

Sparse Matrix Based Low-Complexity, Recursive, and Radix-2 Algorithms for Discrete Sine Transforms

SIRANI M. PERERA AND LEVI E. LINGSCH

1Department of Mathematics, Embry-Riddle Aeronautical University, Daytona Beach, FL 32114, USA
2Department of Aerospace Engineering, Embry-Riddle Aeronautical University, Daytona Beach, FL 32114, USA

Corresponding author: Sirani M. Perera (pereras2@erau.edu)

This work was supported by the Faculty Innovative Research in Science and Technology Grant (grant number 13238) in Embry-Riddle Aeronautical University, USA.

ABSTRACT This paper presents factorizations of each discrete sine transform (DST) matrix of types I, II, III, and IV into a product of sparse, diagonal, bidiagonal, and scaled orthogonal matrices. Based on the proposed matrix factorization formulas, reduced multiplication complexity, recursive, and radix-2 DST I-IV algorithms are presented. We will present the lowest multiplication complexity DST-IV algorithm in the literature. The paper fills a gap in the self-recursive, exact, and radix-2 DST I-III algorithms executed via diagonal, bidiagonal, scaled orthogonal, and simple matrix factors for any input \( n = 2^t \) \((t \geq 1)\). The paper establishes a novel relationship between DST-II and DST-IV matrices using diagonal and bidiagonal matrices. Similarly, a novel relationship between DST-I and DST-III matrices is proposed using sparse and diagonal matrices. These interweaving relationships among DST matrices enable us to bridge the existing factorizations of the DST matrices with the proposed factorization formulas. We present signal flow graphs to provide a layout for realizing the proposed algorithms in DST-based integrated circuit designs. Additionally, we describe an implementation of algorithms based on the proposed DST-II and DST-III factorizations within a double random phase encoding (DRPE) image encryption scheme.

INDEX TERMS Discrete sine transforms, self/completely recursive and radix-2 algorithms, complexity and performance of algorithms, sparse and orthogonal matrices, signal flow graphs.

I. INTRODUCTION

Discrete sine transforms are real-valued transform matrices and a subclass of the discrete Fourier transform (DFT) matrix. The DFT matrix can be described as a Vandermonde matrix having nodes as the primitive \( n^{th} \) roots of unity, where \( n \) is a positive integer. The fast Fourier transform (FFT) is an algorithm that can be used to compute the DFT and its inverse efficiently.

For a given vector \( x = [x_0, x_1, \ldots, x_{n-1}]^T \in \mathbb{R}^n \), its DST can be expressed as \( Sx \), where \( S \) is the DST matrix. The DST matrix has four main variants, i.e., I-IV as defined below

\[
S_{n-1}^I = \sqrt{\frac{2}{n}} \left[ \sin \left( \frac{(j+1)(k+1)\pi}{n} \right) \right]_{j,k=0}^{n-1} ,
\]

\[
S_{n}^II = \sqrt{\frac{2}{n}} \left[ \epsilon_n(j+1) \sin \left( \frac{(j+1)(2k+1)\pi}{2n} \right) \right]_{j,k=0}^{n-1} ,
\]

where \( \epsilon_n(0) = \epsilon_n(n) = \frac{1}{\sqrt{2}} ; \epsilon_n(j) = 1 \) for \( j \in \{1, 2, \cdots, n-1\} \) and \( n \geq 2 \) is an even integer. Here, we use the superscript to denote the type of a given DST matrix and the subscript to denote the order of a DST matrix. Among the DST-I-IV matrices, \( S_{n-1}^I \) and \( S_{n}^IV \) were introduced in [1], [2], and \( S_{n}^II \) and its inverse \( S_{n}^{III} \) were introduced in [3] to digital signal processing. The complete set of even discrete cosine transform (DCT) and DST variants was originally presented in [4]. DST-II is a complementary or alternative transform to DCT-II, which is used in transform coding. Like the DFT and DCT, these DST matrices hold linearity, convolution-multiplication, orthogonality, and shift properties. Moreover, the DSTs of types II and III are related via \( S_{n}^{III} = [S_{n}^{II}]^T \). In this paper, we will establish relationships
between $S_n^H$ and $S_n^{IV}$ as well as $S_{n-1}^I$ and $S_n^{III}$. The matrix factorization for DST-I in [5] used the results from [6] to decompose DST-I into the DCT and DST. Though one can find orthogonal matrix factorizations for the DCT and DST in [7], the resulting algorithms in [7] are not completely recursive and hence do not lead to simple recursive algorithms. An alternative factorization for DCT-I in [8] and DST-I in [9] can be seen in [10] and [11], but the factorizations in the latter papers do not solely depend on DCT I-IV or DST-I-IV. Moreover, [11] has used the same factorization for DST-II and IV as in [7]. The authors in [9], [12], however, produce radix-2, completely recursive, and stable DST I-IV algorithms. Furthermore, [12] established a connection between algebraic operations used in the sparse and scaled orthogonal factorization of DST-IIV matrices and signal flow graphs. We refer to many more historical remarks on DST factorizations and the corresponding signal flow graphs in [12]. We note here that the DCT algorithms in [8], [9], and [13] and the relationship between cosine and sine transform matrices in [10] can be used to obtain the DST algorithms in [12]. On the other hand, one can use an elegant factorization for Chebyshev-like Vandermonde matrices to factor the DCT and DST matrices as described in [14]. However, there are no simplest sparse factors, no explicit algorithms, and no complexity analysis to compute DCT and DST matrices by a vector in [14]. In this paper, we obtain simple factors, i.e., diagonal, bidiagonal, sparse, and scaled orthogonal factors, to compute DST matrices by a vector. Additionally, we present self-recursive/completely-recursive and radix-2 DST algorithms with the lowest multiplication complexity in the literature. In recent years, developments in calculating the DST have widely broadened the algorithm’s applications. Methods based upon the recursive nature of Chebyshev-polynomials have led to a drastic decrease in hardware complexity for applications that only require the transmission of a few key transform coefficients, creating a DST algorithm that is suitable for very large scale integrated circuits (VLSI) [15]–[17]. Similarly, a first-order moment-based approach has been developed and can be used to calculate the 1-dimensional DST-II without the need for multiplications by using special accumulator arrays and without having explicit factors for the DST-II to decrease computation time for certain applications [18]. For complex inputs, sparse factorization-based methods offer an extremely reduced flop count when compared to the brute-force method. The DST-II algorithms proposed in [19] take this a step further by developing a radix-2 based even-odd output vector at each stage, but without having explicit factorization for the DST-II matrix for any $n$.

The development of the convolution-multiplication property for the families of DSTs and DCTs led way for the DST to become an alternative to the DFT in many filtering applications [20]. Similarly, the duality theorem proposed in [21] and [22] now allows for the avoidance of calculating the inverse DST (IDST) in applications where similar signals are commonly encountered. Recently, the DST has expanded to applications including underwater signal transmission [23], [24], analysis and noise estimation of speech [25], finite impulse response (FIR) frequency filtering [26], and feature extraction of modulated signals in orthogonal frequency division multiplexing (OFDM) systems [27]. Similarly, the noise filtering capabilities have led to a new DST-based approach to measuring the volatility of stock prices [28], [29]. In hardware applications, the DST provides advantages over DCT-based approaches for statistical processes like the first-order Markov sequences with specific boundary conditions, and it achieves a lower bit error rate (BER) than the DCT for signals with low correlation [30]. It has also been shown that the DST may be implemented in the design of fractional order differentiators [31]. With the ability to develop DST algorithms with regularity, modularity, pipelining capability, and local connectivity, the DST will continue to expand its role in VLSI implementation [15]–[19], [26], [30], [31].

Although the DCT has traditionally played the pivotal role of frequency decomposition for data compression in methods like JPEG and MPEG, recent works have suggested that the DST may be applied to develop new methods of compression with increased performance for image and audio analysis as well as high-efficiency video encoding (HEVC) [22], [32]–[35]. The DST has proven to be an effective tool in the medical field as well, combining multiple images of different modes into a singular, highly detailed image for more accurate diagnoses [36], [37]. Expansion upon the fractional DST has also broadened the transform’s role in the field of cyber security, particularly for applications in image encryption [38]–[40]. It was shown in [38] that the DST increases the entropy of encryption schemes that rely on processing within transform domains. This paper explores this concept further in Section VIII.

We have observed that the lowest multiplicative complexity, radix-2, and recursive DST algorithms having diagonal, bidiagonal, sparse, and scaled orthogonal matrices are missing elements in the literature. Thus, in this paper, we follow a similar approach in deriving DCT II/III algorithms in [41] and the DCT I/IV algorithms in [42] to obtain novel lowest multiplication complexity, recursive, and radix-2 DST algorithms execute via diagonal, bidiagonal, sparse, and scaled orthogonal matrices. We also obtain novel relationships between DST-II and DST-IV matrices, and DST-I and DST-III matrices in terms of diagonal, bidiagonal, and sparse matrices.

We first introduce frequently used notation in Section II. In Section III, we obtain simple, diagonal, bidiagonal, sparse, and scaled orthogonal factors for the DST matrices. In Section IV, we derive relationships between DST-I and DST-III matrices, and also between DST-II and DST-IV matrices, using bidiagonal, diagonal, and sparse matrices. Next, in Section V, we state recursive and radix-2 DST algorithms to compute the DST matrices by a vector based on the factorization of the DST matrices described in Section III. In Section VI, we establish arithmetic complexities for
computing the proposed DST algorithms. Within the same section, we compare the arithmetic complexities in computing the proposed DST algorithms with some known DST algorithms. In Section VII, we show the connection between the algebraic operations used to compute the proposed DST algorithms and the signal flow graph building blocks to provide an architecture for the DST-based integrated circuit design. Finally, we describe in Section VIII an application of the proposed DST algorithms within a DRPE image encryption scheme.

II. FREQUENTLY USED NOTATIONS

First, we provide notations in [14] to distinguish and show the novelty of the proposed DST matrix factorization in comparison to the results in [14]. Next, we will introduce additional notations that we use throughout the paper.

A. FACTORIZATION FOR DST-I AND DST-II IN [14]

Here, we provide formulas in computing DST-I/II matrices in [14]. The relationship \( F_n = W_0 \cdot V_0 \), where \( F_n \) is the DCT or DST matrix of types I-IV, \( W_0 \) is the weight matrix, and \( V_0 \) is the Chebyshev-like polynomial Vandermonde matrix for the DCT or DST matrices, was first introduced in [42]. Later in [14], the polynomial Vandermonde matrix \( V_0 = [Q_k(x_i)]_{k=1}^{n-1} \) was further split into self contained factors to obtain factorization for the DCT and DST matrices.

