Dimensionless Fast Fourier Transforms

L. Auslander
Department of Mathematics
The City University of New York
New York, NY 10036

J. R. Johnson
Department of Mathematics and Computer Science
Drexel University
Philadelphia, PA 19104

R. W. Johnson
Department of Computer Science
St. Cloud State University
St. Cloud, MN 56301

Technical Report DU-MCS-97-01
Department of Mathematics and Computer Science
Drexel University
Philadelphia, PA 19104

January 1997

Abstract

This paper shows that there are fast Fourier transform (FFT) algorithms that work, for a fixed number of points, independent of the dimension. Changing the dimension is achieved by relabeling the input and the output and changing the "twiddle factors." An important consequence of this result, is that a program designed to compute the 1-dimensional Fourier transform can be easily modified to compute the 2-dimensional and 3-dimensional Fourier transform on the same number of points.

1 Introduction

We have shown that for a fixed number of points there are Cooley-Tukey fast Fourier transform (FFT) algorithms that work for all multi-dimensional discrete Fourier transforms (DFT) with the same number of input points. Changing the dimension is achieved by relabeling the input and the output and changing the "twiddle factors." Such an algorithm is called a dimensionless FFT. The

dataflow of these algorithms are fixed, independent of the dimension. It is only the constant twiddle factors that must be changed when the dimension changes. The sequence of arithmetic operations and the order that the data is accessed remain the same.

An important consequence of this result, is that a program designed to compute the 1-dimensional DFT can be easily modified to compute the 2-dimensional and 3-dimensional DFTs on the same number of points. The modifications needed, input and output permutations and twiddle factors are well within the capability of reconfigurable hardware. So for a given number of points, we can design a device to compute the DFT for any appropriate dimension[1]. For instance, a chip can be designed to compute both a 1024-point 1-dimensional DFT and also a 32×32-point 2-dimensional DFT.

This counter intuitive result is an easy consequence of viewing multi-dimensional DFTs abstractly as Fourier transforms of finite Abelian groups. In this setting, dimension disappears, and FFT constructions can be developed that depend only on the order of the subgroups used in the Abelian group version of the FFT. This observation was emphasized in the previous paper [1], where it was shown how to classify multi-dimensional FFT algorithms in terms of the twiddle factors. This classification and the role played by the twiddle factors motivated the results in this paper. The key contribution of this paper is to recognize the consequences of the abstract point of view on the design and interpretation of the corresponding algorithms. Moreover, concrete formulas are derived that show how to compute the twiddle factors as a function of the dimension.

In Section 2 establishes the significance of a dimensionless FFT algorithm and states our main result in a form that connects the problem with the mathematics needed to solve it. Section 3 proves the existence of a dimensionless FFT using the fast Fourier transform theory of finite Abelian groups. In this setting, the proof is nearly identical to the derivation of the standard FFT. Finally, in Section 3, a second proof of the main theorem is provided. This proof, unlike the proof in Section 4 explicitly computes the twiddle factors and relabeling of data required to compute a given multi-dimensional DFT. These proofs are far more cumbersome, due to the notation required to keep track of the twiddle factors, however, they provide the detailed information required to implement the algorithm in hardware or software.

2 Problem Statement and Main Result

Let $X(a_1, \ldots, a_t)$ be a function of t variables, where $0 \leq a_j < n_j$. The t-dimensional $n_1 \times \cdots \times n_t$ DFT of X is

$$Y(b_1, \ldots, b_t) = \sum_{0 \leq a_j < n_j} e^{\frac{2\pi i}{n_1} a_1 b_1} \cdots e^{\frac{2\pi i}{n_t} a_t b_t} X(a_1, \ldots, a_t) \tag{1}$$

[1] U.§. Patent No. 6,003,056

Since the summation indices are independent Equation 1 can be written as a nested sum.

$$Y(b_1,\ldots,b_t) = \sum_{0\le a_1<n_1} e^{\frac{2\pi i}{n_1}a_1b_1}\left(\cdots\left(\sum_{0\le a_t<n_t} e^{\frac{2\pi i}{n_t}a_tb_t}X(a_1,\ldots,a_t)\right)\cdots\right).$$

The nested sum implies that the multi-dimensional DFT can be computed by applying one-dimensional DFTs along each dimension. A sequence of n_t-point DFTs are applied to the functions obtained by fixing the first $t-1$ inputs to X. Then a sequence of n_{t-1}-point DFTs are applied to the resulting function with all but the $(t-1)$-st variable fixed. This process continues until finally a sequence of n_1-point DFTs are applied. The one-dimensional DFTs are computed using the FFT. In the case of two-dimensions this is called the row-column algorithm since if the function $X(a_1,a_2)$ is stored lexicographically as an $n_1\times n_2$ matrix, the computation proceeds by applying n_2-point DFTs to the rows of X followed by n_1-point DFTs applied to the columns of the partially transformed matrix.

While this is the standard approach to computing multi-dimensional DFTs, it is not the only approach to computing multi-dimensional DFTs (see [1, 12, 13]). In this paper we show that there is a family of FFT algorithms that can compute any multi-dimensional DFT on a fixed number of points, independent of the dimension involved. The dataflow of these algorithms do not depend on dimension. The only thing that changes with dimension are the twiddle factors, multiplicative constants used in the algorithm, and the order of of the inputs and outputs. Any such algorithm is called a dimensionless FFT.

To illustrate the basic idea behind a dimensionless FFT we show a signal flow graph that is capable of computing all multi-dimensional DFTs on 16 points. The notation used for the signal flow graph follows the convention used in Van Loan[13]. The signal flow graph in Figure 1 corresponds to an iterative 16-point radix two FFT[2]. Computation in the signal flow graph proceeds from left to right. Inputs are entered at the 16 nodes on the left. Computation is denoted by the boxes, where the edges to the left of a box connect the inputs of the computation with the outputs on the right. Whenever, the inputs are available to a computation box, the computation can occur, and the computed values will then be available to the output nodes. When all values are propogated through the signal flow graph, the outputs of the computation appear at the ouput nodes at the right. The signal flow graph in Figure 1 consists of 4 stages with each stage consisting of 8 unspecified butterfly operations denoted by an empty square with two inputs and two outputs. A butterfly operation labled by a number ω and whose inputs are x_0 and x_1 and outputs are y_0 and y_1 specifies the computation:

$$y_0 = x_0 + \omega x_1 \tag{2}$$

$$y_1 = x_0 - \omega x_1 \tag{3}$$

A dimensionless FFT for 16 points is capable of computing a one-dimensional DFT on 16 points, a two-dimensional DFT on 2×8, 4×4, or 8×2 points, a

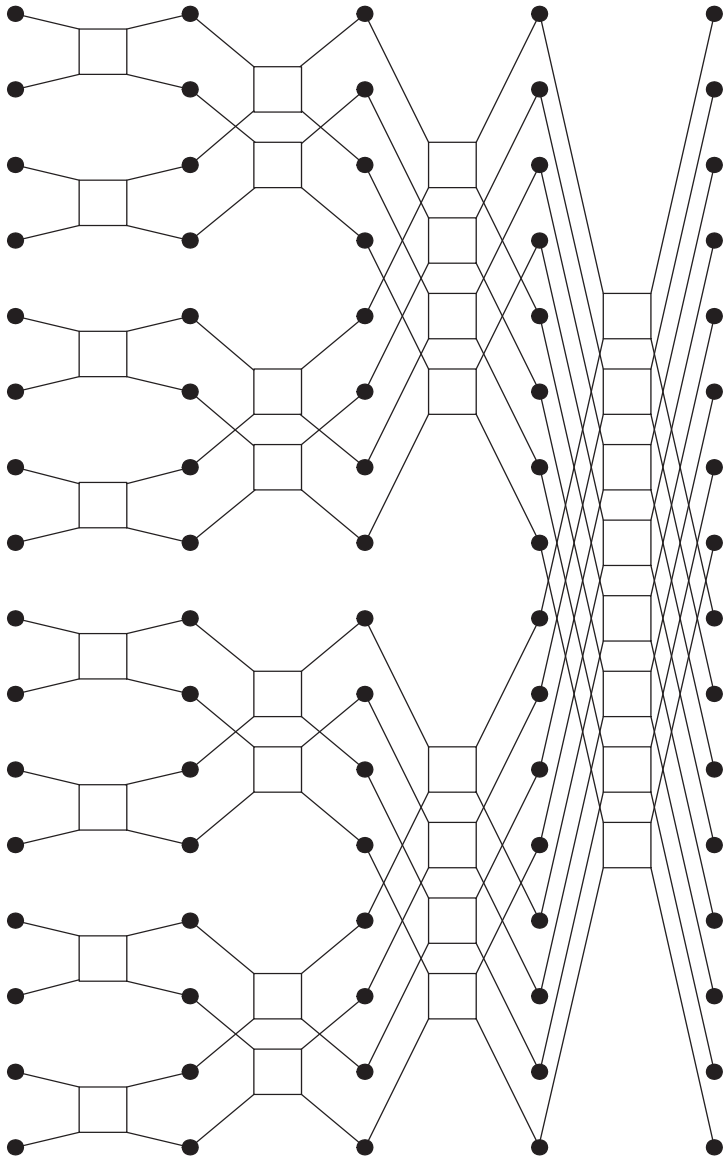

Figure 1: 16 point Cooley-Tukey dataflow

three-dimensional DFT on $4 \times 2 \times 2$, $2 \times 4 \times 2$, or $2 \times 2 \times 4$ points, and a four-dimensional DFT on $2 \times 2 \times 2 \times 2$ points. The signal flow graph in Figure 1 can be used to compute any of these transforms by inserting appropriate numbers, twiddle factors, in the butterfly operations and labeling the input and output points appropriately. Figure 2 shows the configuration for the one-dimensional 16-point DFT and Figure 3 shows the configuration for the two-dimensional 4×4 DFT. All twiddle factors are expressed as powers of $\omega = \exp(2\pi i/16)$. In these figures the initial configuration of the data is shown as an explicit permutation. Nonetheless, the initial permutation can be combined with loading the input data and is not viewed as an "essential" permutation. Other than this the internal dataflow is identical for both transforms. Similar configurations exist for the remaining two, three and four dimensional 16-point DFTs.

