The Cooley-Tukey FFT and Group Theory

David K. Maslen and Daniel N. Rockmore

Pure and Applied Mathematics—Two Sides of a Coin

In November of 1979 there appeared in the *Bulletin of the AMS* a paper by L. Auslander and R. Tolimieri [3] with the delightful title "Is Computing with the Finite Fourier Transform Pure or Applied Mathematics?" This rhetorical question was answered by showing that in fact the finite Fourier transform and the family of efficient algorithms used to compute it (the Fast Fourier Transform (FFT), a pillar of the world of digital signal processing) are of interest to both pure and applied mathematicians.

Auslander had come of age as an applied mathematician at a time when pure and applied mathematicians still received much the same training. The ends towards which these skills were then directed became a matter of taste. As Tolimieri retells it,[1] Auslander had become distressed at the development of a separate discipline of applied mathematics which had grown apart from much of core mathematics. The effect of this development was detrimental to both sides. On the one hand, applied mathematicians had fewer tools to bring to problems, and, conversely, pure mathematicians were often ignoring the fertile bed of inspiration provided by real-world problems. Auslander hoped their paper would help mend a growing perceived rift in the mathematical community by showing the ultimate unity of pure and applied mathematics.

We will show that investigation of finite and fast Fourier transforms continues to be a varied and interesting direction of mathematical research. Whereas Auslander and Tolimieri concentrated on relations to nilpotent harmonic analysis and theta functions, we emphasize connections between the famous *Cooley-Tukey FFT* and group representation theory. In this way we hope to provide further evidence of the rich interplay of ideas which can be found at the nexus of pure and applied mathematics.

David K. Maslen is a mathematician at Susquehanna International Group LLP. His e-mail address is david@maslen.net.

Daniel N. Rockmore is professor of mathematics and computer science at Dartmouth College and on the external faculty of the Santa Fe Institute. His e-mail address is rockmore@cs.dartmouth.edu. *He is supported in part by NSF PFF Award DMS-9553134, AFOSR F49620-00-1-0280, and DOJ 2000-DT-CX-K001. He would also like to thank the Santa Fe Institute and the Courant Institute for their hospitality during some of the writing. Pieces of the introduction are similar to his paper "The FFT—an algorithm the whole family can use", which appeared in* Computing in Science & Engineering*, January 2000, pp. 62–67.*

[1] *Private communication.*

Background

The finite Fourier transform or discrete Fourier transform (DFT) has several representation theoretic interpretations: either as an exact computation of the Fourier coefficients of a function on the cyclic group $\mathbf{Z}/n\mathbf{Z}$ or a function of band-limit n on the circle S^1, or as an approximation to the Fourier transform of a function on the real line. For each of these points of view there is a natural group-theoretic generalization and also a corresponding set of efficient algorithms for computing the quantities involved. These algorithms collectively make up the *Fast Fourier Transform* or *FFT*.

Formally, the DFT is a linear transformation mapping any complex vector of length n, $f = (f(0), \ldots, f(n-1))^t \in \mathbf{C}^n$, to its *Fourier transform*, $\hat{f} \in \mathbf{C}^n$. The kth component of $\hat{f}$, *the DFT of f at frequency k*, is

$$\hat{f}(k) = \sum_{j=0}^{n-1} f(j) e^{2\pi i jk/n}, \tag{1}$$

where $i = \sqrt{-1}$, and the *inverse Fourier transform* is

$$f(j) = \frac{1}{n} \sum_{k=0}^{n-1} \hat{f}(k) e^{-2\pi i jk/n}. \tag{2}$$

Thus, with respect to the standard basis, the DFT can be expressed as the matrix-vector product $\hat{f} = \mathcal{F}_n \cdot f$, where $\mathcal{F}_n$ is the *Fourier matrix* of order n whose j,k entry is equal to $e^{2\pi ijk/n}$. Computing a DFT directly would require n^2 scalar operations.[2] Instead, the FFT is a family of algorithms for computing the DFT of any $f \in \mathbf{C}^n$ in $O(n \log n)$ operations. Since inversion can be framed as the DFT of the function $\check{f}(k) = \frac{1}{n}\hat{f}(-k)$, the FFT also gives an efficient inverse Fourier transform.

One of the main practical implications of the FFT is that it allows any cyclically invariant linear operator to be applied to a vector in only $O(n \log n)$ scalar operations. Indeed, the DFT diagonalizes any cyclic group-invariant operator, making possible the following algorithm: (1) Compute the Fourier transform (DFT). (2) Multiply the DFT by the eigenvalues of the operator, which are also found using the Fourier transform. (3) Compute the inverse Fourier transform of the result. This technique is the basis of efficient digital filter (i.e., convolution) and is also used for the efficient numerical solution of partial differential equations.

Some History

Since the Fourier matrix is effectively the character table of a cyclic group, it is not surprising that some of its earliest appearances are in number theory, the subject which gave birth to character theory. Consideration of the Fourier matrix goes back at least as far as to Gauss, who was interested in its connections to quadratic reciprocity. In particular, Gauss showed that for odd primes p and q,

$$\left(\frac{p}{q}\right)\left(\frac{q}{p}\right) = \frac{\mathrm{Trace}(\mathcal{F}_{pq})}{\mathrm{Trace}(\mathcal{F}_p)\mathrm{Trace}(\mathcal{F}_q)}, \tag{3}$$

where $\left(\frac{p}{q}\right)$ denotes the Legendre symbol. Gauss also established a formula for the quadratic Gauss sum $\mathrm{Trace}(\mathcal{F}_n)$, which is discussed in detail in [3].

Another early appearance of the DFT occurs in the origins of representation theory in the work of Dedekind and Frobenius on the group determinant. For a finite group G, the group determinant Θ_G is defined as the homogeneous polynomial in the variables x_g (for each $g \in G$) given by the determinant of the matrix whose rows and columns are indexed by the elements of G with g,h-entry equal to $x_{gh^{-1}}$. Frobenius showed that when G is abelian, Θ_G admits the factorization

$$\Theta_G = \prod_{\chi \in \hat{G}} \left(\sum_{g \in G} \chi(g) x_g \right), \tag{4}$$

where $\hat{G}$ is the set of characters of G. The linear form defined by the inner sum in (4) is a "generic" DFT at the frequency χ.

