COOLEY-TUKEY FFT LIKE ALGORITHM FOR THE DISCRETE TRIANGLE TRANSFORM

Markus Püschel

Dept. of Electrical and Computer Engineering
Carnegie Mellon University, Pittsburgh, U.S.A.

Martin Rötteler

Dept. of Combinatorics and Optimization
University of Waterloo, Waterloo, Canada

ABSTRACT

The discrete triangle transform (DTT) was recently introduced [1] as an example of a non-separable transform for signal processing on a two-dimensional triangular grid. The DTT is built from Chebyshev polynomials in two variables in the same way as the DCT, type III, is built from Chebyshev polynomials in one variable. We show that, as a consequence, the DTT has, as the DCT, type III, a Cooley-Tukey FFT type fast algorithm. We derive this algorithm and an upper bound for the number of complex operations it requires. Similar to most separable two-dimensional transforms, the operations count of this algorithm is $O(n^2 \log(n))$ for an input of size $n \times n$.

1. INTRODUCTION

Two-dimensional signal processing is mainly based on separable transforms, which are Kronecker or tensor products of their one-dimensional counterparts. The $n \times n$ input to such a transform is assumed to be located on a square grid with boundary conditions depending on the transform used. For example, the two-dimensional DFT imposes periodic boundary conditions and the grid thus becomes a torus.

In [1] we have introduced the *discrete triangle transform (DTT)*, a *non-separable* two-dimensional signal transform for signal processing on a triangular grid (see Fig. 1, the boundary conditions are omitted, see [1]). The DTT is derived from Chebyshev polynomials in two variables in the same way as the DCT, type III, is derived from Chebyshev polynomials in one variable [2].

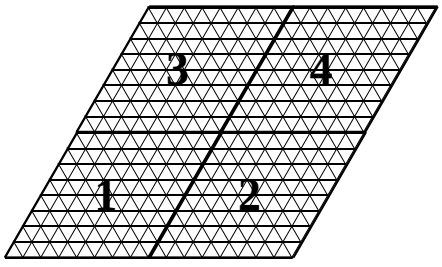

Fig. 1. The triangular grid corresponding to the DTT. The division into four sectors is used for the algorithm derivation.

In this paper, we derive a fast algorithm for the DTT that requires $O(n^2 \log(n))$ arithmetic operations. The algorithm is, in a strict mathematical sense, analogous to the Cooley-Tukey FFT or its analogue for the DCT [3], namely derived by a stepwise decomposition of the *polynomial algebra* associated to the transform.

The work of Markus Püschel was supported by NSF award 0310941; the work of Martin Rötteler was supported by CFI, ORDCF, and MITACS.

2. BACKGROUND: POLYNOMIAL ALGEBRAS

Many fast transform algorithms can be readily derived by identifying the polynomial algebra associated to a transform [2, 3]. We provide the necessary background in this section.

Polynomial Algebras. An *algebra* $\mathcal{A}$ is a vector space that is also a ring, i.e., it permits multiplication of elements. Examples include $\mathbb{C}$ and the sets $\mathbb{C}[x]$, $\mathbb{C}[x, y]$ of polynomials in one or two variables, respectively. Let $p(x)$ be a polynomial of degree n, then the set of polynomials of degree less than n is an algebra w.r.t. multiplication modulo p. We denote this algebra by $\mathbb{C}[x]/p(x)$. Similarly, we can consider two polynomials $p(x, y)$, $q(x, y)$ in two variables and construct the polynomial algebra

$$\mathcal{A} = \mathbb{C}[x, y]/\langle p(x, y), q(x, y)\rangle \tag{1}$$

with ordinary addition and multiplication modulo p and q.

Polynomial Transforms. Let $\mathcal{A} = \mathbb{C}[x]/p(x)$ be a polynomial algebra, let $\alpha = (\alpha_0, \ldots, \alpha_{n-1})$ be the zeros of p assumed to be pairwise distinct. Further, let $b = (p_0, \ldots, p_{n-1})$ be a basis of $\mathbb{C}[x]/p(x)$. Then, by the Chinese Remainder Theorem (CRT), $\mathcal{A}$ decomposes as

$$C[x]/p(x) \cong \bigoplus_{0 \le k < n} \mathbb{C}[x]/(x - \alpha_k) \tag{2}$$

and the corresponding decomposition matrix, built from the projections $p_\ell(x) \equiv p_\ell(\alpha_k) \bmod (x - \alpha_k)$, is the *polynomial transform*

$$\mathcal{P}_{b,\alpha} = [p_\ell(\alpha_k)]_{0 \le k, \ell < n}. \tag{3}$$

An example is the DFT_n, which is a polynomial transform for $\mathcal{A} = \mathbb{C}[x]/(x^n - 1)$ with basis $p_\ell = x^\ell$. Further, all DCTs and DSTs are polynomial transforms [2].