The weight matrix for DST-I in [14] and [43] for odd \( N \) is given by

\[
W_{Q_I} = \sqrt{\frac{2}{N+1}} \cdot \text{diag} \left[ \sin \left( \frac{\pi}{N+1} \right), \sin \left( \frac{2\pi}{N+1} \right), \ldots, \sin \left( \frac{N\pi}{N+1} \right) \right].
\]

(1)

The weight matrix for DST-II in [14] and [43] for even \( n \) is given by

\[
W_{Q_{II}} = \sqrt{\frac{2}{n}} \cdot \text{diag} \left[ \sin \left( \frac{\pi}{2n} \right), \sin \left( \frac{\pi}{n} \right), \ldots, \sin \left( \frac{(n-1)\pi}{2n} \right), \frac{1}{\sqrt{2}} \sin \left( \frac{\pi}{2} \right) \right].
\]

(2)

The self-contained factorization for the Chebyshev-like polynomial Vandermonde matrix for DST-I (for odd \( N \)) and DST-II (for even \( n \)) was derived in [14] and stated as follows

\[
V_{Q_I/II} = \begin{bmatrix} V_{\text{odd}} - V_{\text{even}} \end{bmatrix} H_{Q_I/II},
\]

where

\[
H_{Q_I} = \begin{bmatrix} I_{\frac{N-1}{2}, \frac{N+1}{2}} & -I_{\frac{N-1}{2}, \frac{N+1}{2}} \\ I_{\frac{N+1}{2}, \frac{N-1}{2}} & -I_{\frac{N+1}{2}, \frac{N-1}{2}} \end{bmatrix},
\]

(4)

\[
H_{Q_{II}} = \begin{bmatrix} I_{\frac{N-1}{2}, \frac{N+1}{2}} & I_{\frac{N+1}{2}, \frac{N-1}{2}} \\ I_{\frac{N+1}{2}, \frac{N-1}{2}} & I_{\frac{N-1}{2}, \frac{N+1}{2}} \\ 0 & 0 \\ 0 & 0 \end{bmatrix},
\]

(5)

\( \tilde{T} \) is the anti-diagonal matrix. Moreover, the Chebyshev-like polynomial Vandermonde matrix \( V_{even} = [Q_k(d_{2k})]_{k=1}^{n-1} \) is defined using \( \hat{U}_k(d) = U_k(d) \) for DST-I and \( V_{even} = [Q_k(d_{2k})]_{k=1}^{n-1} \) is defined using \( \hat{U}_k(d) = U_k(d) - U_{k-1}(d) \) for DST-II with respect to the classical Chebyshev polynomials \( U_k(d) = \sin((k+1)\alpha(d)) \) for \( k = 0, 1, \ldots, N-1 \) or \( n-1 \), respectively for DST-I and DST-II.

The factorization for \( V_{odd} \) was derived for DST-I (for odd \( N \)) and has the form

\[
V_{odd} = G_{SI} \begin{bmatrix} V_{\text{even}} \\ 0 \end{bmatrix} \hat{G}_{SI},
\]

where

\[
G_{SI} = \begin{bmatrix} \sin(\text{arccos}(d_1)) & \frac{(-1)^0}{\sin(\text{arccos}(d_1))} \\ \sin(\text{arccos}(d_1)) & \frac{1}{\sin(\text{arccos}(d_1))} \\ \sin(\text{arccos}(d_{2k-1})) & \frac{(-1)^{k-1}}{\sin(\text{arccos}(d_{2k-1}))} \\ \sin(\text{arccos}(d_{2k})) & \frac{(-1)^{k-1}}{\sin(\text{arccos}(d_{2k}))} \\ \cdots & \cdots \\ \sin(\text{arccos}(d_{n-1})) & \frac{(-1)^{n-2}}{\sin(\text{arccos}(d_{n-1}))} \\ \sin(\text{arccos}(d_{n})) & \frac{(-1)^{n-2}}{\sin(\text{arccos}(d_{n}))} \end{bmatrix},
\]

(7)

and

\[
\hat{G}_{SI} = \frac{1}{2} \cdot \text{diag} \left[ \frac{1}{T_1(d_1)}, \frac{1}{T_2(d_1)}, \ldots, \frac{1}{T_{n-1}(d_1)} \right],
\]

(8)

with respect to the nodes

\[
\{d_k\}_{k=1}^{N} = \left\{ \cos \left( \frac{k\pi}{N+1} \right) \right\}_{k=1}^{N},
\]

(9)

and the classical Chebyshev polynomials \( T_k(d) = \cos(k \text{arccos } d) \).

The factorization for \( V_{odd} \) was derived for DST-II (for even \( n \)) and has the form

\[
V_{odd} = G_{SI} V_{even} \hat{G}_{SI},
\]

(10)

where

\[
G_{SI} = \begin{bmatrix} \sin(\text{arccos}(d_1)) & \frac{1}{\sin(\text{arccos}(d_1))} \\ \sin(\text{arccos}(d_1)) & \frac{1}{\sin(\text{arccos}(d_1))} \\ \sin(\text{arccos}(d_{2k-1})) & \frac{1}{\sin(\text{arccos}(d_{2k-1}))} \\ \sin(\text{arccos}(d_{2k})) & \frac{1}{\sin(\text{arccos}(d_{2k}))} \\ \cdots & \cdots \\ \sin(\text{arccos}(d_{n-1})) & \frac{1}{\sin(\text{arccos}(d_{n-1}))} \\ \sin(\text{arccos}(d_{n})) & \frac{1}{\sin(\text{arccos}(d_{n}))} \end{bmatrix},
\]

(11)

and

\[
\hat{G}_{SI} = \frac{1}{2} \cdot \text{diag} \left[ \cos \left( \frac{\pi}{2n} \right), \cos \left( \frac{3\pi}{2n} \right), \ldots, \cos \left( \frac{(n-1)\pi}{2n} \right) \right],
\]

(12)
with respect to the nodes
\[ \{d_k\}_{k=1}^n = \left\{ \cos \left( \frac{k \pi}{n} \right) \right\}_{k=1}^n. \]  

By using the above factorization formulas for the Chebyshev-like polynomial Vandermonde matrices and the relationship \( F_n = \hat{W}_Q \cdot V_Q \) in [43], the following results were obtained in [14] to factor DST-I (for odd \( N \)) and DST-II (for even \( N \)) matrices
\[ S_N^I = W_Q^{\frac{1}{2}} \begin{bmatrix} \text{odd - even permutation} \end{bmatrix}^{-1} \begin{bmatrix} W_{N-1}^{-\frac{1}{2}} \end{bmatrix} I \times \begin{bmatrix} S_{N-1}^{-\frac{1}{2}} \end{bmatrix} G_{S^I}^{-1} W_{N-1}^{-\frac{1}{2}} S_{N-1}^{-\frac{1}{2}} \hat{G}_{S^I} \end{bmatrix} H_{S^I}. \]  
\[ S_N^II = W_Q^{\frac{1}{2}} \begin{bmatrix} \text{odd - even permutation} \end{bmatrix}^{-1} \begin{bmatrix} W_{N}^{-\frac{1}{2}} \end{bmatrix} I \times \begin{bmatrix} S_{N}^{-\frac{1}{2}} \end{bmatrix} G_{S^II}^{-1} W_{N}^{-\frac{1}{2}} S_{N}^{-\frac{1}{2}} \hat{G}_{S^II} \end{bmatrix} H_{S^II}. \]  

The factorization formulas (14) and (15) were obtained by using the factorization formula (3) with the replacement of \( V_{Q_{S^III}} \) by \( W_{N-1}^{-\frac{1}{2}} S_{N-1}^{-\frac{1}{2}} \) (from [43]), \( V_{Q_{S^IV}} \) by \( W_{N}^{-\frac{1}{2}} S_{N}^{-\frac{1}{2}} \) and \( V_{Q_{S^II}} \) by \( W_{N}^{-\frac{1}{2}} S_{N}^{-\frac{1}{2}} \) (from [43]), respectively. Although it says in [14] that the first matrix in the factorization (3) is the odd-even permutation matrix, it must actually be the transpose (inverse) of the odd-even permutation matrix. We also note here that the factorization formulas to compute DST-III/IV matrices in [14] are not included because the proposed DST-III/IV matrix factorization formulas are completely different from the factorization formulas in [14].

**B. OTHER NOTATION**

We state here diagonal, bidiagonal, sparse, and orthogonal matrices which are frequently used in this paper. For a given vector \( x \in \mathbb{R}^n \) and an integer \( n \geq 3 \), let us introduce an even-odd permutation matrix \( P_n \) by
\[ P_n x = \begin{cases} [x_0, x_2, \ldots, x_{n-2}, x_1, x_3, \ldots, x_{n-1}]^T, & \text{if } n \text{ is even} \\ [x_0, x_2, \ldots, x_{n-1}, x_1, x_3, \ldots, x_{n-2}]^T, & \text{if } n \text{ is odd}. \end{cases} \]

For even \( n \) s.t. \( n \geq 4 \), we introduce a sparse matrix
\[ \hat{B}_n = \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ \cdot & \cdot \\ 1 & 1 \\ -1 & -1 \end{bmatrix}. \]  

a bidiagonal matrix
\[ B_\frac{n}{2} = \begin{bmatrix} 1 & 1 \\ 1 & \ddots & 1 \\ \ddots & \ddots & \ddots & \ddots \\ 1 & \ddots & \ddots & \ddots & 1 \\ -1 & \cdots & \cdots & \cdots & -1 \end{bmatrix}, \]  

diagonal matrices
\[ \hat{W}_\frac{n}{2} = \begin{bmatrix} \cos(\frac{(n-2k+1)\pi}{2}) \end{bmatrix}_{k=1}^{\frac{n}{2}} \]  

and
\[ \hat{W}_\frac{n}{2} = \begin{bmatrix} \frac{\csc(\frac{(n-2k+1)\pi}{2})}{2} \end{bmatrix}_{k=1}^{\frac{n}{2}} \]  

sparse and orthogonal matrices
\[ H_n = \begin{bmatrix} I_\frac{n}{2} & \hat{I}_\frac{n}{2} \\ -\hat{I}_\frac{n}{2} & I_\frac{n}{2} \end{bmatrix} = \frac{1}{\sqrt{2}} H_{n-1}, \]  
\[ \hat{H}_{n-1} = \begin{bmatrix} I_{\frac{n-1}{2}} & -\hat{I}_{\frac{n-1}{2}} \\ \hat{I}_{\frac{n-1}{2}} & I_{\frac{n-1}{2}} \end{bmatrix} = \frac{1}{\sqrt{2}} H_{n-1}, \]  
\[ V_n = \begin{bmatrix} 1 & \text{diag} S_{\frac{n}{2}} & \text{diag} C_{\frac{n}{2}} \end{bmatrix} \begin{bmatrix} I_{\frac{n}{2}} \\ \hat{I}_{\frac{n}{2}} \\ \hat{I}_{\frac{n}{2}} \end{bmatrix}, \]

where \( J_n = \text{diag} \left( (-1)^k \right)_{k=0}^{n-1} \) a rotational-reflection matrix
\[ Q_n = \begin{bmatrix} J_\frac{n}{2} & I_\frac{n}{2} & -\hat{I}_\frac{n}{2} & \hat{I}_\frac{n}{2} \end{bmatrix} \begin{bmatrix} \text{diag} S_{\frac{n}{2}} & \text{diag} C_{\frac{n}{2}} \end{bmatrix} \begin{bmatrix} I_{\frac{n}{2}} \\ -\hat{I}_{\frac{n}{2}} \\ \hat{I}_{\frac{n}{2}} \\ -\hat{I}_{\frac{n}{2}} \end{bmatrix}. \]
where
\[
C_2 = \left[ \cos \frac{(2k+1)\pi}{4n} \right]_{k=0}^{2n-1},
\]
\[
S_2 = \left[ \sin \frac{(2k+1)\pi}{4n} \right]_{k=0}^{2n-1},
\]
and a matrix with three liftings (we will show later that the following sparse and orthogonal matrix, \(\tilde{Q}_n\), is the same as the matrix \(Q_n\) with three lifting steps)
\[
\tilde{Q}_n = \left[ \begin{array}{c|c}
J_2 & J_2 \left( \text{diag} T_2 \right) I_2 \\
0 & -I_2 \end{array} \right] \left[ \begin{array}{c|c}
I_2 & 0 \\
-\tilde{I}_2 \left( \text{diag} S_2 \right) I_2 & 0
\end{array} \right] \times \left[ \begin{array}{c|c}
\text{diag} T_2 & I_2 \\
I_2 & 0
\end{array} \right],
\]
(25)
where
\[
T_2 = \left[ \frac{\tan \left( \frac{(2k+1)\pi}{4n} \right)}{2^n} \right]_{k=0}^{2n-1}.
\]

III. SPARSE, ORTHOGONAL, BIDIAGONAL, DIAGONAL, AND SELF-CONTAINED FACTORS FOR DST MATRICES

In this section, we will propose formulas to factor DST I-IV matrices into the product of sparse, diagonal, bidiagonal, and orthogonal matrices. Among the factorization formulas, the self-contained factorization of the DST-II matrix will be used to establish a factorization formula for the DST-III matrix. Also, an orthogonal matrix with three lifting steps and the factorization formula of the DST-II matrix are utilized to obtain a factorization formula for the DST-IV matrix.

