is one or two samples longer. For those types of symmetric convolution that produce an output whose implicit symmetry type is WS at either endpoint and the zero-padding is done as in Table IX, there will be an extraneous sample with the value zero at each WS point in the output. In the table, we have marked with an asterisk where this will occur.

We give an example in Fig. 5 that demonstrates how, after zero-padding of one of its operands, a symmetric convolution produces the same result as a linear convolution. The filter $h(n)$ is WA and has seven taps. We have chosen to use a symmetric convolution that implies a HSHS-SPS representation for the data and, consequently, a WAWA-SPS representation for the filter. Table IX tells us to first pad with three zeros at each end of $x(n)$; what results is the sequence $\hat{x}(n)$ in the figure. Notice that the sequences $x(n)$ and $h^r(n)$ are the same as in Fig. 4, but the output $w(n)$ is that of a true symmetric convolution whereas the output $y(n)$ is that of a true linear convolution. Notice also that the sequence $w(n)$ can be obtained from $y(n)$ by folding in the samples at the ends and aliasing. Because $w(n)$ is implicitly HAHA, the relationship is

$$w(n) = \begin{cases} y(n+3) - y(2-n) & n = 0, 1, 2 \\ y(n+3) & n = 3, 4, 5, 6, 7, 8 \\ y(n+3) - y(26-n) & n = 9, 10, 11 \end{cases}. \quad (38)$$

Because it is possible to do linear convolution by symmetric convolution it follows that both of the techniques of *overlap-add* and *overlap-save* can be used with the DTT's. We must pad and overlap at both ends of the data sequence, where the amount of padding and overlapping is as given in Table IX.

VIII. A New Look at Earlier Work

The first attempt to develop a convolution-multiplication property for the DCT was published in 1976 by Chen and Fralick [8]. They analyzed the effect of taking an inverse type-2 DCT of the product of the forward type-2 DCT of the data sequence $x(n)$ with the forward type-2 DCT of the filter $h(n)$. The resulting sequence was found equal to the circular convolution of three terms. Two of those terms represented

double-length versions of $x(n)$ and $h(n)$, respectively, created by reflecting signal values on the right. In our terminology, these were HS extensions. The third term was cumbersome and is not repeated here.

The presence of the third term in the above convolution is a consequence of using an invalid mix of DCT types. If we wish to multiply two type-2 DCT's, then we must use a type-1 DCT for the inverse transformation. A type-2 inverse DCT could be used only if we were multiplying a type-2 DCT with a type-1 DCT. This latter case is what is actually happening in the equations derived by Chen and Fralick. Their third term is a correction term needed to replace the type-2 DCT of $h(n)$ with a type-1 DCT of the related filter $g(n)$, where the symmetrically extended $g(n)$ is equal to the circular convolution of the symmetrically extended $h(n)$ with that third term. In terms of symmetric convolution, the Chen and Fralick equation should have been $x(n)$ <HSHS$\overset{S}{C}$WAWA> $g(n)$, or equivalently, $x(n)$<C2e$\overset{S}{C}$C1e>$g(n)$. The appropriate inverse transform would be $\mathcal{C}_{2e}^{-1}$.

A new convolution-multiplication property was presented by Chitprasert and Rao that applies only when the filter frequency response is real and even [9]. By restricting the filter to be zero-phase, odd-length, and symmetric, they found that they could perform the circular convolution of the symmetrically extended $x(n)$ with the already symmetric $h(n)$ by taking an inverse type-2 DCT of the product of the forward type-2 DCT of $x(n)$ with the DFT of $h(n)$. Harada also reported this mix of DCT and DFT [10]. In addition, he derived another mix of DCT and DFT as well as two forms combining the DST and DFT.

In the properties of Chitprasert and Rao and of Harada we can remove the need to use the DFT by noting that these DFT's are all equivalent to a DST or DCT. For example, if $h(n)$ is odd-length and symmetric, then half of the DFT of $h(n)$ is the same as the type-1 DCT of half of $h(n)$. A similar relation holds for the DST. Thus, these earlier results are merely special cases of symmetric convolution and the convolution-multiplication properties we present in this paper.

Finally, we note that there has appeared in the literature some discussion of the link between symmetric-periodic sequences and the DCT's [25], [35], [36]. However, these presentations have been limited to only a few cases and not developed further. To our knowledge, this paper is the first to fully exploit the concept of symmetric-periodic sequences and their link to the DTT's and use it to derive all possible convolution-multiplication properties.