We can also define a polynomial transform for the algebra (1) as follows. We assume that the common zeros of $p(x, y)$ and $q(x, y)$ are pairwise distinct[1] and given by the pairs (α_k, β_k), $0 \le k < n^2$. Again we invoke the Chinese Remainder Theorem to obtain the decomposition

$$\mathcal{A} \to \bigoplus_{0 \le k < n^2} \mathbb{C}[x, y]/\langle x - \alpha_k, y - \beta_k\rangle$$

into one-dimensional algebras, which implies that $\mathcal{A}$ has dimension n^2. With respect to a chosen basis $(p_\ell(x, y) | 0 \le \ell < n^2)$ of $\mathcal{A}$, the corresponding matrix of projections

$$[p_\ell(\alpha_k, \beta_k)]_{0 \le k, \ell < n^2}$$

is a generalization of (3) and is an example of a polynomial transform in two variables.

[1] Note that for "generic" polynomials this is true, which follows from Bézout's theorem [4].

Cooley-Tukey type algorithms. A fast algorithm for a polynomial transform $\mathcal{P}_{b,\alpha}$ for $\mathcal{A} = \mathbb{C}[x]/p(x)$ can be constructed if the polynomial p decomposes as $p(x) = q(r(x))$. In this case $\mathcal{A}$ decomposes in steps as follows. We assume $\deg(q) = k$, $\deg(r) = m$, which implies $n = km$, and denote with $\beta = (\beta_0, \ldots, \beta_{k-1})$ the zeros of the outer polynomial q, and with $\alpha'_i = (\alpha_{i,0}, \ldots, \alpha_{i,m-1})$ the zeros of $r(x) - \beta_i$. Then each $\alpha_{i,j}$ is a zero α_ℓ of p. We obtain the decomposition

$$\mathbb{C}[x]/p(x) \quad \to \quad \bigoplus_{0 \le i < k} \mathbb{C}[x]/(r(x) - \beta_i) \tag{4}$$

$$\to \quad \bigoplus_{0 \le i < k} \bigoplus_{0 \le j < m} \mathbb{C}[x]/(x - \alpha_{i,j}) \tag{5}$$

$$\to \quad \bigoplus_{0 \le \ell < n} \mathbb{C}[x]/(x - \alpha_\ell). \tag{6}$$

This decomposition leads to a recursive factorization of the associated polynomial transform into a product of *four* sparse matrices. The first arises from a base change B in $\mathbb{C}[x]/p(x)$, not explained due to lack of space (see [3]). The other three factors correspond to steps (4)–(6). Steps (4) and (5) are applications of the CRT, step (6) is a permutation P of the summands. The result has the form

$$\mathcal{P}_{b,\alpha} = P(\bigoplus_i \mathcal{P}_i)(\mathcal{P}' \otimes \mathrm{I}_m)B, \tag{7}$$

where $\mathcal{P}'$ is a transform for $\mathbb{C}[x]/q(x)$ and the $\mathcal{P}_i$ are transforms for $\mathbb{C}[x]/(r(x) - \beta_i)$. This method yields the Cooley-Tukey FFT if applied to the decomposition $p(x) = x^n - 1 = (x^m)^k - 1$, for $n = km$, and also leads to fast Cooley-Tukey type algorithms for the DCTs of type II and III [3], which are based on the decomposition $T_n(x) = T_k(T_m(x))$, where T_n is a Chebyshev polynomial in one variable.

3. THE DISCRETE TRIANGLE TRANSFORM

In this section we define the discrete triangle transform and give its polynomial algebra, which is a special case of (1). See [1] for full details. We use the notation $\overline{p}(x, y) = p(y, x)$ for the polynomial obtained from $p(x, y) \in \mathbb{C}[x, y]$ by reversing the arguments.

Chebyshev polynomials (two variables). The Chebyshev polynomials $T_n(x, y)$ in two variables (e.g., [5, 6]) are defined by the recursion

$$T_n = 3xT_{n-1} - 3yT_{n-2} + T_{n-3}.$$

The initial conditions are given by $T_{-1} = y$, $T_0 = 1$, $T_1 = x$. Using this recursion in both directions we define T_n for all $n \in \mathbb{Z}$. A full two-dimensional family of Chebyshev polynomials $T_{n,m}$, for $n, m \in \mathbb{Z}$, is obtained by defining $T_{n,0} = T_n, T_{0,n} = \overline{T}_n$ and

$$T_{n,m} = (3T_n\overline{T}_m - T_{n-m})/2. \tag{8}$$

The following main properties can be derived from this definition.

$$T_{n,-m} = T_{n-m,m}, \quad T_{-n,m} = T_{n,m-n} \tag{9}$$

$$T_k T_{n,m} = \tfrac{1}{3}(T_{n-k,m+k} + T_{n,m-k} + T_{n+k,m}) \tag{10}$$

$$T_{km} = T_k(T_m, \overline{T}_m),\ \overline{T}_{km} = \overline{T}_k(T_m, \overline{T}_m). \tag{11}$$

Property (10) defines the triangular structure of the grid, and (11) is the decomposition property [7] we use for deriving the algorithm given below. The following parameterization of $T_{k,\ell}$ (see [5]) is useful:

$$T_{k,\ell}(x, y) = \tfrac{1}{6}(u^k v^{-\ell} + u^{-\ell} v^k + u^{k+\ell} v^\ell + u^\ell v^{k+\ell} + u^{-k-\ell} v^{-k} + u^{-k} v^{-k-\ell}), \tag{12}$$

where $x = \frac{1}{3}(u + v + (uv)^{-1})$ and $y = \frac{1}{3}(u^{-1} + v^{-1} + uv)$.