A. SPARSE, ORTHOGONAL, DIAGONAL, AND SELF-CONTAINED FACTORS TO COMPUTE DST-I MATRIX

Let us state a novel self-enclosed factorization to compute the DST-I matrix using simple, diagonal, sparse, and orthogonal matrices.

Lemma 1 (Self-enclosed DST-I factorization): For an even integer \(n \geq 4\), the matrix \(S_{n-1}^I\) can be factored into the form
\[
S_{n-1}^I = \tilde{P}_n \left[ \begin{array}{c|c}
\tilde{I}_2 & 0 \\
0 & \tilde{B}_2
\end{array} \right] \left[ \begin{array}{c|c}
S_{n-1}^I & 0 \\
0 & \tilde{W}_2
\end{array} \right] \tilde{H}_{n-1},
\]
(26)
Proof: We factor the weight matrix \(W_{Q_{SI}}\) defined in the equation (1) into the product of matrices s.t.
\[
W_{Q_{SI}} = \sqrt{\frac{\pi}{n}} \tilde{P}_n \left[ \begin{array}{c|c}
W_{\text{even}} & 0 \\
0 & W_{\text{odd}}
\end{array} \right] \tilde{P}_{n-1},
\]
(27)
where \(W_{\text{even}} = \text{diag} \left[ \sin \frac{(2k)\pi}{n} \right]_{k=1}^{n-1}\) and \(W_{\text{odd}} = \text{diag} \left[ \sin \frac{(2(k-1))\pi}{n} \right]_{k=1}^{n-1}\). Now, we use the trigonometric identities to factor \(G_{SI'}\) in equation (7) and \(\hat{G}_{SI'}\) in equation (8) into the product of diagonal and sparse matrices s.t.
\[
G_{SI'} = W_{\text{even}}^{-1} \tilde{B}_2 \left[ \begin{array}{c|c}
W_{\text{even}} & 0 \\
0 & 1
\end{array} \right]
\]
and
\[
\hat{G}_{SI'} = \frac{1}{2} \left[ \tilde{I}_2 \tilde{I}_2^{-1} \right] \left[ \begin{array}{c|c}
W_{h}^{-1} & 0 \\
0 & \sqrt{2}
\end{array} \right] \tilde{I}_2^{-1}
\]
(29)
where \(W_h = \text{diag} \left[ \sin \frac{(k\pi}{n} \right]_{k=1}^{n-1}\) and \(\tilde{B}_2\) is defined in (16). Hence by (27), (28), (29), \(S_{n-1} = W_{Q_{SI}} \tilde{Q}_{SI'}, \tilde{Q}_{SI'} = \tilde{P}_n \left[ \begin{array}{c|c}
V_{\text{even}} & 0 \\
0 & V_{\text{odd}}
\end{array} \right] \hat{H}_{SI'}, \text{and using the } V_{\text{odd}} \text{ defined in (6), we can obtain}
\[
S_{n-1} = \sqrt{\frac{\pi}{n}} \tilde{P}_n \left[ \begin{array}{c|c}
W_{\text{even}} & 0 \\
0 & \tilde{B}_2 \left[ \begin{array}{c|c}
W_{\text{even}} & 0 \\
0 & 1
\end{array} \right] \tilde{W}_2
\end{array} \right] \tilde{H}_{n-1}.
\]
(30)

By simplifying \(\tilde{I}_2\tilde{I}_2^{-1} W_{\text{even}}^{-1} \tilde{I}_2^{-1}\) and \(\hat{H}_{SI'}\) (defined in (4)), we can obtain \(W_h\) in (18) and \(\tilde{H}_{n-1}\) in (21), respectively. By following [43], we can obtain \(S_{n-1}^I = \sqrt{\frac{\pi}{n}} W_{\text{even}}\). Thus, we substitute this relationship into the equation (30) and modify the last entry in the (2,2) block of the middle matrix in (30). This is modified to fix the scaling because the \(\left(\frac{n}{2}, \frac{n}{2}\right)\) entry of \(\hat{G}_{SI'}\) is \(\frac{1}{\sqrt{2}}\) and scaling factor of \(S_{n-1}^I\) is \(\sqrt{\frac{\pi}{n}}\). Then, we add a scaling factor of \(\frac{1}{\sqrt{n}}\) to \(\tilde{H}_{n-1}\) (i.e. modify the \((n-1, \frac{n}{2})\) entry of \(\tilde{H}_{n-1}\) to obtain the factorization proposed in (26).

Remark 2: By comparing the factorizations of the DST-I matrix in (14) and (26) (i.e. distinguishing the proposed self-contained factorization of the DST-I matrix from the DST-I factorization in [14]) one can observe that (a), the weight matrix is congregated in (26) as opposed to scattered weight matrices in (14), (b), the weight matrix \(W_{Q_{SI}}\) in (2) was factored into the product of permutation and odd/even weight matrices in (27), (c), the matrix \(G_{SI'}\) in (29) has been simplified and factored into the product of sparse and diagonal matrices as opposed to the matrix \(G_{SI'}\) in (7), (d), the matrix \(\hat{G}_{SI'}\) in (29) has been simplified and factored into the product of diagonal matrices as opposed to \(\hat{G}_{SI'}\) in (8), (f), the matrix \(\tilde{H}_{n-1}\) in (26) is orthogonal as opposed to the nonorthogonal matrix \(H_{SI'}\) in (4). Thus, the proposed self-contained factorization of the DST-I matrix in (26) is in the simplest form as opposed to the factorization of the DST-II matrix in (14).

Remark 3: The factorization of the DST-I matrix established in Lemma 1 also yields to the following alternative factorization formula through a permutation
\[
S_{n-1} = \tilde{P}_n \left[ \begin{array}{c|c}
\tilde{B}_2 & 0 \\
0 & I_2^{-1}
\end{array} \right] \left[ \begin{array}{c|c}
\tilde{W}_2 & 0 \\
0 & I_2^{-1}
\end{array} \right] \tilde{H}_{n-1},
\]
(31)
where \( P_{n-1} \) is an even-odd permutation matrix and \( \tilde{H}_{n-1} \) is defined in (22).

### B. BIDIODAGNAL, DIAGONAL, ORTHOGONAL, AND SELF-CONTAINED FACTORS TO COMPUTE DST-II MATRIX

Let us state a novel self-contained factorization to compute the DST-II matrix using sparse, bidiagonal, diagonal, and orthogonal matrices.

**Lemma 4 (Self-enclosed DST-II factorization):** For an even integer \( n \geq 4 \), the matrix \( S_{II} \) can be factored into the form

\[
S_{II}^n = \tilde{P}_n \begin{bmatrix} I_n & 0 & 0 & 0 \\ 0 & S_{II} \hat{I}_2 & 0 & 0 \\ 0 & 0 & S_{II} \hat{I}_2 & 0 \\ 0 & 0 & 0 & W_2 \end{bmatrix} \tilde{H}_n. \tag{32}
\]

**Proof:** We factor the weight matrix \( W_{Q_{II}} \) defined in the equation (2) into the product of matrices s.t.

\[
W_{Q_{II}} = \sqrt{\frac{\pi}{n}} \tilde{P}_n \begin{bmatrix} W_{even} & 0 & 0 & W_{odd} \end{bmatrix} \tilde{P}_n, \tag{33}
\]

where \( W_{even} = \text{diag}\left[ \sin\left(\frac{k\pi}{n}\right) \right]_{k=1}^{\frac{n-1}{2}} \) and \( W_{odd} = \text{diag}\left[ \sin\left(\frac{(2k-1)\pi}{n}\right) \right]_{k=1}^{\frac{n-1}{2}} \). Now, we can use the inverse, reciprocal, and cofunction trigonometric identities to factor \( G_{II} \) in the equation (11) and \( \tilde{G}_{II} \) in the equation (12) into the product of bidiagonal, diagonal, and sparse matrices s.t.

\[
G_{II}^n = W_{odd} B_2 W_{even} \quad \text{and} \quad \tilde{G}_{II}^n = \frac{1}{\sqrt{n}} I_2 W_{odd} I_2, \tag{34}
\]

where \( B_2 \) is defined in (17). While factoring the matrix \( G_{II}^n \), the \((\frac{2}{n}, \frac{2}{n})\) entry of \( B_2 \) is modified into \( \sqrt{2} \) because the \((\frac{2}{n}, \frac{2}{n})\) entry of \( W_{even} \) is defined as \( \frac{1}{\sqrt{2}} \) in (33). Hence by (33), (34),

\[
S_{II}^n = W_{Q_{II}} V_{Q_{II}}, \quad V_{Q_{II}} = \tilde{P}_n \begin{bmatrix} W_{even} \\ 0 \\ B_2 W_{even} V_{odd} I_2 \end{bmatrix} H_{II}, \quad \text{and using the} \quad V_{odd} \text{defined in (10)}, \quad \text{we can obtain}
\]

\[
S_{II}^n = \sqrt{\frac{\pi}{n}} \tilde{P}_n \begin{bmatrix} W_{even} V_{even} & 0 & 0 & W_{odd} V_{odd} \end{bmatrix} \tilde{H}_n. \tag{35}
\]

where \( \tilde{H}_n \) is given in (20). By following [43], we can obtain

\[
S_{even}^{II} = \sqrt{\frac{\pi}{n}} W_{even} V_{even}. \tag{36}
\]

Thus, from the equations (35) and (36) we get

\[
S_{II}^n = \tilde{P}_n \begin{bmatrix} S_{II}^n & 0 & 0 & 0 \\ 0 & I_n & S_{II} \hat{I}_2 & 0 \\ 0 & 0 & S_{II} \hat{I}_2 & 0 \\ 0 & 0 & 0 & W_2 \end{bmatrix} \tilde{H}_n. \tag{37}
\]

Finally, by using the above equation and the trigonometric identities to simplify the formula \( I_2 W_{odd} I_2 \), we can obtain the factorization formula proposed in (32).