In order to prove the existence of arbitrary dimensionless FFTs and to facilitate the computation of the twiddles necessary to reconfigure the computation, we introduce a matrix representation for the computation.

The multidimensional DFT can be interpreted as a matrix-vector product. Let x and y be the vectors of size $N = n_1 \cdots n_t$ obtained by ordering the elements of X and Y lexicographically. Then, $y = (F_{n_1} \otimes \cdots \otimes F_{n_t})x$, where $\otimes$, denotes the tensor (Kronecker) product and F_{n_i} is the n_i-point discrete Fourier matrix [13]. The Fourier matrix is defined by

$$F_n = \left[\omega_n^{jk}\right]_{0 \le j,k < n},$$

where $\omega_n = \exp(2\pi i/n)$. The FFT provides an $O(n \log(n))$ algorithm to apply this matrix to an input vector. FFT algorithms can be obtained by factoring this matrix into a product of $O(\log(n))$ sparse structured matrices, each of which can be applied to a vector in $O(n)$ time [8, 11, 13].

The one-dimensional 16-point FFT in Figure 2 is equivalent to the factorization

$$F_{16} = (F_2 \otimes I_8)T_4(I_2 \otimes F_2 \otimes I_4)T_3(I_4 \otimes F_2 \otimes I_2)T_2(I_8 \otimes F_2)T_1 P \qquad (4)$$

where I_N is the $N \times N$ identity matrix,

$$\begin{aligned}
T_1 &= \mathrm{diag}(1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1),\\
T_2 &= \mathrm{diag}(1,1,1,\omega^4,1,1,1,\omega^4 1,1,1,\omega^4,1,1,1,\omega^4),\\
T_3 &= \mathrm{diag}(1,1,1,1,1,\omega^2,1,\omega^4,1,\omega^6,1,1,1,1,1,\omega^2,1,\omega^4,1,\omega^6),\\
T_4 &= \mathrm{diag}(1,1,1,1,1,1,1,1,1,\omega,\omega^2,\omega^3,\omega^4,\omega^5,\omega^6,\omega^7),
\end{aligned}$$

and P is the permutation matrix corresponding to the permutation

$$\begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 \\ 0 & 8 & 4 & 12 & 2 & 10 & 6 & 14 & 1 & 9 & 5 & 13 & 3 & 11 & 7 & 15 \end{pmatrix},$$

where the elements in the top row are mapped to the corresponding elements in the bottom row.

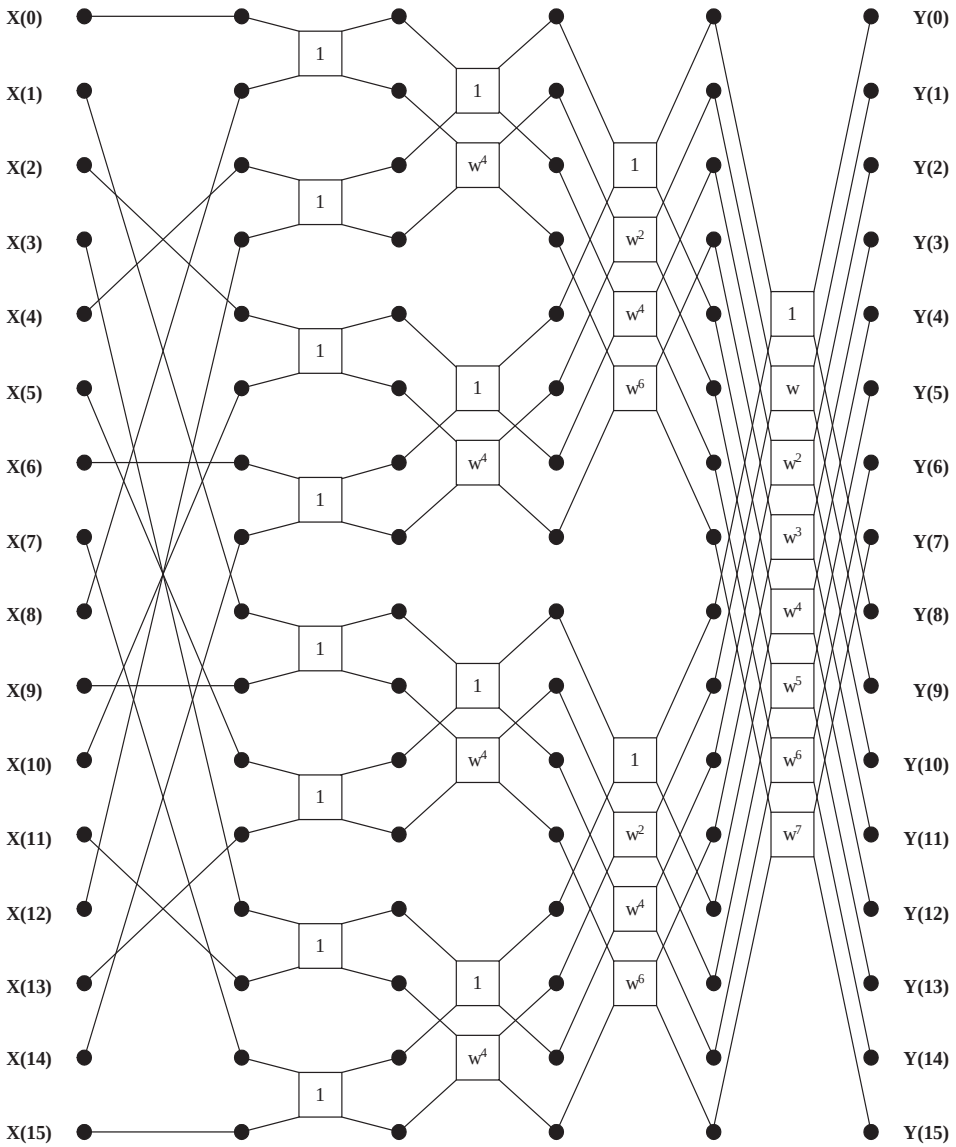

Figure 2: One-dimensional Cooley-Tukey signal flow graph

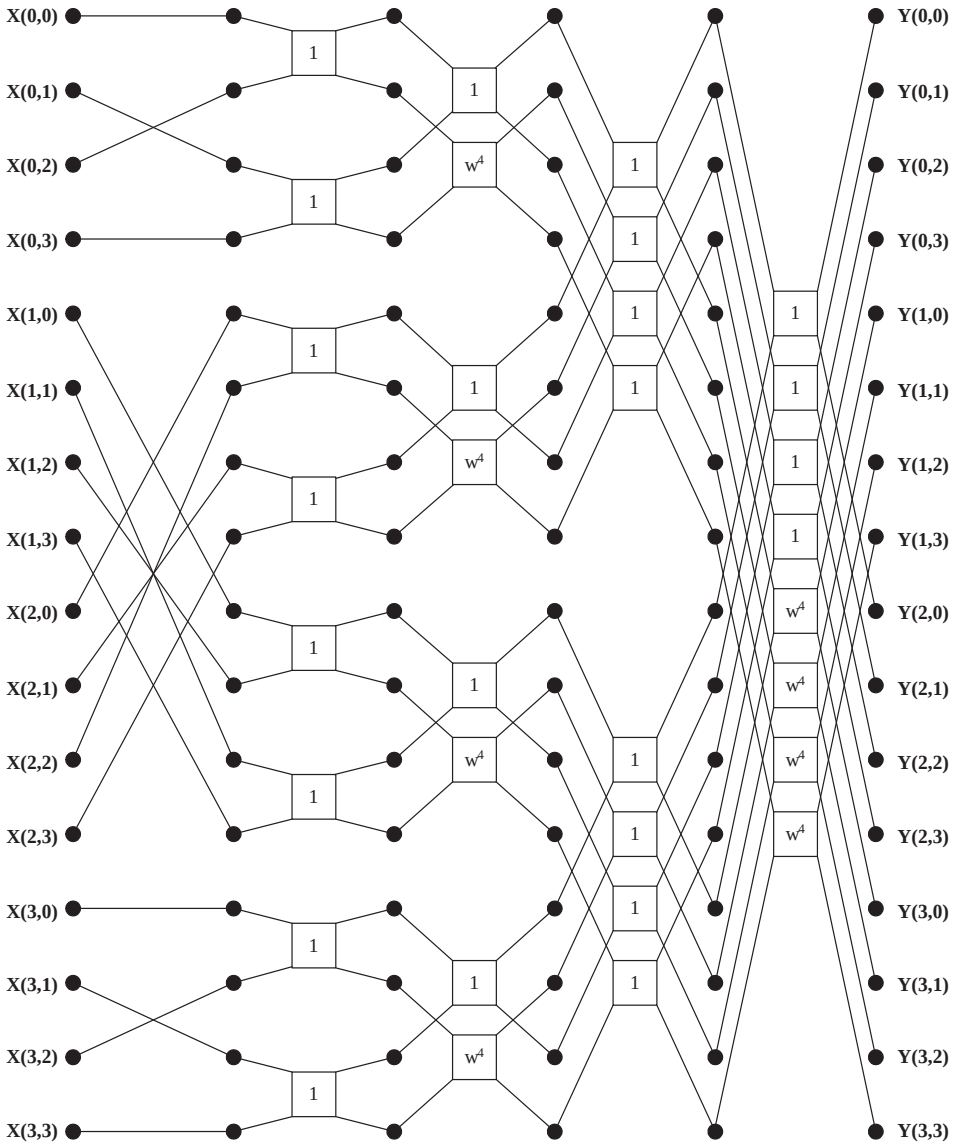

Figure 3: Two-dimensional Cooley-Tukey signal flow graph

The two-dimensional FFT in Figure 3 is equivalent to the factorization

$$F_4 \otimes F_4 = (F_2 \otimes I_8)T_4(I_2 \otimes F_2 \otimes I_4)T_3(I_4 \otimes F_2 \otimes I_2)T_2(I_8 \otimes F_2)T_1 P \qquad (5)$$