[2] *At this point we must come clean about how we count operations. Our count is either the number of complex additions or the number of complex multiplications, whichever is greater.*

In the nonabelian case, Θ_G admits an analogous factorization in terms of irreducible polynomials of the form

$$\Theta_D(G) = \det\left(\sum_{g \in G} D(g) x_g \right),$$

where D is an *irreducible matrix representation* of G. The inner sum here is a generic Fourier transform over G. See [12] for a beautiful historical exposition of these ideas.

Gauss's interests ranged over all areas of mathematics and its applications, so it is perhaps not surprising that the first appearance of an FFT can also be traced back to him [10]. Gauss was interested in certain astronomical calculations, a recurrent area of application of the FFT, needed for interpolation of asteroidal orbits from a finite set of equally spaced observations. Surely the prospect of a huge laborious hand calculation was good motivation for the development of a fast algorithm. Making fewer hand calculations also implies less opportunity for error and hence increased numerical stability!

Gauss wanted to compute the Fourier coefficients a_k and b_k of a function represented by a Fourier series of bandwidth n,

$$f(x) = \sum_{k=0}^{m} a_k \cos 2\pi kx + \sum_{k=1}^{m} b_k \sin 2\pi kx, \tag{5}$$

where $m = (n-1)/2$ for n odd and $m = n/2$ for n even. He first observed that the Fourier coefficients can be computed by a DFT of length n using the values of f at equispaced sample points. Gauss then went on to show that if $n = n_1 n_2$, this DFT can in turn be reduced to first computing n_1 DFTs of length n_2, using equispaced subsets of the sample points, i.e., a subsampled DFT, and then combining these shorter DFTs using various trigonometric identities. This is the basic idea underlying the *Cooley-Tukey FFT*.

Unfortunately, this reduction never appeared outside of Gauss's collected works. Similar ideas, usually for the case $n_1 = 2$, were rediscovered intermittently over the succeeding years. Notable among these is the doubling trick of Danielson and Lanczos (1942), performed in the service of x-ray crystallography, another frequent employer of FFT technology. Nevertheless, it was not until the publication of Cooley and Tukey's famous paper [7] that the algorithm gained any notice. The story of Cooley and Tukey's collaboration is an interesting one. Tukey arrived at the basic reduction while in a meeting of President Kennedy's Science Advisory Committee, where among the topics of discussions were techniques for offshore detection of nuclear tests in the Soviet Union. Ratification of a proposed United States/Soviet Union nuclear test ban depended upon the development of a method for detecting the tests without actually visiting the Soviet nuclear facilities. One idea required the

analysis of seismological time series obtained from offshore seismometers, the length and number of which would require fast algorithms for computing the DFT. Other possible applications to national security included the long-range acoustic detection of nuclear submarines.

R. Garwin of IBM was another of the participants at this meeting, and when Tukey showed him this idea, Garwin immediately saw a wide range of potential applicability and quickly set to getting this algorithm implemented. Garwin was directed to Cooley, and, needing to hide the national security issues, told Cooley that he wanted the code for another problem of interest: the determination of the periodicities of the spin orientations in a 3-D crystal of He^3. Cooley had other projects going on, and only after quite a lot of prodding did he sit down to program the "Cooley-Tukey" FFT. In short order Cooley and Tukey prepared their paper, which, for a mathematics/computer science paper, was published almost instantaneously—in six months! This publication, Garwin's fervent proselytizing, as well as the new flood of data available from recently developed fast analog-to-digital converters, did much to help call attention to the existence of this apparently new fast and useful algorithm. In fact, the significance of and interest in the FFT was such that it is sometimes thought of as having given birth to the modern field of analysis of algorithms. See also [6] and the 1967 and 1969 special issues of the *IEEE Transactions in Audio Electronics* for more historical details.

The Fourier Transform and Finite Groups

One natural group theoretic interpretation of the Fourier transform is as a change of basis in the space of complex functions on $\mathbf{Z}/n\mathbf{Z}$. Given a complex function f on $\mathbf{Z}/n\mathbf{Z}$, we may expand f in the basis of irreducible characters $\{\chi_k\}$ defined by $\chi_k(j) = e^{2\pi ijk/n}$. By (2) the coefficient of χ_k in the expansion is equal to the scaled Fourier coefficient $\frac{1}{n}\hat{f}(-k)$, whereas the Fourier coefficient $\hat{f}(k)$ is the inner product of the vector of function values of f with those of the character χ_k.

For an arbitrary finite group G there is an analogous definition. The characters of $\mathbf{Z}/n\mathbf{Z}$ are the simplest example of a *matrix representation*, which for any group G is a matrix-valued function $\rho(g)$ on G such that $\rho(ab) = \rho(a)\rho(b)$ and $\rho(e)$ is the identity matrix. Given a matrix representation ρ of dimension d_ρ and a complex function f on G, the *Fourier transform of f at ρ* is defined as the matrix sum

$$\hat{f}(\rho) = \sum_{x\in G} f(x)\rho(x). \tag{6}$$

Computing $\hat{f}(\rho)$ is equivalent to the computation of the d_ρ^2 scalar Fourier transforms at each of the individual *matrix elements* ρ_{ij},

$$\hat{f}(\rho_{ij}) = \sum_{x\in G} f(x)\rho_{ij}(x). \tag{7}$$

A set of matrix representations $\mathcal{R}$ of G is called *a complete set of irreducible representations* if and only if the collection of matrix elements of the representations, relative to an arbitrary choice of basis for each matrix representation in the set, forms a basis for the space of complex functions on G. The Fourier transform of f with respect to $\mathcal{R}$ is then defined as the collection of individual transforms, while *the Fourier transform on G* means any Fourier transform computed with respect to some complete set of irreducibles. In this case, the inverse transform is given explicitly as

$$f(x) = \frac{1}{|G|}\sum_{\rho\in\mathcal{R}} d_\rho \mathrm{Trace}(\hat{f}(\rho)\rho(x^{-1})). \tag{8}$$

Equation (8) shows us a relation between the group Fourier transform and the expansion of a function in the basis of matrix elements. The coefficient of ρ_{ij} in the expansion of f is the Fourier transform of f at the dual representation $[\rho_{ji}(g^{-1})]$ scaled by the factor $d_\rho/|G|$.