IX. Applications

In this section, we briefly mention some applications of symmetric convolution and its implementation using DTT's. Further details can be found in the references cited. We consider the area of image processing. Symmetric convolution implements linear phase FIR filters and such filtering occurs often in image processing. The eye is particularly sensitive to phase distortion in images. When filters have linear phase, phase distortion is reduced to only a spatial shift. The methods of overlap-add and overlap-save are now possible for the DTT's as well as for the DFT. We can use either transform since the results are the same and the computational complexities are comparable [7]. We choose between them based on such criteria as which transform we want or need to use, whether it makes a difference if the arithmetic is real or complex, and how we want to handle the image boundaries.

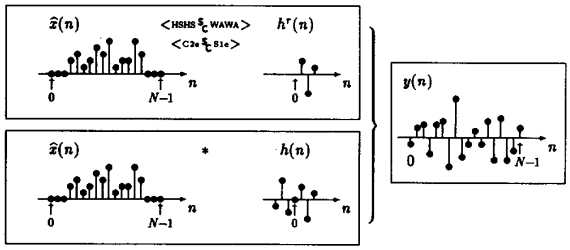


Fig. 5. Example showing how the same $y(n)$ results from a symmetric convolution as from a conventional linear convolution when the data have been zero-padded at both ends.

When filtering images, there are image boundary conditions that need to be specified. If we assume that the samples outside the boundaries are zero, then the output image from FIR filtering will be larger than the input image. If we require both images to be the same size, as is often the case, then we must delete some samples. We can instead use symmetric convolution to produce an output image that is the same size as the input image without having to delete samples. The circular convolution of the DFT also gives a same-size result, but the implied periodic extensions may produce objectionable artifacts [7], [37]. The smooth transitions that result from the implied symmetric extensions of the DTT's should not degrade the quality of the filtered output. If we use the DTT's and overlap-add or overlap-save, we can do conventional overlapping for all but the boundary blocks, then use the DTT's without overlap at the boundaries. There is also a way to do all filtering without overlapping any of the blocks [7], [37]. Such an approach using DTT's would make it possible to use the same transform operation for filtering and block transform coding of an image.

Another application for symmetric convolution is the implementation of the filters of a multirate filter bank. Symmetric convolution provides an elegant way to avoid the problem of subband signal size expansion in general multirate filter banks for images [7], [38]. The *symmetric extension method* [36] is just one particular type of symmetric convolution. By formulating the solution in terms of symmetric convolution, all types of symmetric extension are naturally included and there are no cases that would require special handling. In addition, we could use DTT's to implement these banks without incurring the overhead of explicitly performing the symmetric extensions prior to filtering.

X. Conclusion

We have introduced and explained the details of a special type of convolution called symmetric convolution. We have also presented new convolution-multiplication properties for the entire family of discrete sine and cosine transforms.

We have shown how filtering can be accomplished using symmetric convolution and the DTT's. With proper zero-padding of the data, we can use symmetric convolution to compute a linear convolution. Such capabilities should furnish new uses for emerging DST and DCT chips. Symmetric convolution and the DTT's are now an alternative to the DFT for efficient filtering.

APPENDIX
THE 16 DISCRETE TRIGNOMETRIC TRANSFORMS

The eight versions of the discrete cosine transform and the eight versions of the discrete sine transform in the family of discrete trigonometric transforms sharing a common value of N are given below in matrix notation. These matrices are not orthogonal. The expression for each forward transform defines the entry at row m and column n of the matrix. The meaning of the symbols in each matrix identifier should be self-explanatory. The ranges shown for m and n are the exclusive index ranges. The weighting functions are defined by

$$k_p = \begin{cases} 1/2 & p = 0 \text{ or } N \\ 1 & p = 1, 2, \cdots, N-1 \end{cases}$$

$$l_p = \begin{cases} 1 & p = 0, 1, \cdots, N-2 \\ 1/2 & p = N-1. \end{cases}$$

$$[\mathcal{C}_{1e}]_{mn} = 2k_n \cos\left(\frac{\pi mn}{N}\right) \qquad m, n = 0, 1, \cdots, N \tag{A.1}$$

$$[\mathcal{C}_{2e}]_{mn} = 2\cos\left(\frac{\pi m\left(n+\frac{1}{2}\right)}{N}\right) \qquad m, n = 0, 1, \cdots, N-1 \tag{A.2}$$

$$[\mathcal{C}_{3e}]_{mn} = 2k_n \cos\left(\frac{\pi\left(m+\frac{1}{2}\right)n}{N}\right) \qquad m, n = 0, 1, \cdots, N-1 \tag{A.3}$$