The equations $T_n(x, y) = \overline{T}_n(x, y) = 0$ have precisely n^2 common zeros over the complex numbers. It is convenient to represent these zeros in the parameterization (12), in which they are given by all pairs

$$(u_i, v_j) = (\omega_n^i, \omega_{3n}^{1+3j}), \quad 0 \le i, j < n, \quad \omega_n = e^{-2\pi\sqrt{-1}/n}.$$

The discrete triangle transform. We define the discrete triangle transform $\mathrm{DTT}_{n \times n}$ for input size $n \times n$ as the polynomial transform for the polynomial algebra $\mathbb{C}[x, y]/\langle T_n, \overline{T}_n \rangle$ with basis $b_n = (T_{k,\ell} \mid 0 \le k, \ell < n)$, and list of zeros $\alpha = ((\alpha_{i,j}, \beta_{i,j}) \mid 0 \le i, j < n)$, where $\alpha_{i,j}$ and $\beta_{i,j}$ are determined by $(u_i, v_j) = (\omega_n^i, \omega_{3n}^{1+3j})$ in the parameterization (12). In other words, $\mathrm{DTT}_{n \times n}$ is the $n^2 \times n^2$ matrix

$$\mathrm{DTT}_{n \times n} = [T_{k,\ell}(\alpha_{i,j}, \beta_{i,j})]_{0 \le i,j < n,\ 0 \le k,\ell < n}. \tag{13}$$

The double index (i, j) is the row index, and (k, ℓ) is the column index, both ordered lexicographically. To compute the matrix entries, the polynomials $T_{k,\ell}$ can be evaluated at the zeros $(u_i, v_j) = (\omega_n^i, \omega_{3n}^{1+3j})$ by using the parameterization (12).

We consider the example $n = 2$. Here we have to evaluate the polynomials $T_{0,0} = 1, T_{0,1} = y, T_{1,0} = x, T_{1,1} = \frac{1}{2}(3xy - 1)$ at the zeros of the equation $T_{2,0} = T_{0,2} = 0$. The solutions are given by the four points

$$(\tfrac{2}{3}, \tfrac{2}{3}), (0, 0), (\tfrac{2}{3}\omega_3, \tfrac{2}{3}\omega_3^2), (\tfrac{2}{3}\omega_3^2, \tfrac{2}{3}\omega_3). \tag{14}$$

Hence $\mathrm{DTT}_{2 \times 2}$ is the 4×4 matrix

$$\mathrm{DTT}_{2 \times 2} = \begin{bmatrix} 1 & \frac{2}{3} & \frac{2}{3} & \frac{1}{6} \\ 1 & 0 & 0 & -\frac{1}{2} \\ 1 & \frac{2}{3}\omega_3^2 & \frac{2}{3}\omega_3 & \frac{1}{6} \\ 1 & \frac{2}{3}\omega_3 & \frac{2}{3}\omega_3^2 & \frac{1}{6} \end{bmatrix}. \tag{15}$$

4. COOLEY-TUKEY TYPE ALGORITHM FOR THE DTT

Based on the decomposition property (11) of Chebyshev polynomials in two variables, we derive a fast algorithm for the discrete triangle transform $\mathrm{DTT}_{n \times n}$, where we assume $n = 2m$. The algorithm arises from the following decomposition of the algebra.

$$\begin{aligned}
& \mathbb{C}[x, y]/\langle T_2(T_m, \overline{T}_m), \overline{T}_2(T_m, \overline{T}_m) \rangle \\
\to \quad & \mathbb{C}[x, y]/\langle T_m - \tfrac{2}{3}, \overline{T}_m - \tfrac{2}{3} \rangle \\
& \oplus\, \mathbb{C}[x, y]/\langle T_m, \overline{T}_m \rangle \\
& \oplus\, \mathbb{C}[x, y]/\langle T_m - \tfrac{2}{3}\omega_3, \overline{T}_m - \tfrac{2}{3}\omega_3^2 \rangle \\
& \oplus\, \mathbb{C}[x, y]/\langle T_m - \tfrac{2}{3}\omega_3^2, \overline{T}_m - \tfrac{2}{3}\omega_3 \rangle \\
\to \quad & \bigoplus_{0 \le i,j < m} \mathbb{C}[x, y]/\langle x - \alpha_{2i,2j}, y - \beta_{2i,2j} \rangle \\
& \oplus \bigoplus_{0 \le i,j < m} \mathbb{C}[x, y]/\langle x - \alpha_{2i,2j+1}, y - \beta_{2i,2j+1} \rangle \\
& \oplus \bigoplus_{0 \le i,j < m} \mathbb{C}[x, y]/\langle x - \alpha_{2i+1,2j}, y - \beta_{2i+1,2j} \rangle \\
& \oplus \bigoplus_{0 \le i,j < m} \mathbb{C}[x, y]/\langle x - \alpha_{2i+1,2j+1}, y - \beta_{2i+1,2j+1} \rangle \\
\to \quad & \bigoplus_{0 \le i,j < m} \mathbb{C}[x, y]/\langle x - \alpha_{i,j}, y - \beta_{i,j} \rangle
\end{aligned}$$