Viewing the Fourier transform on G as a simple matrix-vector multiplication leads to some simple bounds on the number of operations required to compute the transform. The computation clearly takes no more than the $|G|^2$ scalar operations required for any matrix-vector multiplication. On the other hand, the column of the Fourier matrix corresponding to the trivial representation is all 1s, so at least $|G| - 1$ additions are necessary. One main goal of this finite group FFT research is to discover algorithms which can significantly reduce the upper bound for various classes of groups or even all finite groups.

The Current State of Affairs for Finite Group FFTs

Analysis of the Fourier transform shows that for G abelian, the number of operations required is bounded by $O(|G|\log|G|)$. For arbitrary groups G, upper bounds of $O(|G|\log|G|)$ remain the holy grail in group FFT research. In 1978 A. Willsky provided the first nonabelian example by showing that certain metabelian groups have an $O(|G|\log|G|)$ Fourier transform algorithm [20]. Implicit in the big-O notation is the idea that a family of groups is under consideration, with the size of the individual groups going to infinity.

Since Willsky's initial discovery, much progress has been made. U. Baum has shown that the supersolvable groups admit an $O(|G|\log|G|)$ FFT, while others have shown that symmetric groups admit $O(|G|\log^2|G|)$ FFTs (see the section below on symmetric groups). Other groups for which highly improved (but not $O(|G|\log^c|G|)$) algorithms have been discovered include the matrix groups over finite fields and, more generally, the Lie groups of finite type. See [15] for pointers to

the literature. There is much work to be done finding new classes of groups which admit fast transforms and improving on the above results. The ultimate goal is to settle or make progress on the following conjecture.

Conjecture. There exist constants c_1 and c_2 such that for any finite group G there is a complete set of irreducible matrix representations for which the Fourier transform of any complex function on G may be computed in fewer than $c_1|G|\log^{c_2}|G|$ scalar operations.

The Cooley-Tukey Algorithm

Cooley and Tukey showed [7] how the Fourier transform on the cyclic group $\mathbf{Z}/n\mathbf{Z}$, where $n = pq$ is composite, can be written in terms of Fourier transforms on the subgroup $q\mathbf{Z}/n\mathbf{Z} \cong \mathbf{Z}/p\mathbf{Z}$. The trick is to change variables so that the one-dimensional formula (1) is turned into a two-dimensional formula which can be computed in two stages. Define variables j_1, j_2, k_1, k_2 through the equations

$$j = j(j_1, j_2) = j_1 q + j_2, \quad 0 \le j_1 < p, \quad 0 \le j_2 < q; \tag{9}$$

$$k = k(k_1, k_2) = k_2 p + k_1, \quad 0 \le k_1 < p, \quad 0 \le k_2 < q. \tag{10}$$

It follows from (9) and (10) that (1) can be rewritten as

$$\hat{f}(k_1, k_2) = \sum_{j_2=0}^{q-1} e^{2\pi i j_2 (k_2 p + k_1)/n} \sum_{j_1=0}^{p-1} e^{2\pi i j_1 k_1 / p} f(j_1, j_2). \tag{11}$$

We now compute $\hat{f}$ in two stages:

- Stage 1: For each k_1 and j_2 compute the inner sum

$$\tilde{f}(k_1, j_2) = \sum_{j_1=0}^{p-1} e^{2\pi i j_1 k_1 / p} f(j_1, j_2). \tag{12}$$

This requires at most $p^2 q$ scalar operations.

- Stage 2: For each k_1 and k_2 compute the outer sum

$$\hat{f}(k_1, k_2) = \sum_{j_2=0}^{q-1} e^{2\pi i j_2 (k_2 p + k_1)/n} \tilde{f}(k_1, j_2). \tag{13}$$

This requires an additional $q^2 p$ operations. Thus, instead of $(pq)^2$ operations, the above algorithm uses $(pq)(p+q)$ operations.

Stage 1 has the form of a DFT on the subgroup $q\mathbf{Z}/n\mathbf{Z} \cong \mathbf{Z}/p\mathbf{Z}$, embedded as the set of multiples of q, whereas Stage 2 has the form of a DFT on a cyclic group of order q. So if n could be factored further, we could apply the same trick to these DFTs in turn. Thus, if N has the prime factorization $N = p_1 \cdots p_m$, then we reobtain Cooley and Tukey's original m-stage algorithm, which requires $N \sum_i p_i$ operations [7].

A Group Theoretic Interpretation

Auslander and Tolimieri's paper [3] related the Cooley-Tukey algorithm to the Weil-Brezin map for the finite Heisenberg group. Here we present an alternate group theoretic interpretation, originally due to Beth [4], that is more amenable to generalization.

The change of variables (9) may be interpreted as the factorization of the group element j as the (group) product of $j_1 q \in q\mathbf{Z}/n\mathbf{Z}$ with the coset representative j_2. Thus, if we write $G = \mathbf{Z}/n\mathbf{Z}$, $H = q\mathbf{Z}/n\mathbf{Z}$, and let Y denote our set of coset representatives, (9) can be rewritten as

$$g = y \cdot h, \quad y \in Y, \quad h \in H. \tag{14}$$

The second change of variables (10) can be interpreted using the notion of restriction of representations. It is easy to see that restricting a representation on a group G to a subgroup H yields a representation of that subgroup. In the case of $q\mathbf{Z}/n\mathbf{Z}$ this amounts to the observation that

$$e^{2\pi i j_1 q (k_2 p + k_1)/n} = e^{2\pi i j_1 k_1 / p},$$

which is used to prove (11).

The restriction relations between representations may be represented diagrammatically using a directed graded graph with three levels. At level zero there is a single vertex labeled 1, called the root vertex. The vertices at level one are labeled by the irreducible representations of $\mathbf{Z}/p\mathbf{Z}$, and the vertices at level two are labeled by the irreducible representations of $\mathbf{Z}/n\mathbf{Z}$. Edges are drawn from the root vertex to each of the vertices at level one, and from a vertex at level one to a vertex at level two if and only if the representation at the tip restricts to the representation at the tail. The directed graph obtained is the *Bratteli diagram* for the chain of subgroups $\mathbf{Z}/n\mathbf{Z} > \mathbf{Z}/p\mathbf{Z} > 1$. Figure 1 shows the situation for the chain $\mathbf{Z}/6\mathbf{Z} > 2\mathbf{Z}/6\mathbf{Z} \cong \mathbf{Z}/3\mathbf{Z} > 1$.