**Remark 5:** By comparing the factorizations of the DST-II matrix in (15) and (32) (i.e. distinguishing the proposed self-contained factorization of the DST-II matrix from the DST-II factorization in [14]) one can observe that (a), there are not many weight matrices spread in (32) like in (15), (b), a factorization for \( W_{Q_{II}} \) is presented via a permutation and odd/even weight matrices in (33) as opposed to \( W_{Q_{II}} \) in (2), (c), a factorization for \( G_{II} \) is presented via a simple bidiagonal and diagonal matrices in (34) as opposed to \( G_{II} \) in (11), (d), a factorization for \( \tilde{G}_{II} \) is presented via antidiagonal and diagonal matrices in (34) as opposed to \( \tilde{G}_{II} \) in (12), (e), an orthogonal matrix \( H_n \) is given in (20) as opposed to the scaled orthogonal matrix \( H_{Q_{II}} \) in (15), and (f), identified the exact permutation matrix for \( V_{Q_{II}} \) within the Lemma 4 as opposed to \( V_{Q_{II}} \) in (3). Thus, we have expressed an exact, simplified, and self-contained factorization formula to compute the DST-II matrix using bidiagonal, diagonal, sparse, and orthogonal factors in (32) as opposed to (15).

**Remark 6:** The factorization of the DST-II matrix established in Lemma 1 also yields to the following alternative factorization formula through a permutation

\[
S_{II}^n = P_n^T \begin{bmatrix} B_n & 0 & 0 \\ 0 & I_2 & S_{II}^n \hat{I}_2 \\ 0 & 0 & W_2 \end{bmatrix} H_n, \tag{38}
\]

where \( P_n \) is an even-odd permutation matrix and

\[
H_n = \frac{1}{\sqrt{n}} \begin{bmatrix} I_2 & \tilde{T}_n \tilde{I}_2 \\ \tilde{T}_n & I_2 \end{bmatrix}. \]

### C. BIDIODAGNAL, DIAGONAL, ORTHOGONAL, AND SELF-CONTAINED FACTORS TO COMPUTE DST-III MATRIX

Let us state a novel self-contained factorization to compute the DST-III matrix using sparse, bidiagonal, diagonal, and orthogonal matrices.

**Corollary 7 (Self-enclosed DST-III factorization):** For an even integer \( n \geq 4 \), the matrix \( S_{III} \) can be factored into the form

\[
S_{III}^n = H_n^T \begin{bmatrix} W_2 & 0 & 0 \\ 0 & I_2 & S_{III} \hat{I}_2 \\ 0 & 0 & S_{III} \hat{I}_2 \\ 0 & 0 & 0 & B_2 \end{bmatrix} P_n. \tag{39}
\]

**Proof:** This is trivial by Lemma 4, \( S_{III} = [S_{II}]^T \), and Remark 6.

### D. SPARSE, ORTHOGONAL, DIAGONAL, BIDIODAGNAL, AND RECURSIVE FACTORS TO COMPUTE DST-IV MATRIX

We propose a novel factorization formula to compute the DST-IV matrix using bidiagonal, diagonal, sparse, and orthogonal matrices. The proposed factorization of the DST-IV matrix is based on the self-enclosed DST-II factorization in Lemma 4. Before we proceed, let us recall the DST-IV...
matrix factorization formula in [9], [12] s.t.

\[ S_n^{IV} = P_n^T V_n \left[ \begin{array}{cc} S_n^{II} & 0 \\ 0 & S_n^{II} \end{array} \right] Q_n. \]  

(40)

To derive a novel factorization formula to compute the DST-IV matrix, we will use the above factorization formula and apply three lifting steps [44] for the rotation matrix \[ \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \] and the rotation/reflection matrix \[ \begin{bmatrix} \cos \theta & \sin \theta \\ -\sin \theta & -\cos \theta \end{bmatrix}. \] Let us first derive a factorization formula for the rotational/rotational-reflection butterfly matrix \( Q_n \) defined in (24).

**Lemma 8** (Butterfly matrix \( Q_n \) is the same as the matrix with three lifting steps \( \tilde{Q}_n \)): Let \( n \geq 2 \) be an even integer. The matrix with three lifting steps, \( \tilde{Q}_n \), is the same as the rotational/rotational-reflection matrix \( Q_n \) and can be factored in the form:

\[ \tilde{Q}_n = \begin{bmatrix} J_2 & J_2 \text{diag}(T_2) \tilde{I}_2 \\ 0 & -I_2 \end{bmatrix} \begin{bmatrix} I_2 & 0 \\ -I_2 \text{diag}(S_2) I_2 & I_2 \end{bmatrix} \begin{bmatrix} \text{diag}(T_2) \tilde{I}_2 & 0 \\ 0 & \tilde{I}_2 \end{bmatrix}. \]  

(41)

where \( T_2 = \left[ \tan \left( \frac{(2k + 1)\pi}{8n} \right) \right]_{k=0}^{n-1}. \)

**Proof:** Let us consider the block products in \( \tilde{Q}_n \) explicitly:

\[ Q_n = \begin{bmatrix} J_2 \text{diag}(T_2) (2I_2 - \text{diag}(S_2)) \text{diag}(T_2) \tilde{I}_2 & (I_2 - \text{diag}(T_2) \text{diag}(S_2)) \tilde{I}_2 \\ -I_2 (I_2 - \text{diag}(T_2) \text{diag}(S_2)) & I_2 \text{diag}(S_2) \tilde{I}_2 \end{bmatrix}. \]  

(42)

To verify the equivalency of \( \tilde{Q}_n \) and \( Q_n \), we will consider the corresponding blocks in both matrices.

We begin with the (1,1) block of \( Q_n \) in (42) s.t.

\[ J_2 \text{diag}(T_2) (2I_2 - \text{diag}(S_2)) \text{diag}(T_2) \tilde{I}_2 = J_2 \text{diag} \left[ \left( \tan \left( \frac{(2k + 1)\pi}{8n} \right) \right) \right]_{k=0}^{n-1} \times \left( 2 - \tan \frac{(2k + 1)\pi}{8n} \sin \frac{(2k + 1)\pi}{4n} \right)_{k=0}^{n-1} = J_2 \text{diag} \left[ 2 \tan \frac{(2k + 1)\pi}{8n} \cos^2 \frac{(2k + 1)\pi}{8n} \right]_{k=0}^{n-1} = J_2 \text{diag} \left[ \sin \frac{(2k + 1)\pi}{4n} \right]_{k=0}^{n-1} = J_2 \text{diag}(S_2). \]  

(43)

Second, we consider the (1,2) block of \( \tilde{Q}_n \) in (42) s.t.

\[ J_2 (I_2 - \text{diag}(T_2) \text{diag}(S_2)) \tilde{I}_2 = J_2 \text{diag} \left[ 1 - \tan \frac{(2k + 1)\pi}{8n} \sin \frac{(2k + 1)\pi}{4n} \right]_{k=0}^{n-1} \tilde{I}_2 = J_2 \text{diag} \left[ 1 - 2 \sin^2 \frac{(2k + 1)\pi}{8n} \right]_{k=0}^{n-1} \tilde{I}_2 = J_2 \text{diag} C_2 \tilde{I}_2. \]  

(44)

Similarly, the (2,1) block of \( \tilde{Q}_n \) in (42) can be expressed as

\[ \tilde{Q}_n = \begin{bmatrix} J_2 & J_2 \text{diag}(T_2) \tilde{I}_2 \\ 0 & -I_2 \end{bmatrix} \begin{bmatrix} I_2 & 0 \\ -I_2 \text{diag}(S_2) I_2 & I_2 \end{bmatrix} \begin{bmatrix} \text{diag}(T_2) \tilde{I}_2 & 0 \\ 0 & \tilde{I}_2 \end{bmatrix}. \]  

(45)

Finally, the (2,2) block of \( \tilde{Q}_n \) in (42) can be expressed as

\[ \tilde{I}_2 \text{diag}(S_2) \tilde{I}_2 = \text{diag} \left[ \sin \frac{(n - 2k - 1)\pi}{4n} \right]_{k=0}^{n-1} = \text{diag}(I_2 S_2). \]  

(46)

Thus, from (43), (44), (45), and (46), we prove the claim.

Next, we use the immediate result to state a novel recursive factorization formula to compute the DST-IV matrix using the DST-II, sparse, and orthogonal matrices. We recall here that the DST-II matrix can further be factored into the product of sparse, bidiagonal, diagonal, and orthogonal matrices as shown in Lemma 4 followed by Remark 6.

**Lemma 9** (Recursive DST-IV Factorization): For an even integer \( n \geq 4 \), the matrix \( S_n^{IV} \) can be factorized into the form

\[ S_n^{IV} = P_n^T V_n \left[ \begin{array}{cc} S_n^{II} & 0 \\ 0 & S_n^{II} \end{array} \right] \tilde{Q}_n. \]  

(47)

where \( V_n \) is defined in (23) and \( \tilde{Q}_n \) is defined in (41).

**Proof:** Lemma 8 shows that \( \tilde{Q}_n \) in (41) is the same as \( Q_n \) in (24). This result together with the explicit derivation for the factorization of the DST-IV matrix in [9] and [12] gives the result (47).

**Remark 10:** The DCT I-IV matrix factorizations proposed in [41], [42] and the interweaving relationships among cosine and sine matrices in [10] can be utilized to obtain variants of the proposed DST I-IV matrix factorization formulas.

### IV. RELATIONSHIP AMONG DST MATRICES AND CONNECTION AMONG DST FACTORIZATION IN THE LITERATURE

We first present a relationship between DST-I and DST-III matrices and next between the DST-II and DST-IV matrices using bidiagonal, diagonal, and sparse matrices. After these relationships are established, we will observe connections among the proposed DST matrix factorizations with the existing factorization of the DST matrices.