As in the one-variable case, we first (not shown above) perform a base change B in $\mathcal{A}$ from b_n to $b'_n = T_{0,0} \cdot b_m \cup T_{m,0} \cdot b_m \cup T_{0,m} \cdot b_m \cup T_{m,m} \cdot b_m$. In the first step shown above, we used the decomposition $T_n = T_2(T_m, \overline{T}_m)$, and, in analogy to (4), we determine first the solutions of the outer equations $T_2(x, y) = \overline{T}_2(x, y) = 0$, which are given in (14). The corresponding matrix is $\mathrm{DTT}_{2\times 2} \otimes \mathrm{I}_{m^2}$ and the result are four algebras of the form $\mathbb{C}[x, y]/\langle T_m - \alpha, \overline{T}_m - \beta\rangle$, with bases b_m, which are decomposed, respectively, by transforms we call *skew* DTTs (compare to the skew DCTs introduced in [3]). In these algebras, each (α, β) is a zero $(\alpha^{(2)}_{r,s}, \beta^{(2)}_{r,s})$ of $T_2 = \overline{T}_2 = 0$, where $0 \le r, s < 2$. We denote the associated skew DTT by $\mathrm{DTT}_{m\times m}(\alpha^{(2)}_{r,s}, \beta^{(2)}_{r,s})$. From the second step, we see which zeros of $T_n = \overline{T}_n = 0$ belong to which skew DTT, e.g., the first one collects all zeros with even i and even j. The final step is the permutation $P = \mathrm{L}^{n^2}_{n^2/2}(\mathrm{L}^{2n}_n \otimes \mathrm{I}_{n/2})$ (we omit the proof), where L^n_k is the stride permutation [8]. The final factorization is analogous to (7):

$$\begin{aligned}\mathrm{DTT}_{n\times n} = P(&\mathrm{DTT}_{m\times m}(\tfrac{2}{3}, \tfrac{2}{3}) \oplus \mathrm{DTT}_{m\times m}(0, 0)\\ &\oplus \mathrm{DTT}_{m\times m}(\tfrac{2}{3}\omega_3, \tfrac{2}{3}\omega_3^2) \oplus \mathrm{DTT}_{m\times m}(\tfrac{2}{3}\omega_3^2, \tfrac{2}{3}\omega_3))\\ &(\mathrm{DTT}_{2\times 2} \otimes \mathrm{I}_{m^2})B \quad (16)\end{aligned}$$

To obtain a complete recursion we have to further decompose the four occurring skew DTTs. This can be done using the same method, since the decomposition property (11) remains valid if constants are added to the polynomials. To state the decomposition, we first extend the above definition of skew DTTs. Let $(\alpha^{(n)}_{i,j}, \beta^{(n)}_{i,j})$ denote the zeros of $T_n = \overline{T}_n = 0$. Then the zeros of $T_n - \alpha^{(t)}_{r,s} = \overline{T}_n - \beta^{(t)}_{r,s} = 0$ are the subset of $(\alpha^{(tn)}_{i,j}, \beta^{(tn)}_{i,j})$ specified by $i \equiv r \bmod t,\ j \equiv s \bmod t$. We define the skew DTT as the polynomial transform for $\mathbb{C}[x, y]/\langle T_n - \alpha^{(t)}_{r,s}, \overline{T}_n - \beta^{(t)}_{r,s}\rangle$ with basis b_n:

$$\begin{aligned}&\mathrm{DTT}_{n\times n}(\alpha^{(t)}_{r,s}, \beta^{(t)}_{r,s}) =\\ &\qquad [T_{k,\ell}(\alpha^{(tn)}_{i,j}, \beta^{(tn)}_{i,j})]_{0\le i,j<tn,\ i\equiv r \bmod t,\ j\equiv s \bmod t}.\end{aligned}$$