In this way the irreducible representations of $\mathbf{Z}/n\mathbf{Z}$ are indexed by paths (k_1, k_2) in the Bratteli diagram for $\mathbf{Z}/n\mathbf{Z} > \mathbf{Z}/p\mathbf{Z} > 1$. The DFT factorization (11) now becomes

$$\hat{f}(k_1, k_2) = \sum_{y \in Y} \chi_{k_1, k_2}(y) \sum_{h \in H} f(y \cdot h) \chi_{k_1}(h). \tag{15}$$

The two-stage algorithm is now restated as first computing a set of sums that depend on only the first leg of the paths and then combining these to compute the final sums that depend on the full paths.

In summary, the group elements have been indexed according to a particular factorization scheme, while the irreducible representations (the dual group) are now indexed by paths in a Bratteli diagram, describing the restriction of representations. This

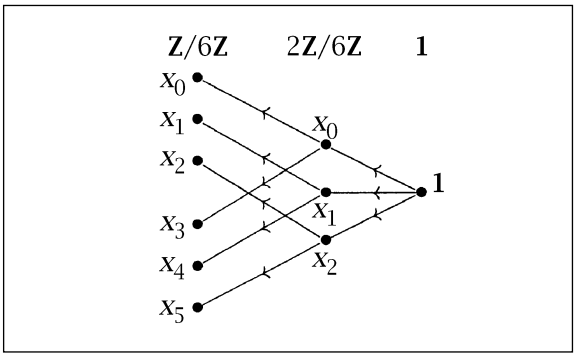


Figure 1. The Bratteli diagram for $\mathbf{Z}/6\mathbf{Z} > 2\mathbf{Z}/6\mathbf{Z} > 1$. The representation χ_k of $\mathbf{Z}/m\mathbf{Z}$ is defined by $\chi_k(l) = e^{2\pi ikl/m}$.

allows us to compute the Fourier transform in stages, using one fewer group element factor at each stage but using paths of increasing length in the Bratteli diagram.

Fast Fourier Transforms on Symmetric Groups

A fair amount of attention has been devoted to developing efficient Fourier transform algorithms for the symmetric group. One motivation for developing these algorithms is the goal of analyzing data on the symmetric group using a spectral approach. In the simpler case of time series data on the cyclic group, this approach amounts to projecting the data vector onto the basis of complex exponentials.

The spectral approach to data analysis makes sense for a function defined on any kind of group, and such a general formulation is due to Diaconis (see, e.g., [8]). The case of the symmetric group corresponds to considering *ranked data*. For instance, a group of people might be asked to rank a list of four restaurants in order of preference. Thus, each respondent chooses a permutation of the original ordered list of four objects, and counting the number of respondents choosing each permutation yields a function on S_4. It turns out that the corresponding Fourier decomposition of this function naturally describes various coalition effects that may be useful in describing the data.

To get some feel for this, notice that the Fourier transform at the matrix element $\rho_{ij}(\pi)$ of the (reducible) defining representation counts the number of people ranking restaurant i in position j. If instead ρ is the (reducible) permutation representation of S_n on unordered pairs $\{i, j\}$, then for each choice of $\{i, j\}$ and $\{k, l\}$ the individual Fourier transforms count the number of respondents ranking restaurants i and j in positions k and l. See [8] for a more thorough explanation.

The first FFT for symmetric groups (an $O(|G| \log^3 |G|)$ algorithm) is due to M. Clausen. In what follows we summarize recent improvements on Clausen's result.

Example: Computing the Fourier Transform on S_4

The fast Fourier transform for S_4 is obtained by mimicking the group theoretic approach to the Cooley-Tukey algorithm. More precisely, we shall rewrite the formula for the Fourier transform using two changes of variables: one using factorizations of group elements, and the other using paths in a Bratteli diagram. The former comes from the reduced word decomposition of $g \in S_4$, by which g may be uniquely expressed as

$$g = s_2^4 \cdot s_3^4 \cdot s_4^4 \cdot s_2^3 \cdot s_3^3 \cdot s_2^2, \tag{16}$$

where s_i^j is either e or the transposition $(i\ i-1)$, and $s_{i_1}^j = e$ implies that $s_{i_2}^j = e$ for $i_2 \le i_1$. Thus any function on the group S_4 may be thought of as a function of the six variables s_2^4, s_3^4, s_4^4, s_2^3, s_3^3, s_2^2.

To index the matrix elements of S_4, paths in a Bratteli diagram are used, this time relative to the chain of subgroups $S_4 \ge S_3 \ge S_2 \ge S_1 \ge 1$. The irreducible representations of S_n are in one-to-one correspondence with partitions of the integer n, with restriction of representations corresponding to deleting a box in the Young diagram. The corresponding Bratteli diagram is called Young's lattice and is shown in Figure 2.

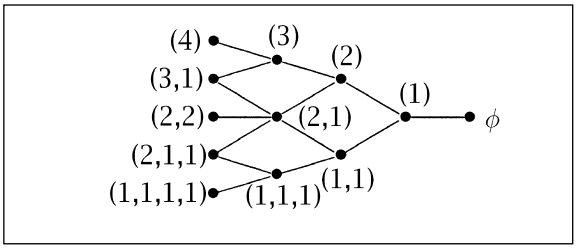


Figure 2. Young's lattice up to level 4.

Paths in Young's lattice from the empty partition ϕ to β_4, a partition of 4, index the basis vectors of the irreducible representation corresponding to β_4. Matrix elements, however, are determined by specifying a pair of basis vectors, so to index the matrix elements, we must use pairs of paths in Young's lattice, starting at ϕ and ending at the same partition of 4. Since there are no multiple edges in Young's lattice, each path may be described by the sequence of partitions $\phi, \beta_1, \beta_2, \beta_3, \beta_4$ through which it passes.