where

$$\begin{aligned}
T_1 &= \mathrm{diag}(1,1,1,\omega^4,1,1,1,\omega^4 1,1,1,\omega^4,1,1,1,\omega^4),\\
T_2 &= \mathrm{diag}(1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1),\\
T_3 &= \mathrm{diag}(1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1),\\
T_3 &= \mathrm{diag}(1,1,1,1,1,1,1,1,1,1,1,1,\omega^4,\omega^4,\omega^4,\omega^4),\\
T_4 &= \mathrm{diag}(1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1),
\end{aligned}$$

and

$$P = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 \\ 0 & 2 & 1 & 3 & 8 & 10 & 9 & 11 & 4 & 6 & 5 & 7 & 12 & 14 & 13 & 15 \end{pmatrix}.$$

Equations 4 and 5 are identical except for the values in the diagonal matrices T_i and the permutation matrix P. The dimensionless FFT algorithm generalizing these examples is formulated in the following theorem, whose proof is given in the next section.

Theorem 1 (Dimensionless FFT) *Let $\mathcal{F}_N$ denote an arbitrary multi-dimensional DFT of size N, and assume that $N = n_1 \times \cdots \times n_t$. Then there exist exist diagonal matrices T_i, $i = 1, \ldots, t$, multi-dimensional DFTs $\mathcal{F}_{n_i}$ of size n_i, and permutation matrices P and Q such that*

$$\mathcal{F}_N = P \left\{ \prod_{i=1}^{t} \left(I_{N(i-1)} \otimes \mathcal{F}_{n_i} \otimes I_{\bar{N}(i)} \right) T_i \right\} Q, \qquad (6)$$

where $N(i-1) = n_1 \cdots n_{i-1}$, $N(0) = 1$, and $\bar{N}(i) = N/N(i)$. In the special case where $N = p^t$ and $n_i = p$ for $i = 1, \ldots, t$, where p is a prime, all of the DFTs occuring in Equation 6 are equal to F_p.

3 A Generic FFT for Finite Abelian Groups

In this section we prove Theorem 1. The theorem is best understood and most easily proved using the FFT theory of finite Abelian groups. In this setting, we prove the existence of a generic FFT algorithm for Abelian groups. A generic FFT algorithm for the Fourier transform of the Abelian group A is an algorithm that depends only on the order of A, not on its type. Before proving the theorem, we review the necessary results from the theory of Abelian groups and their FFTs and we relate the DFTs of finite Abelian group to multi-dimensional DFTs.

Let A be a finite Abelian group whose order is denoted by $|A|$ (the group operation will be denoted by $+$). The set of complex valued functions on A with inner product

$$(f,g) = \frac{1}{|A|}\sum_{a\in A} f(a)\overline{g(a)}$$

form an inner product space denoted by $L^2(A)$.

A non-zero function $\chi \in L^2(A)$ such that

$$\chi(a_1 + a_2) = \chi(a_1)\chi(a_2) \text{ for } a_1, a_2 \in A,$$

is called a character. If χ_1 and χ_2 are characters then we can define $(\chi_1 + \chi_2)(a) = \chi_1(a)\chi_2(a)$. Under this operation the set of characters form a group, called the dual or character group, which we will denote by $\hat{A}$. It is well known that $\hat{A}$ is isomorphic to A. Moreover, since for $\chi_1, \chi_2 \in \hat{A}$,

$$(\chi_1, \chi_2) = \begin{cases} 1 & \text{if } \chi_1 = \chi_2 \\ 0 & \text{if } \chi_1 \neq \chi_2 \end{cases},$$

and $|\hat{A}| = |A|$, $\hat{A}$ is an orthonormal basis for $L^2(A)$. Hence, an arbitrary function, $f \in L^2(A)$ can be written uniquely as

$$f(x) = \sum_{\chi\in\hat{A}} \hat{f}(\chi)\chi(x), \text{ where } \hat{f}(\chi) = (f,\chi). \tag{7}$$

This is the Fourier series expansion of the function f. The coefficients in this expansion are called the Fourier coefficients and are obtained from the Fourier transform of f.

Definition 1 (Fourier Transform)
The Fourier transform of A, $F(A) : L^2(A) \to L^2(\hat{A})$ is defined by

$$F(A)(f)(\chi) = \hat{f}(\chi) = \frac{1}{|A|}\sum_{a\in A} f(a)\overline{\chi(a)} = (f,\chi).$$

By the fundamental theorem of Abelian groups [4], any finite Abelian group is isomorphic to a direct product of cyclic groups, and hence is isomorphic to $\mathbf{Z}/n_1\mathbf{Z}\times\cdots\times\mathbf{Z}/n_t\mathbf{Z}$, where $\mathbf{Z}/n_i\mathbf{Z}$ is the additive group of integers mod n. Such an isomorphism is called a presentation. Choice of presentation is not unique; however, it is possible to determine all possible presentations from the prime power factorization of the orders of the cyclic factors of any given presentation.

Suppose $\mathbf{Z}/n_1\mathbf{Z}\times\cdots\times\mathbf{Z}/n_t\mathbf{Z}$ is a presentation for A, and let $a = (a_1,\ldots,a_t)$ and $x = (x_1,\ldots,x_t)$ be arbitrary elements in A. Define $a \mapsto \hat{a}$ by

$$\hat{a}(x) = \hat{a}_1(x_1)\cdots\hat{a}_t(x_t),$$

where

$$\hat{a}_j(x_j) = e^{\frac{2\pi i}{n_j}a_j x_j}.$$

It is easy to verify that $\hat{a} \in \hat{A}$ and the mapping $a \to \hat{a}$ is an isomorphism from A onto $\hat{A}$.

Using this presentation and the corresponding isomorphism, the definition of $F(A)$ becomes:

$$\begin{aligned} \hat{f}(\hat{a}) &= \hat{f}(\hat{a}_1, \ldots, \hat{a}_t) \\ &= \sum_{(x_1,\ldots,x_t)} f(x_1, \ldots, x_t) e^{\frac{2\pi i}{n_1} a_1 x_1 + \cdots + \frac{2\pi i}{n_t} a_t x_t}. \end{aligned}$$

This is the standard definition of the t-dimensional Fourier transform of $n_1 \times \cdots \times n_t$ points, which we denote by $F_{(n_1,\ldots,n_t)} = F_{n_1} \otimes \cdots \otimes F_{n_t}$.

Our proof of Theorem 1 depends on the generalization of the Cooley-Tukey FFT for Abelian groups.

Theorem 2 *Let A be a finite Abelian group, $B \leq A, C = A/B$ and assume that bases have been chosen for $L^2(A)$ and $L^2(\hat{A})$ so that $F(A)$ can be written as a matrix. Then*

$$F(A) = Q(F(B) \otimes I_{|C|})T(I_{|B|} \otimes F(C))P,$$

where Q and P are permutations and T is a diagonal matrix.

Proof See [5] or [12]. ▮

We will also need the special case of this theorem which generalizes the prime factor FFT [3, 10].

Theorem 3 *Let A be a finite Abelian group and $B \leq A$ such that there exists $C \leq A$ with $A = B \oplus C$ and C isomorphic to A/B. Further assume that bases have been chosen for $L^2(A)$ and $L^2(\hat{A})$ so that $F(A)$ can be written as a matrix. Then there exist permutation matrices such that*

$$F(A) = Q(F(B) \otimes I_{|C|})(I_{|B|} \otimes F(C))P = Q(F(B) \otimes F(C))P.$$

Proof This result is obtained as a corollary to Theorem 2 in [5]. An alternative proof can be found in [12]. ▮

The factorization in the following Theorem depends only on the order of A. Two different Abelian groups, both of order n, have the same factorization except the values of the diagonal elements in T_i may change and the permutations P_A and Q_A depends on the group A. Therefore the factorization provides a generic FFT algorithm for computing the Fourier transform of any Abelian group of order n. This theorem is equivalent to Theorem 1 once presentations are chosen and the Fourier transforms of Abelian groups are replaced by multi-dimensional DFTs.

Theorem 4 (Generic Abelian FFT) *Let A be a finite abelian group of order n and let*

$$F(A) : L^2(A) \longrightarrow L^2(\hat{A})$$

denote the Fourier transform associated with A. $F(A)$ can be viewed as a matrix as soon as bases are chosen for $L^2(A)$ and $L^2(\hat{A})$.