$$[\mathcal{C}_{4e}]_{mn} = 2\cos\left(\frac{\pi\left(m+\frac{1}{2}\right)\left(n+\frac{1}{2}\right)}{N}\right) \qquad m, n = 0, 1, \cdots, N-1 \tag{A.4}$$

$$[\mathcal{C}_{1o}]_{mn} = 2k_n \cos\left(\frac{2\pi mn}{2N-1}\right) \qquad m, n = 0, 1, \cdots, N-1 \tag{A.5}$$

$$[\mathcal{C}_{2o}]_{mn} = 2l_n \cos\left(\frac{2\pi m\left(n+\frac{1}{2}\right)}{2N-1}\right) \qquad m, n = 0, 1, \cdots, N-1 \tag{A.6}$$

$$[\mathcal{C}_{3o}]_{mn} = 2k_n \cos\left(\frac{2\pi\left(m+\frac{1}{2}\right)n}{2N-1}\right) \qquad m, n = 0, 1, \cdots, N-1 \tag{A.7}$$

$$[\mathcal{C}_{4o}]_{mn} = 2\cos\left(\frac{2\pi\left(m+\frac{1}{2}\right)\left(n+\frac{1}{2}\right)}{2N-1}\right) \qquad m, n = 0, 1, \cdots, N-2 \tag{A.8}$$

$$[\mathcal{S}_{1e}]_{mn} = 2\sin\left(\frac{\pi mn}{N}\right) \qquad m, n = 1, 2, \cdots, N-1 \tag{A.9}$$

$$[\mathcal{S}_{2e}]_{mn} = 2\sin\left(\frac{\pi m\left(n+\frac{1}{2}\right)}{N}\right) \qquad \begin{matrix} m = 1, 2, \cdots, N \\ n = 0, 1, \cdots, N-1 \end{matrix} \tag{A.10}$$

$$[\mathcal{S}_{3e}]_{mn} = 2k_n \sin\left(\frac{\pi\left(m+\frac{1}{2}\right)n}{N}\right) \qquad \begin{matrix} m = 0, 1, \cdots, N-1 \\ n = 1, 2, \cdots, N \end{matrix} \tag{A.11}$$

$$[\mathcal{S}_{4e}]_{mn} = 2\sin\left(\frac{\pi\left(m+\frac{1}{2}\right)\left(n+\frac{1}{2}\right)}{N}\right) \qquad m, n = 0, 1, \cdots, N-1 \tag{A.12}$$

$$[\mathcal{S}_{1o}]_{mn} = 2\sin\left(\frac{2\pi mn}{2N-1}\right) \qquad m, n = 1, 2, \cdots, N-1 \tag{A.13}$$

$$[\mathcal{S}_{2o}]_{mn} = 2\sin\left(\frac{2\pi m\left(n+\frac{1}{2}\right)}{2N-1}\right) \qquad \begin{matrix} m = 1, 2, \cdots, N-1 \\ n = 0, 1, \cdots, N-2 \end{matrix} \tag{A.14}$$

$$[\mathcal{S}_{3o}]_{mn} = 2\sin\left(\frac{2\pi\left(m+\frac{1}{2}\right)n}{2N-1}\right) \qquad \begin{matrix} m = 0, 1, \cdots, N-2 \\ n = 1, 2, \cdots, N-1 \end{matrix} \tag{A.15}$$

$$[\mathcal{S}_{4o}]_{mn} = 2l_n \sin\left(\frac{2\pi\left(m+\frac{1}{2}\right)\left(n+\frac{1}{2}\right)}{2N-1}\right) \qquad m, n = 0, 1, \cdots, N-1 \tag{A.16}$$

The relationship between each inverse matrix and its own or another forward matrix is given below. The designation for *even* or *odd* has been omitted from the following equations because the same relation holds for both cases, i.e., each expression is equivalent to two expressions, one for the even transforms and one for the odd transforms. However, the value of M differs. For the even transforms, $M = 2N$; for the odd transforms, $M = 2N - 1$.