Before we can state a formula for the Fourier transform analogous to (11) and (15), we must choose bases for the irreducible representations of S_4 in order to define our matrix elements. Efficient algorithms are known only for special choices of bases, and our algorithm uses the representations in Young's orthogonal form, which is equivalent to the following equation (17) for the Fourier transform in the new sets of variables.

$$\hat{f}\begin{pmatrix}\beta_4\,\beta_3\,\beta_2\,\beta_1\\ \gamma_3\,\gamma_2\,\gamma_1\end{pmatrix} = \sum_{g=s_2^4 s_3^4 s_4^4 s_2^3 s_3^3 s_2^2} \sum_{\varphi_2,\varphi_1,\eta_1} \left[P^4_{s_4^4}\begin{pmatrix}\beta_4\,\beta_3\\ \gamma_3\,\varphi_2\end{pmatrix} P^3_{s_3^4}\begin{pmatrix}\beta_3\,\beta_2\\ \varphi_2\,\varphi_1\end{pmatrix} P^2_{s_2^4}\begin{pmatrix}\beta_2\,\beta_1\\ \varphi_1\end{pmatrix} \times P^3_{s_3^3}\begin{pmatrix}\gamma_3\,\varphi_2\\ \gamma_2\,\eta_1\end{pmatrix} P^2_{s_2^3}\begin{pmatrix}\varphi_2\,\varphi_1\\ \eta_1\end{pmatrix} P^2_{s_2^2}\begin{pmatrix}\gamma_2\,\eta_1\\ \gamma_1\end{pmatrix} f(g)\right]. \tag{17}$$

The functions $P^i_{s_i^j}$ in equation (17) are defined below, and for each i, the variables β_i, γ_i, φ_i, η_i are partitions of i satisfying the restriction relations described by Figure 3. A solid line between partitions means that the right-hand partition is obtained from the left-hand partition by removing a box.

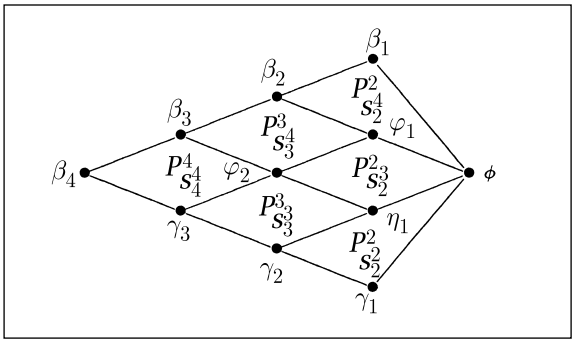


Figure 3. Restriction relations for (17).

The relationship between (17) and Figure 3 is extremely close: we derived the diagram from the reduced word decomposition first and then read the equation off the diagram. Each 2-cell in Figure 3 corresponds to a factor in the product of P functions in (17), and the labels on the boundary of each cell give the arguments of $P^i_{s_i^j}$. The sum in (17) is over those variables occurring in the interior of Figure 3. Thus, the variables describing the Fourier transformed function are exactly those appearing on the boundary of the figure.

Equation (17) can be summarized by saying that we take the product over 2-cells and sum on interior indices in Figure 3. This suggests a generalization of the Cooley-Tukey algorithm that corresponds to building up the diagram one cell at a time. At each stage multiply by the factor corresponding to a 2-cell and form the diagram consisting of those 2-cells that have been considered so far. Then sum over any indices that are in the interior of the diagram for this stage but were not in the interior for previous stages. At the end of this algorithm we have multiplied by the factors for each 2-cell and summed over all the interior indices, and have therefore computed the Fourier transform.

The order in which the cells are added matters, of course. The order s_2^2, s_2^3, s_3^3, s_2^4, s_3^4, s_4^4 is known to be most efficient. Here is the algorithm in detail.

- Stage 0: Start with $f(s_2^4 s_3^4 s_4^4 s_2^3 s_3^3 s_2^2)$, for all reduced words.
- Stage 1: Multiply by $P^2_{s_2^2}$. Sum on s_2^2.
- Stage 2: Multiply by $P^2_{s_2^3}$. Sum on s_2^3.
- Stage 3: Multiply by $P^3_{s_3^3}$. Sum on η_1, s_3^3.
- Stage 4: Multiply by $P^2_{s_2^4}$. Sum on s_2^4.
- Stage 5: Multiply by $P^2_{s_3^4}$. Sum on φ_1, s_3^4.
- Stage 6: Multiply by $P^3_{s_4^4}$. Sum on φ_2, s_4^4.

The indices occurring in each stage of the algorithm are shown in Figure 4.

To count the number of additions and multiplications used by the algorithm, we must count the number of configurations in Young's lattice corresponding to each of the diagrams in Figure 4. This yields a grand total of 130 additions and 130 multiplications for the Fourier transform on S_4.

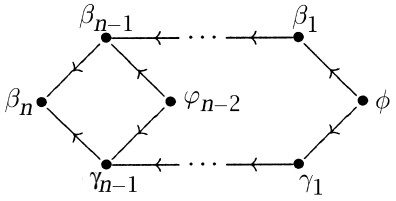
The generalization to higher-order symmetric groups is straightforward. The reduced word decomposition gives the group element factorization, Young's orthogonal form allows us to change variables, and the formula and algorithm for the Fourier transform can be read off a diagram generalizing Figure 3. The diagram for S_5, for example, is shown in Figure 5.

We have computed the exact operation counts for symmetric groups S_n with $n \le 50$,[3] and a general formula seems hard to come by. However, bounds are easier to obtain.

Theorem 1 [13]. *The number of additions (or multiplications) required by the above algorithm (as generalized to $S_n > S_{n-1} > \cdots > S_1$) is exactly*

$$n! \cdot \sum_{k=2}^{n} \frac{1}{k} \sum_{i=2}^{k} \frac{1}{(i-1)!} F_i,$$

where F_i is the number of configurations in Young's lattice of the form

(18)


Furthermore, $F_i \le 3(1-\frac{1}{i})i!$, so the number of additions (multiplications) is bounded by $\frac{3}{4}n(n-1)\cdot n!$.