Suppose $n = n_1 \cdots n_t$, then

$$F(A) = Q_A \left\{ \prod_{i=1}^{t} \left(I_{N(i-1)} \otimes F(A_i) \otimes I_{\bar{N}(i)} \right) T_i \right\} P_A$$

where

$$\begin{array}{rcl} N(i-1) & = & n_1 \cdots n_{i-1} \text{ with } N(0) = 1 \text{ and } N(t) = N, \\ \bar{N}(i) & = & N/N(i), \\ I_N & & \text{is the } N \times N \text{ identity matrix,} \\ T_i & & \text{is a diagonal matrix,} \\ Q_A & & \text{is a permutation matrix,} \\ P_A & & \text{is a permutation matrix,} \\ A_i & & \text{is an Abelian group of order } n_i. \end{array}$$

Proof The proof is similar to the derivation of the iterative FFT (see Theorem 7 in Section 4.2) found in [8] and utilizes simple tensor product manipulation (see Section 4.1 for a summary of the necessary tensor product identities). We proceed by induction on t. Since $n_1 | n$ there exists a subgroup $A_1 < A$ of order n_1. By Theorem 2,

$$F(A) = Q_1 (F(A_1) \otimes I_{n_2 \cdots n_t}) T_1' (I_{n_1} \otimes F(C)) P_1 \tag{8}$$

where $C = A/A_1$ and $|C| = n_2 \cdots n_t$.

By induction the factorization in the theorem holds for C thus

$$F(C) = Q_C \left\{ \prod_{i=2}^{t} (I_{N(i-1)/n_1} \otimes F(A_i) \otimes I_{\bar{N}(i)}) T_i' \right\} P_C. \tag{9}$$

Substituting this into Equation 8 we obtain

$$\begin{array}{rcl} F(A) & = & Q_1 (F(A_1) \otimes I_{\bar{N}(1)}) T_1' \\ & & \left(I_{n_1} \otimes Q_C \left\{ \prod_{i=2}^{t} (I_{N(i-1)/n_1} \otimes F(A_i) \otimes I_{\bar{N}(i)}) T_i' \right\} \right) P_C. \end{array} \tag{10}$$

Applying Proposition 3 the identity I_{n_1} can be brought into the product, which implies

$$\begin{array}{rcl} F(A) & = & Q_1 (F(A_1) \otimes I_{\bar{N}(1)}) T_1' (I_{n_1} \otimes Q_C) \\ & & \left\{ \prod_{i=2}^{t} (I_{N(i-1)} \otimes F(A_i) \otimes I_{\bar{N}(i)}) (I_{n_1} \otimes T_i') \right\} (I_{n_1} \otimes P_C) P_1. \end{array} \tag{11}$$

Since T_1' is a diagonal matrix and $I_{n_1} \otimes Q_C$ is a permutation, there is a diagonal matrix T_1 such that

$$T_1'(I_{n_1} \otimes Q_C) = (I_{n_1} \otimes Q_C)T_1,$$

and by Proposition 4, $(F(A_1) \otimes I_{\bar{N}(1)})(I_{n_1} \otimes Q_C) = (I_{n_1} \otimes Q_C)(F(A_1) \otimes I_{\bar{N}(1)})$. Consequently,

$$\begin{aligned} F(A) &= Q_1(I_{n_1} \otimes Q_C)(F(A_1) \otimes I_{\bar{N}(1)})T_1 \\ &\quad \left\{ \prod_{i=2}^{t} (I_{N(i-1)} \otimes F(A_i) \otimes I_{\bar{N}(i)})(I_{n_1} \otimes T_i') \right\} (I_{n_1} \otimes P_C)P_1. \end{aligned} \tag{12}$$

Setting $P_A = (I_{n_1} \otimes P_C)P_1$, $Q_A = Q_1(I_{n_1} \otimes Q_C)$, and $T_i = (I_{n_1} \otimes T_i')$ for $i = 2, \ldots, t$, the theorem is obtained. ∎

A generic version of the prime factor FFT can be obtained in a similar fashion using Theorem 3.

Theorem 5 (Generic Prime Factor FFT) *Let A be a finite abelian group of order n and let*

$$F(A) : L^2(A) \longrightarrow L^2(\hat{A})$$

denote the Fourier transform associated with A. $F(A)$ can be viewed as a matrix as soon as bases are chosen for $L^2(A)$ and $L^2(\hat{A})$.

Suppose $n = n_1 \cdots n_t$, where $n_i = p_i^{e_i}$, $i = 1, \ldots, t$ and are $p_1, \ldots, p_t$ are distinct primes, then

$$F(A) = Q_A(F(A_1) \otimes \cdots \otimes F(A_t))P_A = Q_A \left\{ \prod_{i=1}^{t} \left(I_{N(i-1)} \otimes F(A_i) \otimes I_{\bar{N}(i)} \right) \right\} P_A,$$

where A_i is a p_i-group of order n_i.

Proof The proof is by induction on t and is similar to the proof of the previous theorem. Let A_1 be the Sylow p_1 subgroup. The subgroup A_1 is of order n_1 and there exits a subgroup, B_1 of order $n_2 \cdots n_t$ such that $A = A_1 \oplus B_1$. By Theorem 3, there exists permutations P_1 and Q_1 such that

$$F(A) = Q_1(F(A_1) \otimes I_{\bar{N}(1)}(I_{n_1} \otimes F(B_1))P_1.$$

The theorem is proved using the inductive hypothesis on $F(B_1)$ and properties of the tensor product as was done in the previous proof.

4 A Concrete Version of the Main Theorem

In this section we provide an alternative proof of Theorem 1 that does not rely on the fast Fourier transform theory of Abelian groups. The benefit of the proof in this section is that it provides explicit formulas for the twiddle factor matrices and the permutations that relabel the input and output vectors. The calculations rely on factorizations of the Fourier matrix and properties of the tensor product. Before proving the theorem we review the necessary tensor product identities and factorization theorems.

4.1 Tensor Product Identities

Let A be an $m \times n$ matrix and let B be a $p \times q$ matrix. The tensor product of A and B, $A \otimes B$, is the $mp \times nq$ block matrix $[a_{i,j}B]_{0 \le i < n,\ 0 \le j < q}$. The tensor product of the product of matrices is the product of the tensor products.

Proposition 1 (Multiplicative Property of Tensor Products)

$$(A \otimes B)(C \otimes D) = AC \otimes BD$$

whenever the matrices are compatible under matrix multiplication.

The following identities follow easily from the multiplicative property of the tensor product.

Proposition 2 *Assume that A_i and B_i are compatible under matrix multiplication. Then*

$$(\bigotimes_{i=1}^{t} A_i)(\bigotimes_{i=1}^{t} B_i) = (\bigotimes_{i=1}^{t} A_i B_i)$$

A useful special case occurs when all of the A_i matrices are the identity.

Proposition 3 *Let A_i, $i = 1, \ldots, t$ be a sequence of matrices and I be an arbitrary identity matrix. Then,*

$$I \otimes \prod_{i=1}^{t} A_i = \prod_{i=1}^{t} (I \otimes A_i).$$

The multiplicative property can be used to decompose a tensor product.

Proposition 4 *Let A be an $m \times n$ matrix and let B be a $p \times q$ matrix. Then*

$$A \otimes B = (A \otimes I_p)(I_n \otimes B) = (I_m \otimes B)(A \otimes I_q). \tag{13}$$

Proposition 5 *Assume that A_i is an $n_i \times n_i$ matrix, and let $N(i) = N_1 \cdots N_i$, $\bar{N}(i) = N/N(i)$, $N(0) = 1$, and $N(t) = N$. Then*

$$\bigotimes_{i=1}^{t} A_i = \prod_{i=1}^{t} (I_{N(i-1)} \otimes A_{n_i} \otimes I_{\bar{N}(i)})$$

This proposition can be applied to the multidimensional DFT.

Proposition 6 *Let $\mathcal{F}_N = F_{n_1} \otimes \cdots \otimes F_{n_t}$ be the t-dimensional $n_1 \times \cdots \times n_t$ DFT. Then*

$$\mathcal{F}_N = \prod_{i=1}^{t} (I_{N(i-1)} \otimes F_{n_i} \otimes I_{\bar{N}(i)})$$

In general, the tensor product is not commutative. However, $A \otimes B$ and $B \otimes A$ only differ by a permutation. The following permutation can be used to commute the tensor product.

Definition 2 (Stride Permutation) *Let e_i^m, $0 \le i < m$ be the vector of size m with a 1 in the i-th position and zeros elsewhere. The stride permutation L_n^{mn} is defined by*

$$L_n^{mn}(e_i^m \otimes e_j^n) = (e_j^n \otimes e_i^m).$$

Proposition 7 (Commuting Tensor Products) *Let A be an $m \times m$ matrix and let B be an $n \times n$ matrix. Then,*

$$A \otimes B = (L_n^{mn})^{-1}(B \otimes A)L_n^{mn} = L_m^{mn}(B \otimes A)L_n^{mn}.$$

Occaisionally we will have need to permute the factors in a tensor product of more than two factors. The following permutation generalizes the stride permutation.

Definition 3 (Digit Permutation) *Let $n = n_1 \cdots n_t$ and let σ be a permutation of the numbers $1, \ldots, n$.*

$$L_\sigma^{(n_1,\ldots,n_t)}(e_{i_1}^{n_1} \otimes \cdots \otimes e_{i_t}^{n_t}) = (e_{i_{\sigma(1)}}^{n_{\sigma(1)}} \otimes \cdots \otimes e_{i_{\sigma(t)}}^{n_{\sigma(t)}})$$

Proposition 8 (Permuting Tensor Products) *Let A_i, $1 \le i \le t$, be an $n_i \times n_i$ matrix, and let σ be a permutation of size t. Then,*

$$A_1 \otimes \cdots \otimes A_t = (L_\sigma^{(n_1,\ldots,n_t)})^{-1}(A_{\sigma(1)} \otimes \cdots \otimes A_{\sigma(t)})L_\sigma^{(n_1,\ldots,n_t)}.$$

Digit reversal is a special digit permutation that arises in FFT algorithms. It is a generalization of the bit-reversal permutation.