$$[\mathcal{C}_1]^{-1} = \frac{1}{M}[\mathcal{C}_1] \tag{A.17}$$

$$[\mathcal{C}_2]^{-1} = \frac{1}{M}[\mathcal{C}_3] \tag{A.18}$$

$$[\mathcal{C}_3]^{-1} = \frac{1}{M}[\mathcal{C}_2] \tag{A.19}$$

$$[\mathcal{C}_4]^{-1} = \frac{1}{M}[\mathcal{C}_4] \tag{A.20}$$

$$[\mathcal{S}_1]^{-1} = \frac{1}{M}[\mathcal{S}_1] \tag{A.21}$$

$$[\mathcal{S}_2]^{-1} = \frac{1}{M}[\mathcal{S}_3] \tag{A.22}$$

$$[\mathcal{S}_3]^{-1} = \frac{1}{M}[\mathcal{S}_2] \tag{A.23}$$

$$[\mathcal{S}_4]^{-1} = \frac{1}{M}[\mathcal{S}_4] \tag{A.24}$$

ACKNOWLEDGMENT

The author wishes to express his deep gratitude for the support and advice he has received from Prof. R. M. Mersereau of the Georgia Institute of Technology while pursuing this research. Special thanks go to Prof. H. S. Malvar of the Universidade de Brasília and Prof. P. P. Vaidyanathan of Caltech for taking the time to read and provide helpful comments on this paper. Thanks also to R. Jonsson, D. Drake, and D. Williams for their proofreading efforts and to the anonymous reviewer whose critique helped the author improve the quality of the presentation.

REFERENCES

[1] G. K. Wallace, "The JPEG still picture compression standard," *Commun. Assoc. Comput. Mach.*, vol. 34, pp. 30–44, Apr. 1991.

[2] M. Liou, "Overview of the px64 kbit/s video coding standard," *Commun. Assoc. Comput. Mach.*, vol. 34, pp. 59–63, Apr. 1991.

[3] D. Le Gall, "MPEG: A video compression standard for multimedia applications," *Commun. Assoc. Comput. Mach.*, vol. 34, pp. 46–58, Apr. 1991.

[4] A. K. Jain, *Fundamentals of Digital Image Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1989.

[5] H. S. Malvar, *Signal Processing with Lapped Transforms*. Norwood, MA: Artech, 1992.

[6] S. A. Martucci, "Convolution-multiplication properties for the entire family of discrete sine and cosine transforms," in *Proc. Twenty-sixth Annu. Conf. Inform. Sci. Syst.* (Princeton, NJ), Mar. 1992, pp. 399–404.

[7] S. A. Martucci, "Symmetric convolution and the discrete sine and cosine transforms: Principles and applications," Ph.D. dissertation, Georgia Institute of Technology, Atlanta, 1993.

[8] W. H. Chen and S. C. Fralick, "Image enhancement using cosine transform filtering," in *Proc. Symp. Current Mathematical Problems Image Sci.* (Monterey, CA), Nov. 1976, pp. 186–192.

[9] B. Chitprasert and K. R. Rao, "Discrete cosine transform filtering," *Signal Processing*, vol. 19, pp. 233–245, Mar. 1990.

[10] H. Harada, "On the convolution properties of DCT's and DST's," in *1990 Int. Symp. Inform. Theory, Applicat.* (Hawaii), Nov. 1990, pp. 591–594.

[11] G. Bongiovanni, P. Corsini, and G. Frosini, "One-dimensional and two-dimensional generalized discrete Fourier transforms," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-24, pp. 97–99, Feb. 1976.

[12] J. L. Vernet, "Real signals fast Fourier transform: Storage capacity and step number reduction by means of an odd discrete Fourier transform," *Proc. IEEE*, vol. 59, pp. 1531–1532, Oct. 1971.

[13] R. O. Rowlands, "The odd discrete Fourier transform," in *Proc. 1976 IEEE Int. Conf. Acoust., Speech, Signal Processing* (Philadelphia, PA), Apr. 1976, pp. 130–133.

[14] G. Bonnerot and M. Bellanger, "Odd-time odd-frequency discrete Fourier transform for symmetric real-valued series," *Proc. IEEE*, pp. 392–393, Mar. 1976.

[15] Z. Wang, "Fast algorithms for the discrete W transform and for the discrete Fourier transform," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-32, pp. 803–816, Aug. 1984.

[16] Z. Wang and B. R. Hunt, "The discrete W transform,"*Appl. Math., Computation*, vol. 16, pp. 19–48, Jan. 1985.

[17] K. R. Rao and P. Yip, *Discrete Cosine Transform: Algorithms, Advantages, Applications*. San Diego, CA: Academic, 1990.

[18] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transform," *IEEE Trans. Comput.*, vol. C-23, pp. 90–93, Jan. 1974.

[19] 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, Mar. 1978.

[20] A. K. Jain, "A sinusoidal family of unitary transforms," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-1, pp. 356–365, Oct. 1979.

[21] H. Kitajima, "A symmetric cosine transform," *IEEE Trans. Comput.*, vol. C-29, pp. 317–323, Apr. 1980.