Why stop at S_n? The algorithm for the FFT on S_n generalizes to any wreath product $S_n[G]$ with

[3] *This would seem to include all cases where the algorithm might ever be implemented, but the same numbers arise in FFTs on homogeneous spaces, which have far fewer elements.*

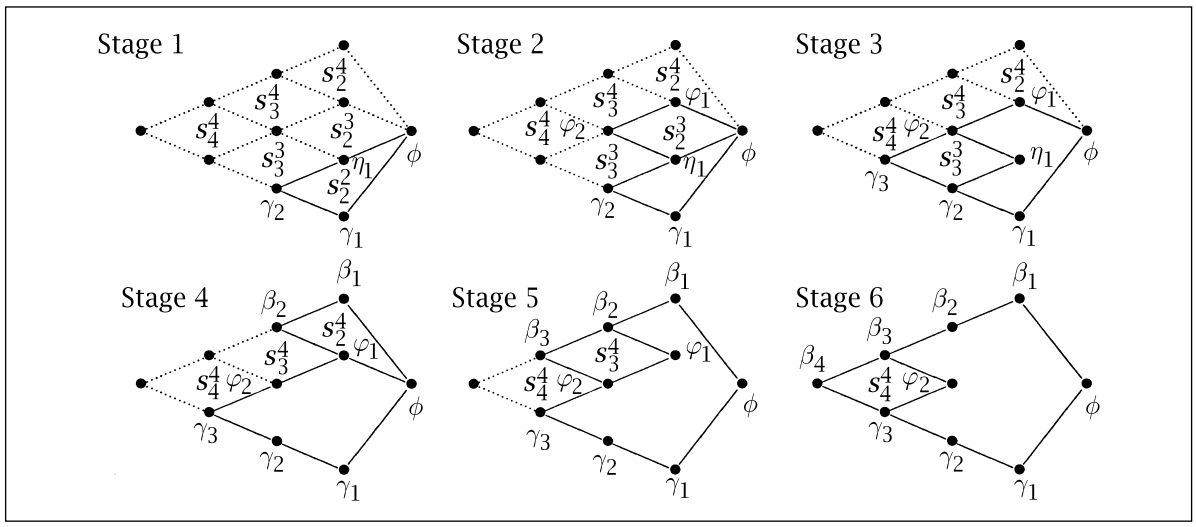


Figure 4. Variables occurring at each stage of the fast Fourier transform for S_4.

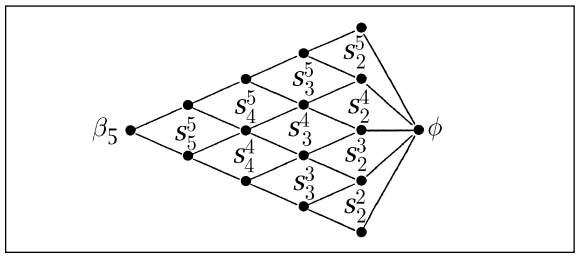


Figure 5. Restriction relations in the Fourier transform formula for S_5.

the symmetric group. The subgroup chain is replaced by the chain

$$S_n[G] > S_{n-1}[G] \times G > S_{n-1}[G] > \cdots > S_2[G] > G \times G > G, \tag{19}$$

and the reduced word decomposition is replaced by the factorization

$$x = s_2^n \cdots s_n^n g^n s_2^{n-1} \cdots s_{n-1}^{n-1} g^{n-1} \cdots s_s^2 g^2 g^1. \tag{20}$$

Adapting the S_n argument along these lines gives the following new result.

Theorem 2. *The number of operations needed to compute a Fourier transform on $S_n[G]$ is at most*

$$\left(\frac{3n(n-1)}{4}|G|d_G^2 + n\left[t_G + \frac{1}{4}|G|(h_G d_G^2 - |G|)\right]\right)|S_n[G]|,$$

where h_G is the number of conjugacy classes in G, d_G is the maximal degree of an irreducible representation of G, and t_G is the number of operations required to compute a Fourier transform on G. If G is abelian, then the inner term $h_G d_G^2 - |G| = 0$.

The functions $P^i_{s_j^i}$ defining Young's orthogonal form are defined as follows: For any two boxes b_1 and b_2 in a Young diagram, we define the axial distance from b_1 to b_2 to be $d(b_1, b_2)$, where $d(b_1, b_2) = \text{row}(b_1) - \text{row}(b_2) + \text{column}(b_1) - \text{column}(b_2)$. Now suppose β_i, β_{i-1}, α_{i-1}, α_{i-2} are partitions and that α_{i-1} and β_{i-1} are obtained from β_i by removing a box and are obtained from α_{i-2} by adding a box. Then the skew diagrams of $\beta_i - \beta_{i-1}$ and $\beta_{i-1} - \alpha_{i-2}$ each consists of a single box, and P^i is given by

$$P^i_e \begin{pmatrix} \beta_i & \beta_{i-1} \\ \alpha_{i-1} & \alpha_{i-2} \end{pmatrix} = \begin{cases} 1 & \text{if } \alpha_{i-1} = \beta_{i-1}, \\ 0 & \text{if } \alpha_{i-1} \neq \beta_{i-1}. \end{cases} \tag{21}$$

$$P^i_{(i\ i-1)} \begin{pmatrix} \beta_i & \beta_{i-1} \\ \alpha_{i-1} & \alpha_{i-2} \end{pmatrix} = \begin{cases} d(\beta_i - \beta_{i-1}, \beta_{i-1} - \alpha_{i-2})^{-1} & \text{if } \alpha_{i-1} = \beta_{i-1}, \\ \sqrt{1 - d(\beta_i - \beta_{i-1}, \beta_{i-1} - \alpha_{i-2})^{-2}} & \text{if } \alpha_{i-1} \neq \beta_{i-1}. \end{cases}$$

For a proof of this formula, in slightly different notation, see [11, Chapter 3].

Generalization to Other Groups

The FFT described for symmetric groups suggests a general approach to computing Fourier transforms on finite groups. Here is the recipe.

1. Choose a chain of subgroups

$$G = G_m \geq G_{m-1} \geq \cdots \geq G_1 \geq G_0 = 1 \tag{22}$$

for the group. This determines the Bratteli diagram that we will use to index the matrix elements of G. In the general case, this Bratteli diagram may have multiple edges, so a path is no longer determined by the nodes it visits.