Definition 4 (Digit Reversal) *The t-digit digit-reversal permutation $R^{(n_1,\ldots,n_t)}$ is a permutation matrix of size $N = n_1 \cdots n_t$ defined by*

$$R^{(n_1,\ldots,n_t)}(e_{i_1}^{(n_1} \otimes \cdots \otimes e_{i_t}^{n_t}) = e_{i_t}^{n_t} \otimes \cdots \otimes e_{i_1}^{n_1}. \tag{14}$$

The special case when $n_1 = \cdots = n_t = n$ is denoted by R_{n^t}.

4.2 Mixed-Radix FFT Algorithms

In this section we review the factorizations of the DFT matrix corresponding to the iterative FFT. Similar presentations can be found in the books by Tolimieri et. al [11] and Van Loan [13]. The notation used here comes from [8].

The diagonal matrices arising in the FFT are called twiddle factor matrices.

Definition 5 (Twiddle Factor Matrix) *The twiddle factor matrix T_n^{mn} is defined by*

$$T_n^{mn}(e_i^m \otimes e_j^n) = \omega_{mn}^{ij}(e_i^m \otimes e_j^n).$$

This implies that

$$T_n^{mn} = \bigoplus_{i=0}^{m-1} W_n(\omega_{mn})^i, \text{ where } W_n(\omega_{mn}) = \mathrm{diag}(1, \omega_{mn}, \ldots, \omega_{mn}^{n-1}).$$

The following theorem is the basis of the FFT. It was originally presented in [2].

Theorem 6 (Cooley-Tukey) *The mn-point DFT matrix is equal to*

$$F_{mn} = (F_m \otimes I_n)T_n^{mn}(I_m \otimes F_n)L_m^{mn}.$$

Assume that $N = p^k$. The following theorem, which corresponds to the standard iterative radix-p FFT, can be obtained by induction using the previous theorem (see [8, 11, 13]).

Theorem 7 (Iterative Radix p FFT) *Let $N = p^k$ and R_N denote the radix p, k-digit, digit-reversal permutation. Then*

$$F_N = \prod_{i=k}^{1} \left\{ (I_{2^{l-1}} \otimes F_2 \otimes I_{2^{k-l}})(I_{2^{l-1}} \otimes T_{2^{k-l}}^{2^{k-l+1}}) \right\} R_N \tag{15}$$

Example 1 (Iterative Radix 2 FFT) *We illustrate the proof and statement of the previous theorem using a 16-point DFT. This computation provides the twiddle factor matrices and initial permutation given in the example in Section 2.*

$$\begin{aligned}
F_8 &= (F_2 \otimes I_4)T_4^8(I_2 \otimes F_4)L_2^8 \\
&= (F_2 \otimes I_4)T_4^8(I_2 \otimes (F_2 \otimes I_2)T_2^4(I_2 \otimes F_2)L_2^4)L_2^8 \\
&= (F_2 \otimes I_4)T_4^8(I_2 \otimes F_2 \otimes I_2)(I_2 \otimes T_2^4)(I_4 \otimes F_2)(I_2 \otimes L_2^4)L_2^8 \\
&= (F_2 \otimes I_4)T_4^8(I_2 \otimes F_2 \otimes I_2)(I_2 \otimes T_2^4)(I_4 \otimes F_2)R_8
\end{aligned}$$

Substituting this expression into

$$F_{16} = (F_2 \otimes I_8)T_8^{16}(I_2 \otimes F_8)L_2^{16}$$

we arrive at

$$\begin{aligned}
F_{16} &= (F_2 \otimes I_8)T_8^{16}(I_2 \otimes F_2 \otimes I_4)(I_2 \otimes T_4^8)(I_4 \otimes F_2 \otimes I_2)(I_4 \otimes T_2^4) \\
&\quad (I_8 \otimes F_2)(I_2 \otimes R_8)L_2^{16} \\
&= (F_2 \otimes I_8)T_8^{16}(I_2 \otimes F_2 \otimes I_4)(I_2 \otimes T_4^8)(I_4 \otimes F_2 \otimes I_2)(I_4 \otimes T_2^4)(I_8 \otimes F_2)R_{16}.
\end{aligned}$$

It is easy to verify that $(I_2 \otimes R_8)L_2^{16} = R_{16}$ by applying both sides of the equation to $e_{i_1}^2 \otimes e_{i_2}^2 \otimes e_{i_3}^2 \otimes e_{i_4}^2$.

In a similar fashion we can obtain the more general mixed-radix factorization.

Theorem 8 *[Iterative Mixed-Radix FFT] Let $N = N_1 \cdots N_t$. Using the notation $N(i) = N_1 \cdots N_i$, $\bar{N}(i) = N/N(i)$, $N(0) = 1$, and $N(t) = N$, Then*

$$F_N = \prod_{i=1}^{t} \left\{ (I_{N(i-1)} \otimes F_{N_i} \otimes I_{\bar{N}(i)})(I_{N(i-1)} \otimes T_{\bar{N}(i)}^{\bar{N}(i-1)}) \right\} R \tag{16}$$

where

$$R = \prod_{i=t}^{1} (I_{N(i-1)} \otimes L_{N_i}^{\bar{N}(i-1)}) = R^{(N_t,\ldots,N_1)} \tag{17}$$

is the digit-reversal permutation $R^{(N_t,\ldots,N_1)}$ *is defined by*

Example 2 (Iterative Mixed-Radix FFT) *In this example, we compute a 16-point DFT using a multi-radix factorization with radices* 2, 4, *and* 2.

$$\begin{aligned}
F_{16} &= (F_2 \otimes I_8) T_8^{16} (I_2 \otimes F_8) L_2^{16} \\
&= (F_2 \otimes I_8) T_8^{16} (I_2 \otimes (F_4 \otimes I_2) T_2^8 (I_4 \otimes F_2) L_4^8) L_2^{16} \\
&= (F_2 \otimes I_8) T_8^{16} (I_2 \otimes F_4 \otimes I_2)(I_2 \otimes T_2^8)(I_8 \otimes F_2)(I_2 \otimes L_4^8) L_2^{16} \\
&= (F_2 \otimes I_8) T_8^{16} (I_2 \otimes F_4 \otimes I_2)(I_2 \otimes T_2^8)(I_8 \otimes F_2) R^{(2,4,2)}
\end{aligned}$$

The fact that $(I_2 \otimes L_4^8) L_2^{16} = R^{(2,4,2)}$ *can easily be verified by applying the equation to* $e_{i_3}^2 \otimes e_{i_2}^4 \otimes e_{i_1}^2$.

Next we review a factorization of the DFT matrix corresponding to the prime factor algorithm. This factorization requires a permutation arising from the Chinese remainder theorem (see [11] for a discussion of the Chinese remainder theorem).

Definition 6 (Chinese Remainder Theorem Permutation) *The Chinese remainder theorem implies that* $\mathbf{Z}/n_1 \cdots n_t \mathbf{Z} \cong \mathbf{Z}/n_1\mathbf{Z} \times \cdots \times \mathbf{Z}/n_t\mathbf{Z}$ *when* $\gcd(n_i, n_j) = 1$ *for* $i \neq j$. *The isomorphism is given by a system of orthogonal idempotents,* $\epsilon_1, \ldots, \epsilon_t$, *where*

$$\epsilon_1 + \cdots + \epsilon_t = 1 \tag{18}$$

$$\epsilon_i \epsilon_j = 0 \text{for } i \neq j \tag{19}$$

$$\epsilon_i \epsilon_i = \epsilon_i \tag{20}$$

This isomorphism defines a permutation of the integers $0, \ldots, N-1$, *where* $N = n_1 \cdots n_t$.

$$\mathrm{CRT}(n_1, \ldots, n_t)(e_{i_1}^{n_1} \otimes \cdots \otimes e_{i_t}^{n_t}) = e_{\epsilon_1 i_1 \pmod{n_1}}^{n_1} \otimes \cdots \otimes e_{\epsilon_t i_t \pmod{n_t}}^{n_t} \tag{21}$$

The prime factor algorithm for two factors is due to Good [3] and Thomas [10].

Theorem 9 (Good-Thomas) *Assume that* $\gcd(m, n) = 1$. *Let* ϵ_m *and* ϵ_n *be the orthogonal idempotents from the Chinese remainder theorem. Define* S_a^n, *for* $\gcd(a, n) = 1$ *be the permutation matrix that maps* e_i^n *to* $e_{ai \pmod{n}}^n$. *Then,*

$$F_{mn} = Q(F_m \otimes I_n)(I_m \otimes F_n)P = Q(F_m \otimes F_n)P,$$

where $P = \mathrm{CRT}(m, n)$ *and* $Q = \mathrm{CRT}(m, n)^{-1}(S_{\epsilon_m/n}^m \otimes S_{\epsilon_n/m}^n)$.

Example 3 (Good-Thomas) *Let $n_1 = 3$ and $n_2 = 5$. Then $\epsilon_3 = 10$ and $\epsilon_5 = 6$.*

$$F_{15} = \mathrm{CRT}(3,5)^{-1}(S_2^3 \otimes S_3^5)(F_3 \otimes F_5)\mathrm{CRT}(3,5),$$

where

$$\mathrm{CRT}(3,5) = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 \\ 0 & 6 & 12 & 3 & 9 & 10 & 1 & 7 & 13 & 4 & 5 & 11 & 2 & 8 & 14 \end{pmatrix},$$

and

$$S_2^3 = \begin{pmatrix} 0 & 1 & 2 \\ 0 & 2 & 1 \end{pmatrix} \text{ and } S_2^5 = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 \\ 0 & 2 & 4 & 1 & 3 \end{pmatrix}.$$

The Good-Thomas algorithm can be generalized to t factors.