**Lemma 11** (Relationship between \( S_n^{I-IV} \) and \( S_n^{II} \)): For an even integer \( n \geq 4 \), the matrix \( S_n^{II} \) is related to the matrix...
$S_{n}^{I}$ in the following way

\[
S_{n}^{III} = \tilde{B}_{n} \left[ S_{n}^{I-1} \sqrt{\frac{2}{n}} \right] \tilde{W}_{n}. \tag{48}
\]

**Proof:** Let us consider the product $A = \tilde{B}_{n} \left[ S_{n}^{I-1} \sqrt{\frac{2}{n}} \right] \tilde{W}_{n}$ explicitly:

\[
A = \frac{1}{\sqrt{n}} \begin{bmatrix}
1 & 1 & 1 & \cdots & 1 \\
1 & 1 & -1 & \cdots & 1 \\
0 & 0 & \cdots & 0 \\
\end{bmatrix} \cdot S_{n}^{III}.
\]

Let us evaluate the elements of $A$. First, we consider elements in the first row, but the first to the second-to-last column of (49), i.e., $j = 0$ and $k = 0, 1, \ldots, \frac{n}{2} - 2$.

\[
[A]_{0,k} = \frac{1}{\sqrt{n}} \cdot \sin \frac{2(k+1)\pi}{n} = \frac{2}{\sqrt{n}} \cdot \frac{\sin (k+1)\pi}{n} = S_{n}^{III}_{0,k}. \tag{50}
\]

Second, we consider elements in the last row, but the first to the second-to-last column of (49), i.e., $j = \frac{n}{2} - 1$ and $k = 0, 1, \ldots, \frac{n}{2} - 2$.

\[
[A]_{\frac{n}{2}-1,k} = \frac{1}{\sqrt{n}} \cdot \sin \frac{(\frac{n}{2} - 1)(k+1)\pi}{n} = \frac{2}{\sqrt{n}} \cdot \frac{\sin (\frac{n}{2} - 1)(k+1)\pi}{n} = S_{n}^{III}_{\frac{n}{2}-1,k}. \tag{51}
\]

Third, we observe elements in the last column of (49) (from the first to last row), i.e., $j = 0, 1, \ldots, \frac{n}{2} - 1$ and $k = \frac{n}{2} - 1$.

\[
[A]_{j,\frac{n}{2}-1} = \frac{1}{\sqrt{n}} \cdot (-1)^{j} \cdot \sqrt{2}
= \frac{2}{\sqrt{n}} \cdot \frac{1}{\sqrt{2}} \cdot \sin \frac{(2j+1)\pi}{n} = S_{n}^{III}_{j,\frac{n}{2}-1}. \tag{52}
\]

Finally, we consider elements in the second to second-to-last row and the first to second-to-last column of (49), i.e., $j = 1, 2, \ldots, \frac{n}{2} - 2$ and $k = 0, 1, \ldots, \frac{n}{2} - 2$.

\[
[A]_{j,k} = \frac{1}{\sqrt{n}} \left( \sin \frac{2(j+1)\pi}{n} \cdot \sin \frac{(n-2(k+1))\pi}{2n} \right)
= \frac{2}{\sqrt{n}} \left( \sin \frac{(2j+1)(k+1)\pi}{n} \cdot \cos \frac{(k+1)\pi}{n} \right)
= \frac{2}{\sqrt{n}} \cdot \frac{\sin (2j+1)(k+1)\pi}{n} = S_{n}^{III}_{j,k}. \tag{53}
\]

Thus, from (50), (51), (52), and (53), we prove the claim.

The immediate result is true for any even integer $n \geq 2$ satisfying the following relationship.

**Corollary 12 (Relationship between $S_{n-1}^{I}$ and $S_{n}^{III}$):** For an even integer $n \geq 2$, the matrix $S_{n}^{III}$ is related to the matrix $S_{n-1}^{I}$ in the following way

\[
S_{n}^{III} = \tilde{B}_{n} \left[ S_{n-1}^{I} \frac{1}{\sqrt{n}} \right] \tilde{W}_{n}. \tag{54}
\]

**Proof:** This is trivial from the Lemma 11.

**Lemma 13 (Relationship Between $S_{n}^{II}$ and $S_{n}^{IV}$):** For an even integer $n \geq 4$, the matrix $S_{n}^{IV}$ is related to the matrix $S_{n}^{II}$ in the following way

\[
S_{n}^{IV} = B_{n} S_{n}^{II} \left[ W_{n} \right]. \tag{55}
\]

**Proof:** Let us consider the product $M = B_{n} S_{n}^{II} \left[ W_{n} \right]$ explicitly

\[
M = \frac{1}{\sqrt{n}} \begin{bmatrix}
1 & 1 & \cdots & 1 \\
1 & 1 & \cdots & 1 \\
\vdots & \vdots & \ddots & \vdots \\
1 & 1 & \cdots & 1 \\
\end{bmatrix} \cdot \begin{bmatrix}
\sin \frac{\pi}{2} & \sin \frac{\pi}{2} & \cdots & \sin \frac{\pi}{2} \\
\sin \frac{\pi}{2} & \sin \frac{\pi}{2} & \cdots & \sin \frac{\pi}{2} \\
\vdots & \vdots & \ddots & \vdots \\
\sin \frac{\pi}{2} & \sin \frac{\pi}{2} & \cdots & \sin \frac{\pi}{2} \\
\end{bmatrix}.
\]

Let us evaluate the elements of $M$. First, we consider elements in the first row of (56), i.e., $j = 0$ and $k = 0, 1, \cdots, n/2 - 1$.

\[
[M]_{0,k} = \frac{1}{\sqrt{n}} \sin \left( \frac{2(k+1)\pi}{n} \right)
= \frac{2}{\sqrt{n}} \sin \left( \frac{2(k+1)\pi}{n} \right) = S_{n}^{IV}_{0,k}. \tag{57}
\]
Second, we consider elements between the second and the \((n/2 - 2)^{th}\) rows of (56), i.e., \(j = 1, 2, \ldots, n/2 - 2\) and \(k = 0, 1, \ldots, n/2 - 1\).

\[
[M]_{j,k} = \frac{1}{\sqrt{n}} \left[ 1 + \sin \left( \frac{(2j+1)\pi}{n} \right) \sin \left( \frac{(n-2k-1)\pi}{n} \right) \right]
\]

Thus, from (57), (58), and (59), we prove the claim.

The immediate result is true for any even integer \(n \geq 2\) satisfying the following relationship.

**Corollary 14 (Relationship between \(S_n^{I} I\) and \(S_n^{IV}\)):** For an even integer \(n \geq 2\), the matrix \(S_n^{IV}\) is related to the matrix \(S_n^{II}\) in the following way

\[
S_n^{IV} = B_n S_n^{II} \left[ W \right]_n.
\]

**Proof:** This is trivial from the Lemma 13.

---

**A. CONNECTION BETWEEN THE PROPOSED DST FACTORIZATION AND THE EXISTING DST FACTORIZATIONS**

In this section, we utilize the relationship among DST matrices established in Section IV to identify a relationship among the proposed DST factorization and the existing factorizations of DST matrices. The relationship between DST-II and DST-IV will be utilized to establish a connection between the proposed DST-II factorization with the existing DST-II factorization in [7], [9], [11], [12], [45]–[47]. We will also show the DST-III factorization in [7], [9]–[12], [46] can be seen as the proposed self-contained DST-III factorization. The relationship between DST-I and DST-III will be utilized to establish a connection between the proposed DST-I factorization with the existing DST-I factorization in [7], [9]–[12], [46], [47].

The following result shows that the proposed factorization of the DST-I matrix can be converted into the existing DST-I factorization in [7], [9]–[12], [46], [47].

**Corollary 15 (Connection between traditional DST-I and the proposed DST-I):** For an even integer \(n \geq 4\), the matrix \(S_n^{I} I\) can be factored into the form

\[
S_n^{I} I = P_n^{T} \left[ S_n^{II} \left[ W \right]_n \right] \hat{H}_{n-1}.
\]

**Proof:** This follows immediately from Lemma 1, Remark 3, and Lemma 11.

The following result shows that the proposed factorization of the DST-II matrix can be converted into the existing DST-II factorization in [7], [9], [11], [12], [45]–[47].

**Corollary 16 (Connection between traditional DST-II and the proposed DST-II):** For an even integer \(n \geq 4\), the matrix \(S_n^{II}\) can be factored into the form

\[
S_n^{II} = P_n^{T} \left[ S_n^{IV} \left[ W \right]_n \right] H_n.
\]

**Proof:** This follows immediately from Lemma 4, Remark 6, and Lemma 13.

The following result shows that the proposed factorization of the DST-III matrix can be converted into the existing DST-III factorization in [7], [9]–[12], [46].

**Corollary 17 (Connection between traditional DST-III and the proposed DST-III):** For an even integer \(n \geq 4\), the matrix \(S_n^{III}\) can be factored into the form

\[
S_n^{III} = H_n^{T} \left[ S_n^{IV} \left[ W \right]_n \right] P_n.
\]

**Proof:** Note that in Lemma 13, it was shown that \(B_n S_n^{III} \left[ W \right]_n = S_n^{IV} \). This, together with the facts \([S_n^{II}]^{T} = S_n^{III}\) and \([S_n^{IV}]^{T} = S_n^{IV}\), gives the result.

**Remark 18:** The derivation of the novel self-contained DST-II/III factorizations, respectively (26), (32), and (39) in this paper, are simpler than the existing interdependent factorizations for the DST-II/III matrices. Moreover, the proposed DST factorizations for the types III/III can be seen as a real variant of the self-contained factorization of the DFT matrices in [48], [49]. Also, the proposed DST-II/III factorizations are easy to implement and establish a bridge to the existing DST factorizations in [7], [9]–[12], [45]–[47]. We will see in Section V that the proposed algorithms are self-recursive as opposed to completely recursive algorithms in the literature. Moreover, the new algorithms require only simple, sparse, scaled orthogonal, diagonal, and bidiagonal matrices to compute the DST matrices by a vector as shown in the next section.

**V. SELF/COMPLETELY RECURSIVE AND RADIX-2 DST-I/II/III/IV ALGORITHMS**

The factorizations of DST I-IV matrices established in Section III lead to self/completely recursive and radix-2 algorithms. These algorithms can be used to compute DST matrices by a vector using simple, sparse, diagonal, bidiagonal,
and orthogonal matrices. In this section, we present low-complexity, recursive, and radix-2 DST I-IV algorithms. Similar to [12], we remove the scaling factor $\frac{1}{\sqrt{2}}$ from $H_n$, $\tilde{H}_{n-1}$, $V_n$, $S_n^H$, and $S_n^{HH}$ for further reduction of multiplication complexity. Compared to most of the existing algorithms, the proposed algorithms are easy to implement and attain the lowest multiplication complexity.

Before stating the algorithms, let us define the following notation to denote diagonal and bidiagonal matrices which will be used hereafter for $n \geq 4$.

$$B_n = \begin{bmatrix} B_{\bar{n}} & 0 \\ 0 & I_{\bar{n}-1} \end{bmatrix}, \quad \tilde{W}_{n-1} = \begin{bmatrix} \tilde{W}_{\bar{n}} & 0 \\ 0 & I_{\bar{n}-1} \end{bmatrix},$$

$$B_n' = \begin{bmatrix} B_{\bar{n}}' & 0 \\ 0 & I_{\bar{n}-1} \end{bmatrix}, \quad W_n' = \begin{bmatrix} W_{\bar{n}}' & 0 \\ 0 & I_{\bar{n}} \end{bmatrix}.$$ (64)

Also, from now onwards we call $\sqrt{2}H_n = \tilde{V}_n$. We recall from here the notations that $\sqrt{2}\tilde{H}_{n-1} = H_{n-1}$ and $\sqrt{2}H_n = H_n$.

Now we compute the scaled orthogonal DST I-IV matrices by a vector via $y = \sqrt{n}S_n^{H,x}$, $y = \sqrt{n}S_n^{H,x}$, $y = \sqrt{n}S_n^{H,x}$, and $y = \sqrt{n}S_n^{H,x}$, respectively. Here we call the corresponding DST I-IV algorithms $\sin 1(x, n - 1)$, $\sin 2(x, n)$, $\sin 3(x, n)$, and $\sin 4(x, n)$, respectively.