In particular, $\mathrm{DTT}_{n\times n} = \mathrm{DTT}_{n\times n}(\alpha^{(1)}_{0,0}, \beta^{(1)}_{0,0}) = \mathrm{DTT}_{n\times n}(0, 0)$. The generalization of (16) to skew DTTs now becomes:

$$\begin{aligned}&\mathrm{DTT}_{n\times n}(\alpha^{(t)}_{r,s}, \beta^{(t)}_{r,s}) =\\ &P(\mathrm{DTT}_{m\times m}(\alpha^{(2t)}_{r,s}, \beta^{(2t)}_{r,s}) \oplus \mathrm{DTT}_{m\times m}(\alpha^{(2t)}_{r,s+t}, \beta^{(2t)}_{r,s+t})\\ \oplus &\mathrm{DTT}_{m\times m}(\alpha^{(2t)}_{r+t,s}, \beta^{(2t)}_{r+t,s}) \oplus \mathrm{DTT}_{m\times m}(\alpha^{(2t)}_{r+t,s+t}, \beta^{(2t)}_{r+t,s+t}))\\ &(\mathrm{DTT}_{2\times 2}(\alpha^{(t)}_{r,s}, \beta^{(t)}_{r,s}) \otimes \mathrm{I}_{m^2})B(\alpha^{(t)}_{r,s}, \beta^{(t)}_{r,s}), \quad (17)\end{aligned}$$

which is very similar to (16), only the base change matrix B depends on $\alpha^{(t)}_{r,s}, \beta^{(t)}_{r,s}$. To evaluate the arithmetic cost of (16) (via the cost of (17)), the main problem is to determine the cost incurred by the base change B, which is done next.

5. THE BASE CHANGE FOR THE RECURSION STEP

In this section we derive the precise form of the base change matrix $B(\alpha, \beta)$ in (17). As mentioned before, this matrix corresponds to the base change from b_n to $b'_n = T_{0,0} \cdot b_m \cup T_{m,0} \cdot b_m \cup T_{0,m} \cdot b_m \cup T_{m,m} \cdot b_m$. In other words we have to express every element $T_{k,\ell} \in b_n$ as a linear combination of the elements b'_n; the coefficient vectors obtained this way are the columns of $B(\alpha, \beta)$. In particular, we will see that the base change matrix is sparse, which makes (17) and thus (16) fast algorithms.

Theorem 1 Let $n = 2m$ and let $0 \le k, \ell < m$. Then the following equations define the base change matrix $B(\alpha, \beta)$ in (17). The special case $B = B(0, 0)$ in (16) is obtained by setting $\alpha = \beta = 0$. The following division into regions is according to Fig. 1.

Region 1: $T_{k,\ell} \in b'_n$ for $0 \le k, \ell < m$.

Region 2: $T_{m+k,\ell} =$

$$\begin{cases} T_{m,0} & : \ k = \ell = 0\\ \frac{3}{2}T_{m,0}T_{0,\ell} - \frac{1}{2}T_{m-\ell,0} & : \ k = 0,\ \ell \ne 0,\\ 3T_{m,0}T_{k,0} - 2T_{m-k,k} & : \ \ell = 0,\ k \ne 0,\\ 3T_{m,0}T_{k,\ell} - T_{m-\ell-k,k} - T_{m-k,k+\ell} & : \ k + \ell < m; k, \ell \ne 0,\\ 3T_{m,0}T_{k,\ell} - \frac{3}{2}T_{0,m}T_{\ell,0} - \frac{1}{2}T_{0,k} & : \ k + \ell = m,\\ 3T_{m,0}T_{k,\ell} - 3T_{0,m}T_{m-k,k+\ell-m} + T_{\ell,2m-k-\ell} & : \ k + \ell > m. \end{cases}$$

Region 3: The representation of polynomials in this region is derived from Region 2 by applying $T_{k,m+\ell} = \overline{T}_{m+\ell,k}$.

Region 4: $T_{m+k,m+\ell} =$

$$\begin{cases} T_{m,m} & : \ k = \ell = 0,\\ 3T_{m,m}T_{0,\ell} - 3T_{m,0}T_{\ell,m-\ell} + T_{0,\ell} & : \ k = 0,\ \ell \ne 0,\\ 3T_{m,m}T_{k,0} - 3T_{0,m}T_{m-k,k} + T_{k,0} & : \ \ell = 0,\ k \ne 0,\\ 6T_{m,m}T_{k,\ell} - T_{m-\ell,m-k} + 2T_{k,\ell} - 3T_{m,0}T_{k+\ell,m-\ell} - 3T_{0,m}T_{m-k,k+\ell} & : \ k + \ell < m,\\ 6T_{m,m}T_{k,\ell} + T_{k,\ell} - \frac{3}{2}T_{m,0}T_{\ell,0} - \frac{3}{2}T_{0,m}T_{0,k} - \frac{3}{2}\alpha T_{0,k} - \frac{3}{2}\beta T_{\ell,0} & : \ k + \ell = m,\\ 6T_{m,m}T_{k,\ell} + 2T_{k,\ell} + 3T_{m,0}T_{2m-\ell-k,k} - 3T_{m,0}T_{m-k,\ell+k-m} + 3T_{0,m}T_{\ell,2m-\ell-k} - 3T_{0,m}T_{\ell+k-m,m-\ell} - (3\alpha + 3\beta + 1)T_{m-k,m-k} & : \ k = \ell = \tfrac{2}{3}m,\\ 6T_{m,m}T_{k,\ell} + 3T_{m,0}T_{2m-\ell-k,k} + 2T_{k,\ell} - T_{m-\ell,m-k} - 3T_{m,0}T_{m-k,\ell+k-m} + 3T_{0,m}T_{\ell,2m-\ell-k} - 3T_{0,m}T_{\ell+k-m,m-\ell} - 3\alpha T_{k+\ell-m,m-\ell} - 3\beta T_{m-k,k+\ell-m} & : \ k + \ell > m. \end{cases}$$