2. Choose a factorization $g = g_n \cdot g_{n-1} \cdots g_1$ of each group element g. Choose the g_i so that they lie in as small a subgroup G_k as possible and commute with as large a subgroup G_l as possible.
3. Choose a system of Gel'fand-Tsetlin bases [9] for the irreducible representations of G relative to the chain (22). These are bases that are indexed by paths in the Bratteli diagram and that behave well under restriction of representations. Relative to such a basis, the representation matrices of g_i will be block diagonal

whenever g_i lies in a subgroup from the chain and block scalar whenever g_i commutes with all elements of a subgroup from the chain.

4. Now write the Fourier transform in coordinates as a function of the pairs of paths in the Bratteli diagram with a common endpoint and with the original function written as a function of $g_1, \ldots, g_n$. This will be a sum of products indexed by edges in the Bratteli diagram which lie in some configuration generalizing (3). This configuration of edges specifies the way in which the nonzero elements of the representation matrices appear in the formula for the Fourier transform in coordinates.
5. The algorithm proceeds by building up the product piece by piece and summing on as many partially indexed variables as possible.

Further Considerations and Generalizations

The efficiency of the above approach—in theory (in terms of algorithmic complexity) and in practice (in terms of execution time)—depends on both the choice of factorization and the Gel'fand-Tsetlin bases. In particular, very interesting work of L. Auslander, J. Johnson, and R. Johnson [2] shows how, in the abelian case, different factorizations correspond to different well-known FFTs, each well suited for execution on a different computer architecture. This work shows how to relate the 2-cocycle of a group extension to construction of the important "twiddle factor" matrix in the factorization of the Fourier matrix. It marks the first appearances of group cohomology in signal processing and derives an interesting connection between group theory and the design of retargetable software.

The analogous questions for nonabelian groups and other important signal processing transform algorithms, i.e., the problem of finding architecture-optimized factorizations, is currently being investigated by the SPIRAL project at Carnegie Mellon [19].

Another Abelian Idea—Rader's Prime FFT

The use of subgroups depends upon the existence of a nontrivial subgroup. Thus, for a reduction in the case of a cyclic group of prime order, a new idea is necessary. In this case, one possibility is an algorithm due to C. Rader [16] which proceeds by turning computation of the DFT into computation of convolution on a different, albeit related, group. Let p be a prime. Since $\mathbf{Z}/p\mathbf{Z}$ is also a finite field, there exists a generator g of $\mathbf{Z}/p\mathbf{Z}^\times$, a cyclic group (under multiplication) of order $p-1$. Thus, for any $f : \mathbf{Z}/p\mathbf{Z} \to \mathbf{C}$ and nonzero frequency index g^{-b}, $\widehat{f}(g^{-a})$ can be written as

$$\widehat{f}(g^{-b}) = f(0) + \sum_{a=0}^{p-2} f(g^a)\zeta_p^{g^{a-b}} \tag{23}$$

The summation in (23) has the form of a convolution on $\mathbf{Z}/(p-1)\mathbf{Z}$, of the sequence $f'(a) = f(g^a)$, with the function $z(a) = \zeta_p^{g^a}$, so that $\widehat{f}$ may be almost entirely computed using Fourier transforms of length $p-1$ for which Cooley-Tukey-like ideas may be used. It is a very interesting open question to discover if this idea has a nonabelian generalization.

Modular FFTs

A significant application of the abelian FFT is in the efficient computation of Fourier transforms for functions on cyclic groups defined over finite fields. These are needed for the efficient encoding and decoding of various polynomial error-correcting codes. Many abelian codes, e.g., the Golay codes used in deep-space communication, are defined as $\mathbf{F}_p$-valued functions on a group $\mathbf{Z}/m\mathbf{Z}$ with the property that $\widehat{f}(k) = 0$ for k in some specified set of indices S, where now the Fourier transform is defined in terms of a primitive $(p-1)$st root of unity.

These sorts of *spectral constraints* define *cyclic codes*, and they may immediately be generalized to any finite group. Recently, this has been done in the construction of codes over $SL_2(\mathbf{F}_p)$ using connections between expander graphs and linear codes discovered by M. Sipser and D. Spielman. For further discussion of this and other applications, see [17].

FFTs for Compact Groups

The DFT and FFT have a natural extension to continuous compact groups. The terminology "discrete Fourier transform" derives from the algorithm having been originally designed to compute the (possibly approximate) Fourier transform of a continuous signal from a discrete collection of sample values.

Under the simplifying assumption of periodicity, a continuous function may be interpreted as a function on the unit circle S^1, a compact abelian group. Any such function f has a *Fourier expansion* defined as

$$f(e^{2\pi it}) = \sum_{l \in \mathbf{Z}} \widehat{f}(l) e^{-2\pi ilt}, \tag{24}$$

where

$$\widehat{f}(l) = \int_0^1 f(e^{2\pi it}) e^{2\pi ilt}\, dt. \tag{25}$$

If $\widehat{f}(l) = 0$ for $|l| \geq N$, then f is *band-limited* with *band-limit* N, and the DFT (1) is in fact a *quadrature rule* or *sampling theorem* for f. That is, the DFT of the function $\frac{1}{2N-1} f(e^{2\pi it})$ on the group of $(2N-1)$st roots of unity computes exactly the Fourier coefficients of the band-limited function. The FFT then efficiently computes these Fourier coefficients.

The first nonabelian FFT for a compact group was a fast spherical harmonic expansion algorithm discovered by J. Driscoll and D. Healy. Several ingredients were required: (1) a notion of "band-limit" for functions on S^2, (2) a sampling theory for such functions, and (3) a fast algorithm for the computation.

The spherical harmonics are naturally indexed according to their order (the common degree of a set of homogeneous polynomials on S^2). With respect to the usual coordinates of latitude and longitude, the spherical harmonics separate as a product of exponentials and associated Legendre functions, each of which separately has a sampling theory. Finally, using the usual FFT for the exponential part and a new fast algorithm (based on three-term recurrences) for the Legendre part forms an FFT for S^2.