**Algorithm 19:** $\sin 1(x, n - 1)$

**Input:** $n = 2^t (t \geq 1)$, $n_1 = \frac{n}{2}$, $x \in \mathbb{R}^{n_1}$.

1) If $n = 2$, then
   $$y := \sqrt{2}x.$$

2) If $n \geq 4$, then
   $$u := H_{n-1}x,$$
   $$[v_j]_{j=0}^{n-2} := [\tilde{W}_{n-1}]_{j}u,$$
   $$z_1 := \sin 1([v_j]_{j=0}^{n-2}, n_1 - 1),$$
   $$z_2 := [v_{n-1}],$$
   $$z_3 := \sin 1([v_j]_{j=0}^{n-2}, n_1 - 1),$$
   $$w := \tilde{B}_{n-1}z_1, z_2, z_3 \right)^T,$$
   $$y := [P_{n-1}]^Tw.$$

**Output:** $y = \sqrt{n}S_n^{H,x}.$

**Algorithm 20:** $\sin 2(x, n)$

**Input:** $n = 2^t (t \geq 1)$, $n_1 = \frac{n}{2}$, $x \in \mathbb{R}^{n_1}$.

1) If $n = 2$, then
   $$y := \begin{bmatrix} 1 \\ 1 \\ 1 \\ -1 \end{bmatrix}x.$$

2) If $n \geq 4$, then
   $$u := H_nx,$$
   $$[v_j]_{j=0}^{n-1} := [W_n]_{j}u,$$
   $$z_1 := \sin 2([v_j]_{j=0}^{n-1}, n_1),$$
   $$z_2 := \sin 2([v_j]_{j=n_1}^{n-1}, n_1),$$
   $$w := B_nz_1, z_2 \right)^T,$$
   $$y := [P_n]w.$$

**Output:** $y = \sqrt{n}S_n^{H,x}.$

**Algorithm 21:** $\sin 3(x, n)$

**Input:** $n = 2^t (t \geq 1)$, $n_1 = \frac{n}{2}$, $x \in \mathbb{R}^{n_1}$.

1) If $n = 2$, then
   $$y := \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}x.$$

2) If $n \geq 4$, then
   $$u := [P_n]x,$$
   $$[v_j]_{j=0}^{n-1-1} := [B_n^H]_{j}u,$$
   $$z_1 := \sin 3([v_j]_{j=0}^{n-1-1}, n_1),$$
   $$z_2 := \sin 3([v_j]_{j=n_1}^{n-1-1}, n_1),$$
   $$w := [\tilde{W}_n]^Tz_1, z_2 \right)^T,$$
   $$y := [\tilde{H}_n^T]w.$$

**Output:** $y = \sqrt{n}S_n^{H,x}.$

**Algorithm 22:** $\sin 4(x, n)$

**Input:** $n = 2^t (t \geq 1)$, $n_1 = \frac{n}{2}$, $x \in \mathbb{R}^{n_1}$.

1) If $n = 2$, then
   $$y := \begin{bmatrix} \sqrt{2} \sin \frac{x}{8} \\ \sqrt{2} \cos \frac{x}{8} \end{bmatrix}.$$

2) If $n \geq 4$, then
   $$[a_j]_{j=0}^{n-1} := \tilde{Q}_n x,$$
   $$z_1 := \sin 2([a_j]_{j=0}^{n-1}, n_1),$$
   $$z_2 := \sin 2([a_j]_{j=n_1}^{n-1}, n_1),$$
   $$w := \left(\tilde{V}_n \right)(T_{c_1}, T_{c_2})^T,$$
   $$y := [P_n]w.$$

**Output:** $y = \sqrt{n}S_n^{H,x}.$

**VI. ARITHMETIC COMPLEXITY OF DST-I/II/III/IV ALGORITHMS**

The arithmetic complexity of an algorithm tells us the number of additions/subtractions and multiplications/divisions required to execute the algorithm. If we compute an order $n \times n$ DST matrix by a vector in the usual way, it requires $O(n^3)$ operations. Using the proposed DST algorithms, it is possible to compute the DST matrices by a vector using $O(n \log n)$ complexity with a significant reduction of multiplication cost.

Here we compute the number of additions (say $#a$) and multiplications (say $#m$) required to produce $y = \sqrt{n}S_n^{H,x}$ and $y = \sqrt{n}S_n^{H,x}$. Note that we do not count multiplication by $\pm 1$.

**A. LOW-COMPLEXITY DST-I/II/III/IV ALGORITHMS**

Here we analyze the arithmetic complexity of the self-recursive and radix-2 DST-I/II/III algorithms and the completely recursive and radix-2 DST-IV algorithm presented in Section V.

**Lemma 23:** For a given $n = 2^t (t \geq 2)$, the arithmetic complexity in computing the DST-I algorithm,
i.e., \( \sin 1(x, n - 1) \) is given by
\[
\#a(DST-I, n - 1) = 2nt - 4n + 4, \\
\#m(DST-I, n - 1) = \frac{1}{2}nt.
\] (65)

**Proof:** Referring to the DST-I algorithm, we get
\[
\#a(DST-I, n - 1) = \{ 2 \cdot \#a(DST-I, \frac{n}{2} - 1) \} + \#a(H^{s_1}_{n-1}) \\
+ \#a(\tilde{W}^{s}_{n-1}) + \#a(\tilde{B}^s_{n-1}).
\]
\[
\#m(DST-I, n - 1) = \{ 2 \cdot \#m(DST-I, \frac{n}{2} - 1) \} + \#m(H^{s_1}_{n-1}) \\
+ \#m(\tilde{W}^{s}_{n-1}) + \#m(\tilde{B}^s_{n-1}).
\]
(66)

Following the structures of \( H^{s_1}_{n-1} \), \( \tilde{W}^{s}_{n-1} \), and \( \tilde{B}^s_{n-1} \), we get
\[
\#a(H^{s_1}_{n-1}) = n - 2, \quad \#m(H^{s_1}_{n-1}) = 1, \\
\#a(\tilde{W}^{s}_{n-1}) = 0, \quad \#m(\tilde{W}^{s}_{n-1}) = n - 2, \\
\#a(\tilde{B}^s_{n-1}) = n - 2, \quad \#m(\tilde{B}^s_{n-1}) = 0.
\]

Using the above result, we can rewrite (66) as
\[
\#a(DST-I, n - 1) = 2 \cdot \#a(DST-I, \frac{n}{2} - 1) + 2n - 4. \\
\#m(DST-I, n - 1) = 2 \cdot \#m(DST-I, \frac{n}{2} - 1) + \frac{n}{2}.
\]

Since \( n = 2^t \), the above simplifies to the first order linear difference equations with respect to \( t \geq 2 \).
\[
\#a(DST-I, 2^t - 1) = 2 \cdot \#a(DST-I, 2^{t-1} - 1) = 2^{t+1} - 4. \\
\#m(DST-I, 2^t - 1) = 2 \cdot \#m(DST-I, 2^{t-1} - 1) = 2^{t-1}.
\]

Solving the above first order linear difference equations using the initial conditions \( \#a(DST-I, 1) = 0 \) and \( \#m(DST-I, 1) = 1 \), we can obtain
\[
\#a(DST-I, 2^t - 1) = 2nt - 4n + 4, \\
\#m(DST-I, 2^t - 1) = \frac{1}{2}nt.
\]
(67)

**Lemma 24:** For a given \( n = 2^t \) \( (t \geq 2) \), the arithmetic complexity in computing the DST-II algorithm \( \sin 2(x, n) \) is given by
\[
\#a(DST-II, n) = \frac{3}{2}nt - n + 1, \\
\#m(DST-II, n) = \frac{1}{2}nt - 1.
\]
(68)

**Proof:** Referring to the DST-II algorithm \( \sin 2(x, n) \), we get
\[
\#a(DST-II, n) = \{ 2 \cdot \#a(DST-II, \frac{n}{2}) \} + \#a(H^s_n) + \#a(W^s_n) \\
+ \#a(B^s_n),
\]
\[
\#m(DST-II, n) = \{ 2 \cdot \#m(DST-II, \frac{n}{2}) \} + \#m(H^s_n) + \#m(W^s_n) \\
+ \#m(B^s_n).
\]
(69)

Following the structures of \( H^s_n, W^s_n, \) and \( B^s_n \), we get
\[
\#a(H^s_n) = n, \quad \#m(H^s_n) = 0, \\
\#a(W^s_n) = 0, \quad \#m(W^s_n) = \frac{n}{2}, \\
\#a(B^s_n) = \frac{n}{2} - 1, \quad \#m(B^s_n) = 1.
\]

Using the above result, we can rewrite (68) as
\[
\#a(DST-II, n) = 2 \cdot \#a(DST-II, \frac{n}{2}) + \frac{3}{2}n - 1, \\
\#m(DST-II, n) = 2 \cdot \#m(DST-II, \frac{n}{2}) + \frac{n}{2} + 1.
\]

Since \( n = 2^t \), the above simplifies to the first order linear difference equations with respect to \( t \geq 2 \).
\[
\#a(DST-II, 2^t) = 2 \cdot \#a(DST-II, 2^{t-1}) = 3 \cdot 2^{t-1} - 1. \\
\#m(DST-II, 2^t) = 2 \cdot \#m(DST-II, 2^{t-1}) = 2^{t-1} + 1.
\]

Solving the above first order linear difference equations using the initial conditions \( \#a(DST-II, 2) = 2 \) and \( \#m(DST-II, 2) = 0 \), we can obtain
\[
\#a(DST-II, 2^t) = \frac{3}{2}nt - n + 1, \\
\#m(DST-II, 2^t) = \frac{1}{2}nt - 1.
\]
(69)

Hence the result.

**Corollary 25:** For a given \( n = 2^t \) \( (t \geq 2) \), the arithmetic complexity in computing the DST-III algorithm \( \sin 3(x, n) \) is given by
\[
\#a(DST-III, n) = \frac{3}{2}nt - n + 1, \\
\#m(DST-III, n) = \frac{1}{2}nt - 1.
\]
(69)

**Proof:** This is trivial as the factorization of the DST-III matrix is obtained using the factorization of the DST-II matrix with the help of the transpose property.

**Remark 26:** The arithmetic complexity of the proposed DST-II/III algorithms is the same as that of the lowest multiplication complexity DCT-II/III algorithms in [42]. This is because the DST-II/III and DCT-II/III matrices are related to each other via transpositions, permutations, and sign flips [50]–[53].

**Lemma 27:** For a given \( n = 2^t \) \( (t \geq 2) \), the arithmetic complexity in computing the DST-IV recursively via \( \sin 4(x, n) \) and \( \sin 2(x, n) \) algorithms is given by
\[
\#a(DST-IV, n) = \frac{3}{2}nt, \\
\#m(DST-IV, n) = \frac{1}{2}nt + n.
\]
(70)
Proof: The number of additions and multiplications required to compute the DST-IV algorithm can be found by using the number of additions and multiplications required to compute the DST-II algorithm at $\frac{n}{2}$.

By using the DST-IV algorithm, we get

$$\#a(DST-IV, n) = [2 \cdot \#a(DST-II, \frac{n}{2})] + \#a(\breve{Q}_n) + \#a(\breve{V}_n),$$

$$\#m(DST-IV, n) = [2 \cdot \#m(DST-II, \frac{n}{2})] + \#m(\breve{R}_n) + \#m(\breve{V}_n).$$

(71)

The structures of $\breve{Q}_n$ and $\breve{V}_n$ gives us

$$\#a(\breve{Q}_n) = \frac{3}{2} n,$$

$$\#m(\breve{Q}_n) = \frac{3}{2} n,$$

$$\#a(\breve{V}_n) = n - 2,$$

$$\#m(\breve{V}_n) = 2.$$  

(72)

Following the arithmetic complexity in computing DST-II algorithm in Lemma 24 and using the equations (72) and (71), we get

$$\#a(DST-IV, n) = 2 \left( \frac{3}{2} \cdot \frac{n}{2} \cdot (t - 1) - \frac{n}{2} + 1 \right) + \frac{5}{2} n - 2.$$  

(73)

$$\#m(DST-IV, n) = 2 \left( \frac{1}{2} \cdot \frac{n}{2} \cdot (t - 1) - 1 \right) + \frac{3}{2} n + 2.$$  

Simplifying the above yields the multiplication complexity of the modified DST-IV algorithm as in (70).