Proof: We focus on one particular case: we show that the entry $T_{m+k,\ell}$ for a polynomial in region 2 is given by the claimed formula in case $k + \ell > m$. The other cases are shown analogously. We have to express every polynomial $T_{m+k,\ell}$, $0 \le k, \ell < m$, in region 2, in the basis b'_n. First note that using property (10) we obtain

$$T_{m,0}T_{k,\ell} = \tfrac{1}{3}(T_{m+k,\ell} + T_{k,\ell-m} + T_{k-m,\ell+m}),$$

which implies

$$T_{m+k,\ell} = 3T_{m,0}T_{k,\ell} - T_{k,\ell-m} - T_{k-m,\ell+m}. \quad (18)$$

This is an expansion of $T_{m+k,\ell}$ into polynomials which are not all in the basis b'_n. Thus, we rewrite the second and third polynomial on the right hand side of (18) using (9). We obtain

$$T_{k,\ell-m} = T_{k+\ell-m,m-\ell} = \begin{cases} T_{k+\ell-m,m-\ell} & : \ k + \ell \ge m,\\ T_{m-\ell-k,k} & : \ k + \ell < m. \end{cases}$$

Note that in both cases the given polynomial shown on the right hand side is an element of b'_n since it lies in region 1.

Next, we modify the term $T_{k-m,\ell+m}$ in (18). Again, we distinguish the two cases $k+\ell < m$ and $k+\ell \geq m$. If $k+\ell < m$, we can use the rule $T_{k-m,\ell+m} = T_{m-k,k+\ell} \in b'_n$ and get

$$T_{m+k,\ell} = 3T_{m,0}T_{k,\ell} - T_{m-k-\ell,k} - T_{m-k,k+\ell}, \qquad (19)$$

provided that $0 \leq k+\ell < m$. Hence, we can assume that $k+\ell \geq m$. We define $\mu = k+\ell-m$ and $\nu := m-k$. Note that $0 \leq \mu, \nu \leq m$ and that $0 \leq \mu+\nu < m$. Hence, we can use (19) as follows:

$$\begin{aligned} T_{k-m,\ell+m} &= \overline{T}_{k+\ell,m-k} \\ &= \overline{T}_{m+\mu,\nu} \\ &= 3\overline{T}_{m,0}T_{\mu,\nu} - \overline{T}_{m-\mu-\nu,\mu} - \overline{T}_{m-\mu,\mu+\nu} \\ &= 3T_{0,m}T_{\nu,\mu} - T_{\mu,m-\mu-\nu} - T_{\mu+\nu,m-\mu}, \end{aligned}$$

where the last line is a decomposition into elements of b'_n. Substituting back the values for μ and ν yields the formula

$$T_{k-m,\ell+m} = 3T_{0,m}T_{m-k,k+\ell-m} - T_{k+\ell-m,m-\ell} - T_{\ell,2m-k-\ell}.$$

Hence, we have found expressions for all terms on the right hand side of (18) and can compute the resulting expansion as

$$\begin{aligned} T_{m+k,\ell} &= 3T_{m,0}T_{k,\ell} - T_{k,\ell-m} - T_{k-m,\ell+m} \\ &= 3T_{m,0}T_{k,\ell} - T_{k+\ell-m,m-\ell} - 3T_{0,m}T_{m-k,k+\ell-m} \\ &\quad + T_{k+\ell-m,m-\ell} + T_{\ell,2m-k-\ell} \\ &= 3T_{m,0}T_{k,\ell} - 3T_{0,m}T_{m-k,k+\ell-m} + T_{\ell,2m-k-\ell} \end{aligned}$$

as claimed for the case $k+\ell > m$ in the theorem. The other cases $k=0$, $\ell=0$, and $k+\ell=m$ for polynomials in region 2 arise as special cases in which the linear combinations can be further simplified. □

6. ARITHMETIC COST

In this section we determine the arithmetic cost of the DTT algorithm (16), with the skew DTTs recursively expanded as in (17). Our cost measure is *complex* additions and multiplications. We assume a DTT for input size $n \times n$ and that n is a power of 2.