Theorem 10 (Prime Factor FFT) *Let $N = n_1 \cdots n_t$, with $n_j = p_j^{e_j}$. Then there exists permutations $P^{(n_1,\ldots,n_t)}$ and $Q^{(n_1,\ldots,n_t)}$ such that*

$$\begin{aligned} F_N &= Q^{(n_1,\ldots,n_t)}(F_{n_1} \otimes \cdots \otimes F_{n_t})P^{(n_1,\ldots,n_t)} \\ &= Q^{(n_1,\ldots,n_t)}\left(\prod_{j=1}^{t} I_{N(j-1)} \otimes F_{n_j} \otimes \bar{N}(j)\right) P^{(n_1,\ldots,n_t)}, \end{aligned}$$

where $P^{(n_1,\ldots,n_t)} = \mathrm{CRT}(n_1,\ldots,n_t)$ and $Q^{(n_1,\ldots,n_t)} = \mathrm{CRT}(n_1,\ldots,n_t)^{-1}(S^{n_1}_{\epsilon_1 n_1/N} \otimes \cdots \otimes S^{n_t}_{\epsilon_t n_t/N})$

4.3 Dimensionless FFT Algorithms

In this section we provide an explicit version of Theorem 1 where formulas are given for the initial permutation and the twiddle factor matrices. All of the calculations will be done using the one-dimensional Cooley-Tukey factorization and manipulations of tensor products. Since the notation is more difficult than the ideas behind the theorem we will provide motivating examples which illustrate the main ideas behind the statement and proof of the theorem. Additionally we will first prove the special case of Theorem 1 where the radix is a prime. We will reduce the general problem to the case where the size of the transform is a prime power using a concrete version of Theorem 5. This reduction does not cover the full generality of Theorem 1; however, it does provide a dimensionless algorithm that works for all sizes.

An arbitrary multi-dimension DFT can be reduced to a seried of multi-dimension DFT's of prime power size. This reduction, corresponding to Theorem 5, works independent of dimension, since the only structural part of the factorization that depends on the dimension is the initial and final permutations. The following example shows a concrete computation that performs this reduction.

Example 4 (Dimensionless Prime Factor Algorithm) *Let $\mathcal{F}_n$ denote an arbitrary multidimensional DFT of size $n = 2^5 \cdot 3^4 \cdot 5^3 = 324000$. There exists permutations P and Q such that*

$$\mathcal{F}_n = P(\mathcal{F}_{2^5} \otimes \mathcal{F}_{3^3} \otimes \mathcal{F}_{5^3})Q,$$

where $\mathcal{F}_{2^5}$, $\mathcal{F}_{3^4}$, and $\mathcal{F}_{5^3}$ are multi-dimensional DFTs of a given type of size 2^5, 3^4, and 5^3.

For example, consider

$$\mathcal{F}_n = F_{48} \otimes F_{90} \otimes F_{75}.$$

Using the Prime Factor FFT, Theorem 10, on each of the three tensor factors we obtain

$$\begin{aligned}
\mathcal{F}_n &= Q^{(2^4,3)}(F_{2^4} \otimes F_3)P^{(2^4,3)} \otimes Q^{(2,3^2,5)}(F_2 \otimes F_{3^2} \otimes F_5)P^{(2,3^2,5)} \otimes \\
&\quad Q^{(3,5^2)}(F_3 \otimes F_{5^2})P^{(3,5^2)} \\
&= (Q(2^4,3) \otimes Q^{(2,3^2,5)} \otimes Q^{(3,5^2)}) \\
&\quad ((F_{2^4} \otimes F_3) \otimes (F_2 \otimes F_{3^2} \otimes F_5) \otimes (F_3 \otimes F_{5^2})) \\
&\quad (P^{(2^4,3)} \otimes P^{(2,3^2,5)} \otimes P^{(3,5^2)})
\end{aligned}$$

Using Proposition 8 to rearrange the tensor product of DFT matrices we obtain

$$\begin{aligned}
\mathcal{F}_n &= (Q^{(2^4,3)} \otimes Q^{(2,3,5^2)} \otimes Q^{(3,5^2)})L_{\sigma^{-1}}^{(2,2^4,3,3,3^2,5,5^2)} \\
&\quad ((F_2 \otimes F_{2^4}) \otimes (F_3 \otimes F_3 \otimes F_{3^2}) \otimes (F_5 \otimes F_{5^2})) \\
&\quad L_{\sigma}^{(2^4,3,2,3^2,5,3,5^2)}(P^{(2^4,3)} \otimes P^{(2,3^2,5)} \otimes P^{(3,5^2)}),
\end{aligned}$$

where σ is the permutation $(2,3,1,5,6,4,7)$.

As a second example, consider $\mathcal{F}_n = F_{360} \otimes F_{900}$. Following the same procedure as in the first example, we obtain

$$\begin{aligned}
\mathcal{F} &= (Q^{(2^3,3^2,5)} \otimes Q^{(2^2,3^2,5^2)})L_{\sigma^{-1}}^{(2^2,2^3,3^2,3^2,5,5^2)} \\
&\quad ((F_{2^2} \otimes F_{2^3}) \otimes (F_{3^2} \otimes F_{3^2}) \otimes (F_5 \otimes F_{5^2})) \\
&\quad L_{\sigma}^{(2^3,3^2,5,2^2,3^2,5^2)}(P^{(2^3,3^2,5)} \otimes P^{(2^2,3^2,5^2)}),
\end{aligned}$$

where σ is the permutation $(2,3,5,1,4,6)$.

The dimensionless prime factor algorithm is a generalization of this example.

Theorem 11 (Dimensionless Prime Factor Algorithm) *Let $\mathcal{F}_n$ be an arbitary multidimensional DFT of size $n = n_1 \cdots n_t$ with $n_j = p_j^{e_j}$. Then there exists permutations P and Q such that*

$$\mathcal{F}_n = Q\left(\mathcal{F}_{n_1} \otimes \cdots \otimes \mathcal{F}_{n_t}\right)P = Q\left(\prod_{j=1}^{t} I_{N(j-1)} \otimes \mathcal{F}_{n_j} \otimes I_{\bar{N}(j)}\right)P,$$

where $\mathcal{F}_{n_j}$ is a multidimensional DFT of size n_j.

PROOF Assume that $\mathcal{F}_n = F_{m_1} \otimes \cdots \otimes F_{m_s}$ is an s-dimensional $m_1 \times \cdots \times m_s$ where $n = m_1 \cdots m_s$. Apply the prime factor FFT to F_{m_i} for each i and pull the permutations to the front and back. Next apply the appropriate tensor permutation to collect the DFTs for each prime. ∎

The next example shows how to construct a radix-2 dimensionless FFT analogous to Theorem 7.

Example 5 (Radix 2 dimensionless FFT) *We illustrate the proof and statement of the previous theorem using a two-dimensional 4×4 DFT. This computation provides the twiddle factor matrices and initial permutation given in the example in Section 2.*

$$\begin{aligned}
F_4 \otimes F_4 &= (F_2 \otimes I_2)T_2^4(I_2 \otimes F_2)L_2^4 \otimes (F_2 \otimes I_2)T_2^4(I_2 \otimes F_2)L_2^4 \\
&= \big((F_2 \otimes I_2)T_2^4(I_2 \otimes F_2) \otimes (F_2 \otimes I_2)T_2^4(I_2 \otimes F_2)\big)\,(L_2^4 \otimes L_2^4) \\
&= \big((F_2 \otimes I_2)T_2^4(I_2 \otimes F_2) \otimes I_4)(I_4 \otimes (F_2 \otimes I_2)T_2^4(I_2 \otimes F_2)\big)\,(L_2^4 \otimes L_2^4) \\
&= \big((F_2 \otimes I_8)(T_2^4 \otimes I_4)(I_2 \otimes F_2 \otimes I_2)(I_4 \otimes F_2 \otimes I_2)(I_4 \otimes T_2^4)(I_8 \otimes F_2)\big) \\
&\quad (L_2^4 \otimes L_2^4)
\end{aligned}$$

The radix-p dimensionless FFT is a generalization of this example.

Theorem 12 (Radix p Dimensionless FFT) *Let $N = p^K$, p a prime, and assume that $N = n_1 \times \cdots \times n_t$, where $n_i = p^{k_i}$. Let $K(i) = k_1 + \cdots + k_i$, with the boundary conditions $K(t) = K$ and $K(0) = 0$ denote the i-th prefix sum. Let $\mathcal{F}_N$ denote an arbitrary multi-dimensional DFT of size $N = 2^K$. Let $\mathcal{F}_N = F_{n_1} \otimes \cdots \otimes F_{n_t}$ denote an arbitrary multi-dimensional DFT of size $N = p^K$. Then*

$$\mathcal{F}_N = \left\{ \prod_{i=1}^{K} \left(I_{p^{i-1}} \otimes F_p \otimes I_{p^{K-K(i)}} \right) T_i \right\} R, \tag{22}$$

where

$$R = R_{n_1} \otimes \cdots \otimes R_{n_t}$$

and

$$T_i = I_{p^{K(j-1)+l-1}} \otimes T_{p^{k_j-l}}^{p^{k_j-l+1}} \otimes I_{p^{K-K(j)}}$$

with $i = K(j-1) + l$ and $0 \le l < k_j$.

Proof. The proof follows from the one-dimensional FFT factorization (Theorem 7) and the fundamental factorization of tensor products (Proposition 5).