Remark 28: (a) The paper [54] presents the total lowest arithmetic operations in calculating DST-IV using the DFT and DCT-IV while leaving an open question: whether the DST-IV algorithm in [54] leads to practical gains in performance on real computers. To compute DST-IV, the authors of [54] used twiddle factors but these factors can be overlapped with other operations in the CPU [55]. Also to compute DST-IV, authors in [54] have used scaled factors of the form $\pm 1 \pm i \tan(2\pi k/n)$ and $\pm \cot(2\pi k/n) \pm i$, where

$$i^2 = -1$$

but these factors contain singular functions. Hence, the practical gains of the lowest flop count for DST-IV calculation in [54] may lead to a numerically ill-posed problem. We will show in Section VI-B, that the proposed DST-IV algorithm attains the lowest multiplication complexity algorithm to compute the DST-IV matrix by a vector. (b) The arithmetic complexity of the proposed DST-IV is the same as that of lowest multiplication complexity DCT-IV in [42] because DST-IV is exactly equivalent to a DCT-IV in which the outputs are reversed and every other input is multiplied by -1 (or vice-versa) [50]. [54].

### B. NUMERICAL ILLUSTRATIONS FOR THE COMPLEXITY OF DST-I/II/III/IV ALGORITHMS

In this section, we compare the arithmetic complexity of the proposed DST I-IV algorithms with the most existing DST algorithms in the literature. Tables 1 through 4 provide the number of additions and multiplications necessary to compute the DST I-IV algorithms for input sizes ranging from $n = 8$ to $n = 4096$.

Although the total flop count of the DST-I algorithm in [47] is lower than all other DST-I algorithms in [7], [10], [12] and the proposed algorithm, the $\sin 1(x, n)$ algorithm is presented using simple, sparse, diagonal, and scaled orthogonal matrices as opposed to the DST-I factorization in [47].

Based on the counts in computing DST-II/III algorithms, the proposed DST-II/III algorithms have the lowest multiplication counts. We recall here that the authors in [53] calculated the lowest flop count split-radix DST-II/III algorithms. By considering the fact that the authors in [53] have reduced the flop count only by cutting off multiplication counts while preserving the same addition counts as in [56], the explicit multiplication count in computing split-radix DST-II/III algorithms in [53] is given via $(7/18)n + (10/27)n - (1/9)(-1)^t + (7/54)(-1)^t + 1/2$. In this situation, the split-radix DST-II/III algorithms attain the lowest multiplication counts for $n \geq 32$ but without using simple,

| n     | $\sin 1(x, n)$ | [12] | [47] | [10] | [7]  | [57] |
|-------|----------------|------|------|------|------|------|
|       | #a | #m | #a | #m | #a | #m | #a | #m | #a | #m | #a | #m |
| 8     | 20 | 12 | 18 | 9  | 27 | 5  | 18 | 14 | 19 | 5  | 20 | 5  |
| 16    | 68 | 32 | 58 | 26 | 72 | 17 | 58 | 43 | 62 | 17 | 68 | 17 |
| 32    | 196| 80 | 160| 73 | 185| 49 | 160| 118| 175| 51 | 196| 49 |
| 64    | 516| 192| 408| 186| 458| 129| 408| 299| 456| 141| 516| 129|
| 128   | 1284| 448| 990| 457| 1099| 321| 990| 726| 1129| 367| 1284| 321|
| 256   | 3076| 1024| 2326| 1082| 2572| 769| 2326| 1707| 2698| 913| 3076| 769|
| 512   | 7172| 2304| 5340| 2505| 5901| 1793| 5340| 3926| 6283| 2195| 7172| 1793|
| 1024  | 16388| 5120| 12052| 5690| 13326| 4097| 12052| 8875| 14348| 5141| 16388| 4097|
| 2048  | 36868| 11264| 26842| 12745| 29711| 9217| 26842| 19798| 32269| 11799| 36868| 9217|
| 4096  | 81924| 24576| 59145| 28128| 65552| 20481| 59154| 43691| 71694| 26649| 81924| 20481|

TABLE 1. Number of additions and multiplications required to compute DST-I algorithms.
### TABLE 2. Number of additions and multiplications required to compute DST-II algorithms.

| n   | $\sin^2(x, n)$ | [12] | [47], [52], [57]–[59] | [7] | [56] |
|-----|----------------|------|-------------------------|-----|------|
|     | #a  | #m  | #a  | #m  | #a  | #m  | #a  | #m  | #a  | #m  |
| 8   | 29  | 11  | 26  | 16  | 29  | 12  | 29  | 13  | 29  | 13  |
| 16  | 81  | 31  | 72  | 46  | 81  | 32  | 83  | 35  | 81  | 33  |
| 32  | 209 | 79  | 186 | 112 | 209 | 80  | 219 | 91  | 209 | 81  |
| 64  | 513 | 191 | 456 | 270 | 513 | 192 | 547 | 227 | 513 | 193 |
| 128 | 1217| 447 | 1082| 624 | 1217| 448 | 1315| 547 | 1217| 449 |
| 256 | 2817| 1023| 2504| 1422| 2817| 1024| 3075| 1283| 2817| 1025|
| 512 | 6401| 2303| 5690| 3184| 6401| 2304| 7043| 2947| 6401| 2305|
| 1024| 14337| 5119| 12744| 7054| 14337| 5120| 15875| 6659| 14337| 5121|
| 2048| 31745| 11263| 28218| 15472| 31745| 11264| 35331| 14851| 31745| 11265|
| 4096| 69633| 24575| 61896| 33678| 69633| 24576| 77827| 32771| 69633| 24577|

### TABLE 3. Number of additions and multiplications required to compute DST-III algorithms.

| n   | $\sin^3(x, n)$ | [12] | [47], [52], [58], [59] | [57] | [7] | [56] |
|-----|----------------|------|-------------------------|-----|-----|------|
|     | #a  | #m  | #a  | #m  | #a  | #m  | #a  | #m  | #a  | #m  |
| 8   | 29  | 11  | 26  | 16  | 29  | 12  | 29  | 13  | 29  | 13  |
| 16  | 81  | 31  | 72  | 46  | 81  | 32  | 83  | 35  | 81  | 33  |
| 32  | 209 | 79  | 186 | 112 | 209 | 80  | 219 | 91  | 209 | 81  |
| 64  | 513 | 191 | 456 | 270 | 513 | 192 | 547 | 227 | 513 | 193 |
| 128 | 1217| 447 | 1082| 624 | 1217| 448 | 1315| 547 | 1217| 449 |
| 256 | 2817| 1023| 2504| 1422| 2817| 1024| 3075| 1283| 2817| 1025|
| 512 | 6401| 2303| 5690| 3184| 6401| 2304| 7043| 2947| 6401| 2305|
| 1024| 14337| 5119| 12744| 7054| 14337| 5120| 15875| 6659| 14337| 5121|
| 2048| 31745| 11263| 28218| 15472| 31745| 11264| 35331| 14851| 31745| 11265|
| 4096| 69633| 24575| 61896| 33678| 69633| 24576| 77827| 32771| 69633| 24577|

### TABLE 4. Number of additions and multiplications required to compute DST-IV algorithms.

| n   | $\sin^4(x, n)$ | [12] | [47] | [60] | [7] | [56] | [57] |
|-----|----------------|------|------|------|-----|------|------|
|     | #a  | #m  | #a  | #m  | #a  | #m  | #a  | #m  | #a  | #m  |
| 8   | 36  | 20  | 30  | 30  | 36  | 20  | -   | -   | 38  | 22  |
| 16  | 96  | 48  | 82  | 66  | 96  | 48  | -   | -   | 104 | 56  |
| 32  | 240 | 112 | 206 | 158 | 240 | 112 | -   | -   | 264 | 136 |
| 64  | 576 | 256 | 498 | 354 | 576 | 256 | 576 | 274 | 640 | 320 |
| 128 | 1344| 576 | 1166| 798 | 1344| 576 | 2176| 1058| 1504| 736 |
| 256 | 3072| 1280| 2674| 1762| 3072| 1280| 8448| 4162| 3456| 1664|
| 512 | 6912| 2816| 6030| 3870| 6912| 2816| -   | -   | 7808| 3712|
| 1024| 15360| 6144| 13426| 8418| 15360| 6144| 131584| 65794| 17408| 8192|
| 2048| 33792| 13312| 29582| 18206| 33792| 13312| -   | -   | 38400| 17920|
| 4096| 73728| 28672| 64626| 39138| 73728| 28672| -   | -   | 83968| 38912|

VOLUME 9, 2021
sparse, diagonal, bidiagonal, and scaled orthogonal matrices. Furthermore, the split-radix DST-II/III algorithms leave a question based on the practical gains in performance on real computers as the algorithms in [53] use singular functions and hence may lead to ill-posed problems.

By considering the fact that the authors in [53] have reduced the flop count in computing DST-IV matrix by a vector only by cutting off multiplication counts, the explicit multiplication count in computing the split-radix DST-IV algorithm in [53] is given via \( \frac{5}{9}nt + \left(\frac{37}{27}\right)n + \left(\frac{2}{9}\right)(-1)^{t}t - \left(\frac{10}{27}\right)(-1)^{t} - 2 \). Even though the total flop count of the split-radix DST-IV algorithm in [54] is lower than all the other DST-IV algorithms in the literature, the multiplication counts of the proposed \( \sin^4(x, n) \) algorithm and the DST-IV algorithm in [47] is the lowest in the literature. Moreover, the proposed DST-IV algorithm is presented using simple, sparse, diagonal, bidiagonal, and scaled orthogonal matrices as opposed to the DST-IV factorization in [47] and any other existing DST-IV algorithms in the literature. Note that if the factorization of the DCT or DST does not preserve orthogonality, the resulting DCT or DST algorithms lead to inferior numerical stability [61].

To sum up, the proposed self/completely recursive and radix-2 \( \sin^2(x, n) \), \( \sin^3(x, n) \), and \( \sin^4(x, n) \) algorithms can be computed via simple, sparse, diagonal, bidimensional, and scaled orthogonal matrices having the lowest multiplication counts in the literature.

VII. SIGNAL FLOW GRAPHS FOR DST-I/II/III/IV ALGORITHMS

Signal flow graphs can be used to represent a physical system and its controllers with the help of a system of linear equations. We present signal flow graphs of the proposed DST algorithms while illustrating the connection between system components, associated equations, and the simplicity of the proposed factorizations of the DST I-IV matrices. The signal flow graphs are drawn using decimation-in-frequency. It is possible to convert these signal flow graphs into decimation-in-time flow graphs.