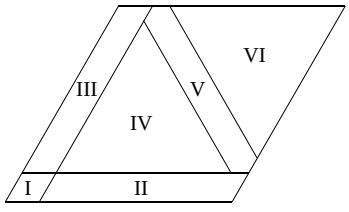

Fig. 2. Subdivision of each region of the grid into six subregions in order to compute the sparsity of the base change. The number of additions and multiplications required to compute the matrix-vector multiplication with a corresponding row depends on the subregion only.

Base change. We start by counting for each region 1–4 (see Fig. 1) the number of additions (adds) and multiplications (mults) necessary for the base change $B(\alpha,\beta)$ in (17). Each region is further subdivided into the six regions shown in Fig. 2. For the polynomials in region 1 we obtain the following table, where each entry denotes the number of complex additions and multiplications.

Region	Occurrences	Adds	Mults
$R_{1,\mathrm{I}}$	1	0	0
$R_{1,\mathrm{II}}$	$m-1$	4 (3)	2 (1)
$R_{1,\mathrm{III}}$	$m-1$	4 (3)	2 (1)
$R_{1,\mathrm{IV}}$	$(m-1)(m-2)/2$	6 (4)	3 (1)
$R_{1,\mathrm{V}}$	$m-1$	3	1
$R_{1,\mathrm{VI}}$	$(m-1)(m-2)/2$	6	1

The numbers in parentheses are the corresponding count for the non-skew DTT, i.e., for the special case $B(0,0)$. Similarly, we compute the number of additions and multiplications for regions 2 and 3 which are summarized in the following table. Note that the numbers $R_{2,\cdot}$ and $R_{3,\cdot}$ are identical. Hence, only the values for region 2 are given.

Region	Occurrences	Adds	Mults
$R_{2,\mathrm{I}}$	1	0	0
$R_{2,\mathrm{II}}$	$m-1$	1	2
$R_{2,\mathrm{III}}$	$m-1$	1	2
$R_{2,\mathrm{IV}}$	$(m-1)(m-2)/2$	2	2
$R_{2,\mathrm{V}}$	$m-1$	1	2
$R_{2,\mathrm{VI}}$	$(m-1)(m-2)/2$	2	2

Finally, the coefficients for region 4 turn out to be very simple; no additions and only one multiplication for each entry has to be performed.

Region	Occurrences	Adds	Mults
$R_{4,\mathrm{I}}$	1	0	0
$R_{4,\mathrm{II}}$	$m-1$	0	1
$R_{4,\mathrm{III}}$	$m-1$	0	1
$R_{4,\mathrm{IV}}$	$(m-1)(m-2)/2$	0	1
$R_{4,\mathrm{V}}$	$m-1$	0	1
$R_{4,\mathrm{VI}}$	$(m-1)(m-2)/2$	0	1

From the above we determine the total number of additions for the matrix-vector multiplication with $B(\alpha,\beta)$ as

$$\begin{aligned} & 17(m-1) + 20(m-1)(m-2)/2 \\ = \;& 10m^2 - 13m + 3, \end{aligned}$$

and the number of multiplications as

$$7m^2 - m - 6.$$

For the special case $B(0,0)$ the count is slightly less, namely $9m^2 - 12m + 3$ additions and $6m^2 - 6$ multiplications.

Base cases. Next, we consider the base case for the recursion, i. e., the operations needed for a skew-DTT of size 4×4. Since $T_{0,0} = 1$, each skew $\mathrm{DTT}_{2\times 2}(\alpha,\beta)$ has in the first column only 1's. Thus, the arithmetic cost is at most 12 additions and 9 multiplications.

For the non-skew DTT 2×2 in (15) the count is obviously less, but we can do even better by generating an algorithm using the GAP package AREP [9, 10]. Namely the function `Ma-`

`trixDecompositionByMonMonSymmetry` produces the following factorization (the dots represent zero entries):

$$\begin{bmatrix} \cdot & 1 & \cdot & \cdot \\ \cdot & \cdot & \cdot & 1 \\ 1 & \cdot & \cdot & \cdot \\ \cdot & \cdot & 1 & \cdot \end{bmatrix} \cdot \begin{bmatrix} 1 & \frac{2}{3}\omega_3^2 & \frac{2}{3}\omega_3 & \cdot \\ 1 & \frac{2}{3} & \frac{2}{3} & \cdot \\ 1 & \frac{2}{3}\omega_3 & \frac{2}{3}\omega_3^2 & \cdot \\ \cdot & \cdot & \cdot & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & \cdot & \cdot & \frac{1}{6} \\ \cdot & 1 & \cdot & \cdot \\ \cdot & \cdot & 1 & \cdot \\ 1 & \cdot & \cdot & -\frac{1}{2} \end{bmatrix}.$$

This shows that 8 additions and 7 multiplications are sufficient to perform a matrix-vector multiplication with $\mathrm{DTT}_{2\times 2}$.