Using the radix p FFT we can write

$$F_{n_j} = B_{n_j} R_{n_j},$$

where

$$B_{n_j} = \prod_{l=1}^{k_j} (I_{p^{l-1}} \otimes F_p \otimes I_{p^{k_j-l}})(I_{p^{l-1}} \otimes T_{p^{k_j-l}}^{p^{k_j-l+1}}).$$

Then

$$\mathcal{F}_N = (B_{n_1} \otimes \cdots \otimes B_{n_t})(R_{n_1} \otimes \cdots \otimes R_{n_t})$$

Furthermore,

$$(B_{n_1} \otimes \cdots \otimes B_{n_t}) = \prod_{j=1}^{t} (I_{p^{K(j-1)}} \otimes B_{n_j} \otimes I_{p^{K-K(j)}})$$

If we expand each B_{n_j}

$$\mathcal{F}_N = \left\{ \prod_{j=1}^{t} \prod_{l=1}^{k_j} (I_{p^{K(j-1)}} \otimes I_{p^{l-1}} \otimes F_p \otimes I_{p^{k_j-l}} \otimes I_{p^{K-K(j)}})(I_{2^{K(j-1)}} \otimes I_{p^{l-1}} \otimes T_{p^{k_j-l}}^{p^{k_j-l+1}} \otimes I_{p^{K-K(j)}}) \right\} R.$$

Combining adjacent identity matrices

$$\mathcal{F}_N = \left\{ \prod_{j=1}^{t} \prod_{l=1}^{k_j} (I_{p^{K(j-1)+l-1}} \otimes F_p \otimes I_{p^{k_j-l+K-K(j)}})(I_{p^{K(j-1)+l-1}} \otimes T_{p^{k_j-l}}^{p^{k_j-l+1}} \otimes I_{p^{K-K(j)}}) \right\} R.$$

Setting $i = k(j-1) + l$ into this equation completes the proof. ∎

Note that if the radix was not prime, the proof would not work in general, since it may not be possible combine adjacent factors to obtain the desired radix.

The multi-radix version of the prime-power dimensionless FFT is slightly more complicated than the radix p version. The following example illustrates the new complication.

Example 6 (Multi-Radix Dimensionless FFT) *This example shows how to compute the two-dimensional 2×8 DFT using a radix four factorization.*

$$\begin{aligned}
F_2 \otimes F_8 &= F_2 \otimes (F_2 \otimes I_4) T_4^8 (I_2 \otimes F_4) L_2^8 \\
&= \left(F_2 \otimes (F_2 \otimes I_4) T_4^8 (I_2 \otimes F_4)\right)(I_2 \otimes L_2^8) \\
&= \left((F_2 \otimes I_8)(I_2 \otimes (F_2 \otimes I_4) T_4^8 (I_2 \otimes F_4)\right)(I_2 \otimes L_2^8) \\
&= \left((F_2 \otimes I_8)(I_2 \otimes F_2 \otimes I_4)(I_2 \otimes T_4^8)(I_4 \otimes F_4)\right)(I_2 \otimes L_2^8) \\
&= \left(((F_2 \otimes F_2) \otimes I_4)(I_2 \otimes T_4^8)(I_4 \otimes F_4)\right)(I_2 \otimes L_2^8)
\end{aligned}$$

Observe that the factorization requires both the two-dimensional 2×2 and the one-dimensional 4-point DFTs. In order to obtain the 4-point DFTs it was necessary to split F_8 into F_2 and F_4 and then to combine the F_2 with the previous F_2. The necessary splitting and combining process is conveniently described using the greatest common refinement of the ordered partitions $[1,3]$ and $[2,2]$ representing the dimension and radix specifications. The partitions are obtained from the base two exponents of the dimensions and radices. The greatest common refinement is equal to the partition $[1,1,2]$. Figure 4 shows the partition $[1,1,2]$ as a refinement of the dimension and radix partitions. The refinement of the dimension partition indicates the necessary applications of the Cooley-Tukey theorem, and the refinement of the radix partition shows the necessary groupings to obtain DFTs of the specified sizes.

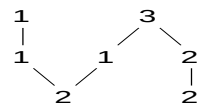

Figure 4: Partition Refinement for Radix 4 Computation of $F_2 \otimes F_8$

The previous example indicates how to construct a formula for a given multidimensional DFT using a specified multi-radix computation. The formula can be obtained from the greatest common refinement of the dimension and radix partitions. To make the notion of the greatest common refinement precise we define the greatest common refinement. If $\pi = [k_1, \ldots, k_t]$, then the i-th prefix sum, $\pi(i) = k_1 + \cdots + k_i$. Given two ordered partitions, π_1 and π_2 or the positive integer n, π_1 is a refinement of π_2 if for any prefix sum of π_2 there exist a prefix sum of π_1 equal to the prefix sum of π_2:

$$\forall i \exists j \pi_1(j) = \pi_2(i).$$

The set of ordered partitions of n form a partially ordered set under refinement. A greatest common refinement $\pi_1 \wedge \pi_2$ of two partitions π_1 and π_2 is a partition that is a refinement of both partitions with the property that if any other partition is a common refinement of π_1 and π_2 then it is a refinement of $\pi_1 \wedge \pi_2$.

Let π' denote the partition of $n - \pi(1)$ obtained from $\pi = [k_1, \ldots, k_t]$ be removing k_1. Assume that $\pi_1(1) \leq \pi_2(1)$. Using an inductive argument it is easy to show that

$$\pi_1 \wedge \pi_2 = \pi_1(1), \pi_1' \wedge \pi_2(1) - \pi_1(1), \pi_2'.$$

Example 7 (Partition Calculation and Multi-Radix Dimensionless FFT) *In this example, we show how to read off the multi-radix factorization of a multi-dimensional DFT using the greatest common refinement of the dimension and radix partitions. Let $\mathcal{F}_{2^{21}}$ be the three-dimensional DFT $F_{2^7} \otimes F_{2^7} \otimes F_{2^7}$ and suppose we wish to factor $\mathcal{F}_{2^{21}}$ using a multi-radix dimensionless FFT with radices $2^5, 2^5, 2^5, 2^5, 2$. The factorization can be obtained from the greatest common refinement of the partitions $[7, 7, 7]$ and $[5, 5, 5, 5, 1]$ of 21. The greatest common refinement is equal to $[5, 2, 3, 4, 1, 5, 1]$ (see Figure 5). The corresponding factorization is equal to*

$$\begin{aligned}
\mathcal{F}_{2^{21}} \quad = \quad & (F_{2^5} \otimes I_{2^{16}}) T_1 \\
& (I_{2^5} \otimes (F_{2^2} \otimes F_{2^3}) \otimes I_{2^{11}}) T_2 \\
& (I_{2^{10}} \otimes (F_{2^4} \otimes F_{2^1}) \otimes I_{2^6}) T_3 \\
& (I_{2^{15}} \otimes F_{2^5} \otimes I_2) T_4 \\
& (I_{2^{20}} \otimes F_2) T_5 \\
& R,
\end{aligned}$$

where

$$\begin{aligned}
T_1 &= T_{2^2}^{2^7} \otimes I_{2^{14}} \\
T_2 &= I_{2^7} \otimes T_{2^4}^{2^7} \otimes I_{2^7} \\
T_3 &= I_{2^{14}} \otimes T_{2^6}^{2^7} \\
T_4 &= I_{2^{14}} \otimes I_2 \otimes T_2^{2^6} = I_{2^{15}} \otimes T_2^{2^6} \\
T_5 &= I_{2^{21}} \\
R &= R^{(2^5,2^2)} \otimes R^{(2^3,2^4)} \otimes R^{(2,2^5,2)}.
\end{aligned}$$

Observe that the initial permutation, R is obtained from viewing the greatest common refinement as a refinement of the dimension partition. The particular DFTs required to compute the specified multi-radix FFT are determined from vieweing the greatest common refinement as a refinement of the radix partition. The twiddle matrices are obtained from the splits needed to refine the dimension partition.

As a second example we consider the 4-dimensional $2^3 \times 2^3 \times 2^3 \times 2^3$ DFT using a multi-radix FFT with radices 2^5, 2^5, 2^2. The factorization can be obtained from the greatest common refinement of the partitions $[3,3,3,3]$ and $[5,5,2]$ of 12. The greatest common refinement is equal to $[3,2,1,3,1,2]$ (see Figure 5). The corresponding factorization is equal to

$$\begin{aligned}
\mathcal{F}_{2^{12}} &= ((F_{2^3} \otimes F_{2^2}) \otimes I_{2^7})T_1 \\
&\quad (I_{2^5} \otimes (F_2 \otimes F_{2^3} \otimes F_2) \otimes I_{2^2})T_2 \\
&\quad (I_{2^{10}} \otimes F_2) \\
&\quad R,
\end{aligned}$$

where

$$\begin{aligned}
T_1 &= I_{2^3} \otimes T_2^{2^3} \otimes I_{2^6} \\
T_2 &= I_{2^9} \otimes T_{2^2}^{2^3} \\
R &= I_{2^3} \otimes R^{(2^2,2)} \otimes I_{2^3} \otimes R^{(2,2^2)}.
\end{aligned}$$

In this example, we see a grouping of 3 DFTs in order to obtain the second radix 2^5 computation. Observe that there can be no non-trivial twiddle factors between the required grouping.

The multi-radix dimensionless FFT generalizes these two examples. To simplify the statement of the theorem and the proof, multiplicative factorizations of the size are used rather than partitions of the exponents. Since the size is a prime power, it is possible to go back and forth between these two notions. Consequently the concept of greatest common refinement carries over to multiplicative factorizations.