For a given input signal \( x \), we present signal flow graphs for the output signal \( y = \sqrt{nS_{n-1}^I} x \), \( y = \sqrt{nS_{n}^II} x \), \( y = \sqrt{nS_{n}^III} x \), and \( y = \sqrt{nS_{n}^{IV}} x \). Fig. 1, Fig. 2, Fig. 3 and Fig. 4 present the signal flow graphs for the \( \sin 1(x, n - 1) \), \( \sin 2(x, n) \), \( \sin 3(x, n) \), and \( \sin 4(x, n) \) algorithms, respectively. The notation in the figures are defined as \( \epsilon := \sqrt{2} \), \( \bar{\omega}_{j,k} := \frac{1}{2} \csc \left(\frac{(k-2)\pi}{2}\right) \), \( w_{j,k} := \frac{1}{2} \csc \left(\frac{(k-2)+1\pi}{2}\right) \), \( t_{j,k} = \tan \left(\frac{(j+1)\pi}{8k}\right) \), \( s_{j,k} = \sin \frac{\pi}{8k} \), and the dotted lines indicate a multiplication by factor -1.

VIII. ENCRYPTION USING DST ALGORITHMS

In [62], a 4f encryption scheme was proposed which introduces a chaotic mapping to the established double random phase encoding (DPRE) scheme originally proposed in [63]–[65]. The DPRE process applies to a phase image a random phase mask followed by a Fourier transform twice in series. While the DPRE scheme maximizes the entropy of the encrypted image, the linearity of the scheme creates vulnerabilities [66], [67]. Thus, [62] proposed an encryption scheme that applies a chaotic pixel mapping based on the famous 3D Lorenz System after the first Fourier transform.
The Lorenz system, also commonly known as the Lorenz path or Lorenz attractor, was first described in [68]. The path itself is the solution to the set of ordinary differential equations

\[
\begin{align*}
\frac{dx}{dt} &= \sigma(y - x), \\
\frac{dy}{dt} &= \rho x - y - xz, \\
\frac{dz}{dt} &= xy - \beta z,
\end{align*}
\]

where \( \sigma, \rho, \) and \( \beta \) are system parameters and the initial conditions are \( x(0) = x_0, y(0) = y_0, z(0) = z_0 \). Given time to diverge, this set of equations will produce completely different paths with even the slightest difference in the system parameters. This chaotic nature, along with the relative speed by which the set of equations may be solved, makes the Lorenz system a valuable tool for mapping and permutation techniques.

In the following, we describe image encryption based on the DPRF encryption technique in [62] and the proposed DST algorithms.

1) An image with pixel values \( f(x, y) \), where \( x \) and \( y \) are locations in the spatial domain, is converted into a phase image \( p_0(x, y) = e^{i\pi f(x, y)} \).

2) A random phase mask is applied to each pixel as defined by \( p_1(x, y) = p_0(x, y)e^{2\pi ir(x, y)} \), where \( r \) is a random matrix having the same size as the original image.

3) The image is transformed into the Fourier domain using the 2D DFT and the proposed \textbf{sin2} algorithm in 2D.

4) A chaotic mapping technique is applied based on the 3D Lorenz system. The Lorenz system is solved using Matlab’s \textit{ode45} function. First, the image is decomposed into \( n \times n \) blocks, where \( n \) is a multiple of the total length of the image. The red, green, and blue color channels of each block are associated with the position vectors of the Lorenz path in time. These blocks are then sorted in ascending, row-major order based on the value of the associated Lorenz system position. This produces a new image \( p_L \).

5) A random phase mask is applied to each pixel as defined by \( p_2(u, v) = p_L(u, v)e^{2\pi ir(u, v)} \), where \( s \) is a random matrix of the same size as the original image, \( u \) and \( v \) are positions in the Fourier domain.

6) The proposed \textbf{sin2} algorithm and the DFT are applied to \( p_2 \) to produce the fully encrypted image.

The final result is an image of high entropy with a Gaussian distribution of pixel values across all channels. To decrypt the image, the process is simply performed in reverse using the same random phase mask keys, Lorenz system parameters, and the inverse of the respective transform. For the decryption of the image, the proposed 2D \textbf{sin3} algorithm and the 2D inverse-DFT are used. For a block length \( n \) less than or equal to \( \frac{1}{8} \) of the total image length (i.e. \( n = 32 \) for a \( 256 \times 256 \) image), attempts to decrypt the image using Lorenz system initial conditions with even the most minute difference will fail. Blocks which are \( \frac{1}{8} \) the total image length (i.e. \( n = 64 \) for a \( 256 \times 256 \) image) will reveal some information if transformed into the Fourier domain using the DFT; however, we will show below that images transformed using the proposed DST algorithms will withstand these attacks. Figure 5 shows the original image which was used for testing purposes and four attempts to decrypt using an incorrect key for the Lorenz path. The corresponding image histograms are also displayed. The Lorenz parameter key used to encrypt the image is

\[
[\sigma = 10, \beta = \frac{8}{3}, \rho = 28]
\]  

and the decryption attack is simulated using the key

\[
[\sigma = 10.000000000000001, \beta = \frac{8}{3}, \rho = 28]
\]  

As shown in Figures 5.c through 5.f, the encryption scheme remains secure for small block sizes when transformed into the Fourier domain through either the proposed \textbf{sin2} algorithm or the DFT. This remains true for the encryption scheme.
FIGURE 5. Comparison of encryption performance using the proposed DST-II/III algorithms and the DFT.

IX. CONCLUSION

We have proposed novel factorization formulas to compute DST I-IV matrices using the product of simple, sparse, diagonal, bidiagonal, and scaled orthogonal matrices. The proposed DST matrix factorizations are used to establish self/completely recursive and radix-2 DST algorithms. We have also established a novel algebraic relationship between DST-II and DST-IV matrices using diagonal and bidiagonal matrices and described connections between traditional DST-II/III algorithms and the proposed DST-II/III algorithms. Moreover, we have established another novel algebraic relationship between DST-I and DST-III matrices using sparse and diagonal matrices. We have also described connections between traditional DST-I algorithms and the proposed DST-I algorithm. The arithmetic complexity of the proposed algorithms is analyzed to show that the proposed algorithms are much more efficient than the existing algorithms. Specifically, the DST-IV algorithm attains the lowest multiplication complexity for all transform matrix sizes \( n \geq 8 \) in the literature. The DST-II/III algorithms are the lowest multiplication complexity, radix-2, and self-recursive algorithms and execute using simple, sparse, diagonal, bidiagonal, and scaled orthogonal matrices for the transform matrix sizes \( n \geq 8 \). The presented signal flow graphs illustrate the simplicity of the proposed DST algorithms and provide a foundational framework for the DST-based integrated circuits. The robust performance of the proposed algorithms within software applications is explored through implementation within a DPRE encryption scheme with a chaotic Lorenz path mapping. The proposed DST algorithms in 2D can increase encryption performance when compared to an implementation using the 2D DFT.

REFERENCES

[1] A. K. Jain, “A sinusoidal family of unitary transforms,” IEEE Trans. Pattern Anal. Mach. Intell., vol. PAMI-1, no. 4, pp. 356–365, Oct. 1979.
[2] A. Jain, “A fast Karhunen-Loeve transform for a class of random processes,” IEEE Trans. Commun., vol. C-24, no. 9, pp. 1023–1029, Sep. 1976.
[3] H. B. Kekre and J. K. Solanki, “Comparative performance of various trigonometric unitary transforms for transform image coding,” Int. J. Electron., vol. 44, no. 3, pp. 305–315, Mar. 1978.
[4] Z. Wang and B. R. Hunt, “The discrete W transform,” Appl. Math. Comput., vol. 16, no. 1, pp. 19–48, Jan. 1985.
[5] P. Yip and K. Rao, “A fast computational algorithm for the discrete sine transform,” IEEE Trans. Commun., vol. C-28, no. 2, pp. 304–307, Feb. 1980.
[6] W.-H. Chen, C. Smith, and S. Fralick, “A fast computational algorithm for the discrete cosine transform,” IEEE Trans. Commun., vol. C-25, no. 9, pp. 1004–1009, Sep. 1977.
[7] Z. Wang, “Fast algorithms for the discrete \( W \) transform and for the discrete Fourier transform,” IEEE Trans. Acoust., Speech, Signal Process., vol. ASSP-32, no. 4, pp. 803–816, Aug. 1984.
X. Shao and S. G. Johnson, “Type-IV DCT, DST, and MDCT algorithms with reduced numbers of arithmetic operations,” *Signal Process.*, vol. 88, no. 6, pp. 1313–1326, Jun. 2008.

S. G. Johnson and M. Frigo, “A modified split-radix FFT with fewer arithmetic operations,” *IEEE Trans. Signal Process.*, vol. 55, no. 1, pp. 111–119, Jan. 2007.

Z. Wang, “On computing the Fourier and cosine transforms,” *IEEE Trans. Acoust. Speech Signal Process.*, vol. ASSP-33, no. 5, pp. 1341–1344, 1985.

P. Yip and K. R. Rao, “Fast decimation-in-time algorithms for a family of discrete sine and cosine transforms,” *Circuits, Syst., Signal Process.*, vol. 3, no. 4, pp. 387–408, Dec. 1984.

O. Ersoy and N. Hu, “A unified approach to the fast computation of all discrete trigonometric transforms,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Apr. 1987, pp. 1843–1846.

Z. Cvetkovic and M. V. Popovic, “New fast recursive algorithms for the computation of discrete cosine and sine transforms,” *IEEE Trans. Signal Process.*, vol. 40, no. 8, pp. 2083–2086, Aug. 1992.

V. Britanak, “New universal rotation-based fast computational structures for an efficient implementation of the DCT-IV/DST-IV and analysis/synthesis MDCT/MDST filter banks,” *Signal Process.*, vol. 89, no. 11, pp. 2213–2232, Nov. 2009.

M. Tasche and H. Zeuner, *Handbook of Analytic-Computational Methods in Applied Mathematics*. Boca Raton, FL, USA: CRC Press, 2000, pp. 357–406.

N. Sharma, I. Saini, A. Yadav, and P. Singh, “Phase-image encryption based on 3D-Lorenz chaotic system and double random phase encoding,” *3D Res.*, vol. 8, no. 4, p. 39, Dec. 2017.

B. Javadi, *Optical and Digital Techniques for Information Security*. Berlin, Germany: Springer, 2006.

B. Javadi, A. Carnicer, M. Yamaguchi, T. Nomura, E. Pérez-Cabrér, M. S. Millàn, N. K. Nishchal, R. Torroba, J. F. Barrera, W. He, and X. Peng, “Roadmap on optical security,” *J. Opt.*, vol. 18, no. 8, 2016, Art. no. 083001.

P. Refregier and B. Javidi, “Optical image encryption based on input plane and Fourier plane random encoding,” *Opt. Lett.*, vol. 20, no. 7, pp. 767–769, Apr. 1995.

P. Kumar, J. Joseph, and K. Singh, “Known-plaintext attack-free double random phase-amplitude optical encryption: Vulnerability to impulse function attack,” *J. Opt.*, vol. 14, no. 4, Apr. 2012, Art. no. 045401.

P. Kumar, J. Joseph, and K. Singh, “Double random phase encoding based optical encryption systems using some linear canonical transforms: Weaknesses and countermeasures,” in *Linear Canonical Transforms*, J. J. Healy, Ed. New York, NY, USA: Springer, 2016, p. 367.

E. N. Lorenz, “Deterministic nonperiodic flow,” *J. Atmos. Sci.*, vol. 20, no. 2, pp. 130–141, 1963.