[22] Z. Wang and B. R. Hunt, "The discrete cosine transform—a new version," in *Proc. 1983 IEEE Int. Conf. Acoust., Speech, Signal Processing* (Boston, MA), Apr. 1983, pp. 1256–1259.

[23] S. Matsumura, "Discrete cosine transforms—theory and LSI implementation," Master's thesis, Linköping University, Sweden, 1985.

[24] M. J. Narasimha and A. M. Peterson, "On the computation of the discrete cosine transform," *IEEE Trans. Commun.*, vol. COM-26, pp. 934–936, June 1978.

[25] J. Makhoul, "A fast cosine transform in one and two dimensions," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-28, pp. 27–34, Feb. 1980.

[26] M. Vetterli and H. J. Nussbaumer, "Simple FFT and DCT algorithms with reduced number of operations," *Signal Processing*, vol. 6, pp. 267–278, Aug. 1984.

[27] B. G. Lee, "A new algorithm to compute the discrete cosine transform," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-32, pp. 1243–1245, Dec. 1984.

[28] Z. Wang, "On computing the discrete Fourier and cosine transforms," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-33, pp. 1341–1344, Oct. 1985.

[29] H. S. Hou, "A fast recursive algorithm for computing the discrete cosine transform," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-35, pp. 1455–1461, Oct. 1987.

[30] S. C. Chan and K. L. Ho, "On computing the discrete cosine and sine transforms from the discrete Fourier transform," in *Proc. Int. Symp. Comput. Architecture, Digital Signal Processing* (Hong Kong), Oct. 1989, pp. 363–366.

[31] Z. Wang, "Fast discrete sine transform algorithms," *Signal Processing*, vol. 19, pp. 91–102, Feb. 1990.

[32] S. C. Chan and K. L. Ho, "Direct methods for computing discrete sinusoidal transforms," in *Inst. Elec. Eng. Proc.*, vol. 137, pt. F, Dec. 1990, pp. 433–442.

[33] S. C. Chan and K. L. Ho, "Fast algorithms for computing the discrete cosine transform," *IEEE Trans. Circuits, Syst.--II: Analog, Digital Signal Processing*, vol. 39, pp. 185–190, Mar. 1992.

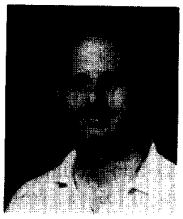
[34] S. A. Martucci, "Signal extension and noncausal filtering for subband coding of images," in *Proc. SPIE Vol. 1605 Visual Commun. Image Processing '91: Visual Commun.* (Boston, MA), Nov. 1991, pp. 137–148.

[35] J. C. Darragh, "Subband and transform coding of images," Ph.D. dissertation, Univ. California, Los Angeles, 1989.

[36] S. L. Eddins, "Subband analysis-synthesis and edge modeling methods for image coding," Ph.D. dissertation, Georgia Institute of Technology, Atlanta, 1990.

[37] S. A. Martucci and R. M. Mersereau, "New approaches to block filtering of images using symmetric convolution and the DST or DCT," in *Proc. 1993 IEEE Int. Symp. Circuits, Syst.* (Chicago, IL), May 1993, pp. 259–262.

[38] S. A. Martucci and R. M. Mersereau, "The symmetric convolution approach to the nonexpansive implementation of FIR filter banks for images," in *Proc. 1993 IEEE Int. Conf. Acoust., Speech, Signal Processing* (Minneapolis, MN), Apr. 1993, pp. V.65–V.68.



Stephen A. Martucci (S'86–M'93) was born in Mineola, NY, in 1960. He received the B.E.E. degree, with highest honor, in 1982, the M.S.E.E. degree in 1987, and the Ph.D. degree in electrical engineering in 1993, all from the Georgia Institute of Technology, Atlanta, GA. He won a DAAD (German Academic Exchange Service) scholarship and spent the period from October 1988 to December 1989 studying under Prof. H. G. Musmann at the *Institut für Theoretische Nachrichtentechnik und Informationsverarbeitung* of the *Technische Universität Hannover* in Germany.

His industrial experience includes three years of computer design and system software development in Massachusetts. He is currently a Member of Technical Staff at the David Sarnoff Research Center, Princeton, NJ. His research interests include discrete transforms, multirate filter banks, image processing, digital video, and computer animation.

Dr. Martucci is a member of ACM, Phi Kappa Phi, Eta Kappa Nu, and Tau Beta Pi. He has been listed in *International Youth in Achievement* and *Who's Who Among Students in American Universities and Colleges*.