Theorem 13 (Multi-Radix Dimensionless FFT) *Let $N = p^K$, p a prime, and assume that $N = n_1 \times \cdots \times n_t$, where $n_j = p^{k_j}$. Let $\mathcal{N}$ denote the list*

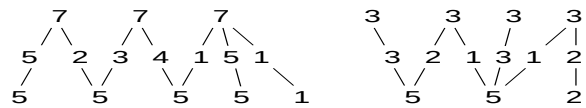

Figure 5: Partition Refinements for Radix $2^5, 2^5, 2^5, 2^5, 2$ Computation of $F_{2^7} \otimes F_{2^7} \otimes F_{2^7}$ and Radix $2^5, 2^5, 2^2$ Computation of $F_{2^3} \otimes F_{2^3} \otimes F_{2^3} \otimes F_{2^3}$

of factors $[n_1, \ldots, n_t]$. *Let* $\mathcal{F}_N = F_{n_1} \otimes \cdots \otimes F_{n_t}$ *denote an arbitrary multi-dimensional DFT of size* N. *Let* $N(j) = n_1 \times \cdots \times n_j$, *with boundary conditions* $N(0) = 1$ *and* $N(t) = n$. *Let* $\bar{N}(j) = n/N(j)$.

Let $\mathcal{M} = [m_1, \ldots, m_s]$ *with* $m_i = p^{l_i}$ *be a second factorization of* N *called the radix factorization with corresponding notation for prefix products.*

Let $\mathcal{M}' = [m_{1,1}, \ldots, m_{1,s_1}, \ldots, m_{s,1}, \ldots, m_{s,s_s}]$ *and* $\mathcal{N}' = [n_{1,1}, \ldots, n_{1,t_1}, \ldots, n_{t,1}, \ldots, n_{t,t_t}]$ *be refinements of* $\mathcal{M}$ *and* $\mathcal{N}$ *respectively with both equal to the greatest common refinement of* $\mathcal{M}$ *and* $\mathcal{N}$. *Following the previous notation for prefix products, let* $N_j(c) = n_{j,1} \cdots n_{j,c}$ *and similarly for* $M_i(c)$.

Then

$$\mathcal{F}_n = \left\{ \prod_{i=1}^{s} \left(I_{M(i-1)} \otimes \mathcal{F}_{m_i} \otimes I_{\overline{M}(i)} \right) T_i \right\} R, \tag{23}$$

where

$$R = R^{(n_{1,1},\ldots,n_{1,t_1})} \otimes \cdots \otimes R^{(n_{t,1},\ldots,n_{t,t_t})},$$

$$\mathcal{F}_{m_i} = F_{m_{i,1}} \otimes \cdots \otimes F_{m_{i,t_i}}$$

and

$$T_i = I_{N(j-1)N_j(c-1)} \otimes T^{n_{j,c}\bar{N}_j(c)}_{\bar{N}_j(c)} \otimes I_{\bar{N}(j)}$$

where $M(i) = N(j-1)N_j(c)$.

Proof

$$\mathcal{F}_n = \prod_{j=1}^{t} (I_{N(j-1)} \otimes F_{n_j} \otimes I_{\bar{N}(j)}$$

Substituting the mixed-radix Cooley-Tukey factorization into this equation

$$\mathcal{F}_n = \prod_{j=1}^{t} (I_{N(j-1)} \otimes \left(\prod_{c=1}^{t_j} (I_{N_j(c-1)} \otimes F_{n_{j,c}} \otimes I_{\bar{N}_j(c)})(I_{N_j(c-1)} \otimes T^{n_{j,c}\bar{N}_j(c)}_{\bar{N}_j(c)}) R^{(n_{j,1},\ldots,n_{j,t_j})} \right) \otimes I_{\bar{N}(j)}.$$

Write the product as a double product and pull the digit-reversal permutations to the front.

$$\mathcal{F}_n \quad = \quad \left(\prod_{j=1}^{t} \prod_{c=1}^{t_j} (I_{N(j-1)} \otimes I_{N_j(c-1)} \otimes F_{n_{j,c}} \otimes I_{\bar{N}_j(c)} \otimes I_{\bar{N}(j)})(I_{N(j-1)} \otimes I_{N_j(c-1)} \otimes T^{n_{j,c}\bar{N}_j(c)}_{\bar{N}_j(c)}) \right)$$

$$
\begin{aligned}
& R^{(n_{1,1},\ldots,n_{1,t_1})} \otimes \cdots \otimes R^{(n_{t,1},\ldots,n_{t,t_t})} \\
= & \left(\prod_{j=1}^{t} \prod_{c=1}^{t_j} (I_{N(j-1)N_j(c-1)} \otimes F_{n_{j,c}} \otimes I_{\bar{N}_j(c)\bar{N}(j)})(I_{N(j-1)N_j(c-1)} \otimes T^{n_{j,c}\bar{N}_j(c)}_{\bar{N}_j(c)}) \right) R
\end{aligned}
$$

Since the $n_{j,c}$'s and the $m_{i,c}$'s are a common refinement for $\mathcal{N}$ and $\mathcal{M}$ we can rewrite this nested product using the refinement of $\mathcal{M}$ instead of the refinement of $\mathcal{N}$.

$$
\mathcal{F}_n = \left(\prod_{i=1}^{s} \prod_{c=1}^{s_i} (I_{M(i-1)M_i(c-1)} \otimes F_{m_{i,c}} \otimes I_{\bar{M}_i(c)\bar{M}(i)}) T_{i,c} \right) R,
$$

where $T_{i,c}$ is one of the twiddles in the product indexed by the refinement of N. Since the refinement is the greatest common refinement $T_{i,c} = I$ for $c < s_i$. This implies we can collect the factors from the inner product.

$$
\mathcal{F}_n = \left(\prod_{i=1}^{s} (I_{M(i-1)} \otimes (\mathcal{F}_{m_i} \otimes I_{\bar{M}(i)}) T_i \right) R
$$

where $\mathcal{F}_{m_i} = F_{m_{i,1}} \otimes \cdots \otimes F_{m_{i,s_i}}$ and $T_i = T_{i,s_i}$. The last step of the proof is to determine j and c such that $T_i = I_{N(j-1)N_j(c-1)} \otimes T^{n_{j,c}\bar{N}_j(c)}_{\bar{N}_j(c)}$. This is done by setting $M(i) = N(j-1)N_j(c)$. ∎

5 Conclusion

This paper presented a dimensionless FFT algorithm for computing multidimensional DFTs. For a fixed number of data points, a dimensionless FFT has the same dataflow patterns independent of the dimension. In this paper a particular dataflow pattern was used; however, we note that using the techniques of [8] it is possible to derive alternative dimensionless FFTs with different dataflow patterns. Since dataflow is the prodominant cost in FFT computations on modern computer architectures, optimizing a dataflow across dimensions provides simultaneously fast implementations of 1, 2, and 3 dimensional FFTs [7].

To configure a dimensionless algorithm to a particular dimension it is necessary to provide twiddle factors and an initial and final ordering of the data that are determined by the dimension. Section 4 provided explicit formulas to determine the necessary twiddle factors and relabelings. Given this information it is possible to produce a universal FFT processor that has a hardwired dataflow and is capable of computing different dimension DFTs (e.g. both a 1-dimensional and 2-dimensional DFT) [6, 9].

References

[1] L. Auslander, J. R. Johnson, and R. W. Johnson. Multidimensional Cooley-Tukey algorithms revisited. *Adv. Appl. Math.*, 19(9):297–301, April 1996.

[2] James W. Cooley and J. W. Tukey. An algorithm for the machine calculation of complex Fourier series. *Math. Comput.*, 19(90):297–301, April 1965.

[3] I. J. Good. The relationship between two fast Fourier transforms. *IEEE Trans. Comput.*, C-20(3):310–317, March 1971.

[4] N. Jacobson. *Basic Algebra I.* W. H. Freeman and Company, San Francisco, 1974.

[5] J. R. Johnson, J. Dwyer, and R. W. Johnson. Fast fourier transform algorithms for finite abelian groups. Technical Report DU-MCS-95-01, Drexel University, 1995.

[6] J. R. Johnson, J. Dwyer, and R. W. Johnson. FFT machine and documentation. Technical Report DU-MCS-97-02, Drexel University, 1997.

[7] J. R. Johnson and R. W. Johnson. Distributed memory fft algorithms and dataflow. In *Proc. of High Performance Embedded Computing (HPEC '99)*, MIT Lincoln Laboratory, September 1999.

[8] J. R. Johnson, R. W. Johnson, D. Rodriguez, and R. Tolimieri. A methodology for designing, modifying, and implementing Fourier transform algorithms on various architectures. *Circuits, Systems, and Signal Processing*, 9(4):449–500, 1990.

[9] P. Kumhom, J. R. Johnson, and P. Nagvajara. Design, optimization, and implementation of a universal fft processor. In *13th IEEE International ASIC/SOC Conference*, Washington, DC, September 2000.

[10] L. Thomas. Using a computer to solve problems in physics. In *Applications of Digital Computers*. Ginn, Boston, 1963.

[11] R. Tolimieri, M. An, and C. Lu. *Algorithms for Discrete Fourier Transform and Convolution*. Springer, Berlin Heidelberg New York Tokyo, 1989.

[12] R. Tolimieri, M. An, and C. Lu. *Mathematics of Multidimensional Fourier Transform Algorithms*. Springer, Berlin Heidelberg New York Tokyo, 1993.

[13] C. Van Loan. *Computational Frameworks for the Fast Fourier Transform*, volume 10 of *Frontiers in Applied Mathematics*. Society for Industrial and Applied Mathematics, Philadelphia, 1992.