Theorem 2 Let $n = 2^k$, where $k \geq 1$. Then the discrete triangle transform $\mathrm{DTT}_{n\times n}$ can be computed using at most

$$\begin{aligned} A(n) &= \frac{11}{2}n^2 \log n - \frac{43}{6}n^2 + \frac{15}{2}n - \frac{1}{3}, \\ M(n) &= 4n^2 \log n - \frac{7}{2}n^2 + \frac{3}{2}n + 2 \end{aligned}$$

additions and multiplications, respectively. In particular, the cost is $O(n^2 \log(n))$.

Proof: We first determine the number of operations for an arbitrary skew $\mathrm{DTT}_{n\times n}(\alpha, \beta)$ computed using the recursion (17), from left to right. The cost for the base change $B(\alpha, \beta)$ was determined above. The matrix $(\mathrm{DTT}_{2\times 2}(\alpha_{r,s}^{(t)}, \beta_{r,s}^{(t)}) \otimes \mathrm{I}_{m^2})$ incurs $12m^2$ additions and $9m^2$ multiplications. Next, the four skew DTTs are computed recursively and P does not incur any arithmetic cost. We obtain the following recurrences for additions and multiplications ($m = n/2$):

$$\begin{aligned} A_s(n) &= 4A_s(m) + 22m^2 - 13m + 3, \\ M_s(n) &= 4M_s(m) + 16m^2 - m - 6. \end{aligned}$$

The base cases are determined by the numbers $A_s(2) = 12$ and $M_s(2) = 9$, determined above. The solutions are the cost of a skew $\mathrm{DTT}_{n\times n}$:

$$\begin{aligned} A_s(n) &= \frac{11}{2}n^2 \log n - \frac{11}{2}n^2 + \frac{13}{2}n - 1 \\ M_s(n) &= 4n^2 \log n - \frac{5}{2}n^2 + \frac{1}{2}n + 2. \end{aligned}$$

Finally, we compute the cost of the non-skew $\mathrm{DTT}_{n\times n}$ from (16) and the cost of the skew DTTs computed above. Note that we take into account that in (16) one of the smaller (size $m \times m$) DTTs is not skew. We obtain the recursions ($m = n/2$)

$$\begin{aligned} A(n) &= A(m) + 3A_s(m) + 17m^2 - 12m + 3, \\ M(n) &= M(m) + 3M_s(m) + 13m^2 - 6, \end{aligned}$$

with initial conditions $A(2) = 8$ and $M(2) = 7$. Solving these recurrences yields the desired result. □

7. CONCLUSIONS

We presented a fast, $O(n^2 \log(n))$, algorithm for the discrete triangle transform (DTT) of input size $n \times n$. This puts the DTT into the same complexity class as other, separable, two-dimensional transforms. Similar to our previous work [2, 3, 9] on other trigonometric transforms, we derived the algorithm not by lengthy matrix entry manipulation, but rather by a stepwise decomposition of the polynomial algebra associated to the transform. More specifically, the derived DTT algorithm is based on a decomposition property of the polynomials $T_n(x, y), \overline{T}_n(x, y)$ that define the algebra and is thus the analogue of the Cooley-Tukey FFT (as shown in [3]), which justifies the title of this paper.

8. REFERENCES

[1] M. Püschel and M. Rötteler, "The Discrete Triangle Transform," in *Proc. ICASSP*, 2004.

[2] M. Püschel and J. M. F. Moura, "The Algebraic Approach to the Discrete Cosine and Sine Transforms and their Fast Algorithms," *SIAM Journal of Computing*, vol. 32, no. 5, pp. 1280–1316, 2003.

[3] M. Püschel, "Cooley-Tukey FFT like Algorithms for the DCT," in *Proc. ICASSP*, 2003, pp. 501–504.

[4] D. A. Cox, J. B. Little, and D. B. O'Shea, *Ideals, Varieties, and Algorithms*, Springer, 1997.

[5] T. Koornwinder, "Orthogonal polynomials in two variables which are eigenfunctions of two algebraically independent partial differential operators (part III)," *Indag. Math.*, vol. 36, pp. 357–369, 1974.

[6] R. Eier and R. Lidl, "A Class of Orthogonal Polynomials in k Variables," *Mathematische Annalen*, vol. 260, pp. 93–99, 1982.

[7] P. E. Ricci, "An Iterative Property of Chebyshev Polynomials of the First Kind in Several Variables," *Rend. Math. Appl.*, vol. 6, no. 4, pp. 555–563, 1986.

[8] R. Tolimieri, M. An, and C. Lu, *Algorithms for Discrete Fourier Transforms and Convolution*, Springer, 1997.

[9] S. Egner and M. Püschel, "Automatic Generation of Fast Discrete Signal Transforms," *IEEE Trans. on Signal Processing*, vol. 49, no. 9, pp. 1992–2002, 2001.

[10] S. Egner and M. Püschel, "Symmetry-Based Matrix Factorization," *Journal of Symbolic Computation*, vol. 37, no. 2, pp. 157–186, 2004.