These ideas generalize nicely. Keep in mind that the representation theory of compact groups is much like that of finite groups: there is a countable complete set of irreducible representations, and any square-integrable function (with respect to Haar measure) has an expansion in terms of the corresponding matrix elements. There is a natural definition of band-limited in the compact case, encompassing those functions whose Fourier expansion has only a finite number of terms. The simplest version of the theory is as follows:

Definition. *Let $\mathcal{R}$ denote a complete set of irreducible representations of a compact group G. A* system of band-limits on G *is a decomposition of $\mathcal{R} = \cup_{b\geq 0}\mathcal{R}_b$ such that*

1. $\mathcal{R}_b$ is finite for all $b \geq 0$,
2. $b_1 \leq b_2$ implies that $\mathcal{R}_{b_1} \subseteq \mathcal{R}_{b_2}$,
3. $\mathcal{R}_{b_1} \otimes \mathcal{R}_{b_2} \subseteq \text{span}_{\mathbf{Z}}\mathcal{R}_{b_1+b_2}$.

Suppose $\{\mathcal{R}_b\}_{b\geq 0}$ is a system of band-limits on G and $f \in L^2(G)$. Then f *is band-limited with band-limit b* if the Fourier coefficients are zero for all matrix elements in ρ for all $\rho \notin \mathcal{R}_b$.

The case of $G = S^1$ provides the classical example. If $\mathcal{R}_b = \{\chi_j : |j| \leq b\}$, where $\chi_j(z) = z^j$, then $\chi_j \otimes \chi_k = \chi_{j+k}$, and the notion of band-limited corresponding to the definition coincides with the usual notion.

For a nonabelian example, consider $G = SO(3)$. In this case the irreducible representations of G are indexed by the nonnegative integers, with V_λ the unique irreducible of dimension $2\lambda + 1$. Let $\mathcal{R}_b = \{V_\lambda : \lambda \leq b\}$. The Clebsch-Gordon relations

$$V_{\lambda_1} \otimes V_{\lambda_2} = \sum_{j=|\lambda_1-\lambda_2|}^{\lambda_1+\lambda_2} V_j \tag{26}$$

imply that this is a system of band-limits for $SO(3)$. When restricted to the quotient $S^2 \cong SO(3)/SO(2)$, band-limits are described in terms of the highest order spherical harmonics that appear in a given expansion.

This notion of band-limit permits the construction of a sampling theory [14]. For example, in the case of the classical groups, a system of band-limits $\mathcal{R}_b^n$ is chosen with respect to a particular norm on the dual of the associated Cartan subalgebra. Such a norm $\|\cdot\|$ (assuming that it is invariant under taking duals and that $\|\alpha\| \leq \|\beta\| + \|\gamma\|$ for α occurring in $\beta \otimes \gamma$) defines a notion of band-limit given by all α with norm less than a fixed b. This generalizes the definition above. The associated sampling sets X_b^n are contained in certain one-parameter subgroups. These sampling sets permit a separation of variables analogous to that used in the Driscoll-Healy FFT. Once again the special functions satisfy certain three-term recurrences which admit a similar efficient divide-and-conquer computational approach (see [15] and references therein). One may derive efficient algorithms for all the classical groups $U(n)$, $SU(n)$, and $Sp(n)$.

Theorem 3. *Assume $n \geq 2$.*

(i) For $U(n)$, $T_{X_b^n}(\mathcal{R}_b^n) \leq O(b^{\dim U(n)+3n-3})$,
(ii) for $SU(n)$, $T_{X_b^n}(\mathcal{R}_b^n) \leq O(b^{\dim SU(n)+3n-2})$,
(iii) for $Sp(n)$, $T_{X_b^n}(\mathcal{R}_b^n) \leq O(b^{\dim Sp(n)+6n-6})$,

where $T_{X_b^n}(\mathcal{R}_b^n)$ denotes the number of operations needed for the particular sample set X_b^n and representations $\mathcal{R}_b^n$ for the associated group.

Further and Related Work

Noncompact Groups

Much of modern signal processing relies on the understanding and implementation of Fourier analysis for $L^2(\mathbf{R})$, i.e., the noncompact abelian group $\mathbf{R}$. Nonabelian, noncompact examples have begun to attract much attention.

In this area some of the most exciting work is being done by G. Chirikjian and his collaborators. They have been exploring applications of convolution on the group of rigid motions of Euclidean space to such diverse areas as robotics, polymer modeling, and pattern matching. See [5] for details and pointers to the literature.

To date the techniques used here are approximate in nature, and interesting open problems abound. Possibilities include the formulation of natural sampling, band-limiting, and time-frequency theories. The exploration of other special cases such as semisimple Lie groups (see [1] for a beautifully written, succinct survey of the Harish-Chandra theory) would be one natural place to start. A sampling and band-limiting theory would be the first step towards developing a computational theory, i.e., FFT. "Fast Fourier transforms on semisimple Lie groups" has a nice ring to it!

Approximate Techniques

The techniques in this paper are all exact, in the sense that if computed in exact arithmetic, they yield exactly correct answers. Of course, in any actual implementation, errors are introduced, and the utility of an algorithm will depend highly on its numerical stability.

There are also "approximate methods", approximate in the sense that they guarantee a certain specified approximation to the exact answer

that depends on the running time of the algorithm. For computing Fourier transforms at nonequispaced frequencies, as well as spherical harmonic expansions, the fast *multipole method* due to V. Rokhlin and L. Greengard is a recent and very important approximate technique. Multipole-based approaches efficiently compute these quantities approximately in such a way that the running time increases by a factor of $\log(\frac{1}{\epsilon})$, where ϵ denotes the precision of the approximation. M. Mohlenkamp has applied quasi-classical frequency estimates to the approximate computation of various special function transforms.

Quantum Computing

Another related and active area of research involves connections with quantum computing. One of the first great triumphs of the quantum computing model is P. Shor's fast algorithm for integer factorization on a quantum computer [18]. At the heart of Shor's algorithm is a subroutine which computes (on a quantum computer) the DFT of a binary vector representing an integer. The implementation of this transform as a sequence of one- and two-bit quantum gates is the *quantum FFT*, effectively the Cooley-Tukey FFT realized as a particular factorization of the Fourier matrix into a product of matrices composed as tensor products of certain 2×2 unitary matrices, each of which is a "local unitary transform". Extensions of these ideas to the more general group transforms mentioned above are a current important area of research of great interest in computer science.

Final Remarks

So, these are some of the things that go into the computation of the finite Fourier transform. It is a tapestry of mathematics both pure and applied, woven from algebra and analysis, complexity theory, and scientific computing. It is on the one hand a focused problem, but like any good problem, its "solution" does not end a story, but rather initiates an exploration of unexpected connections and new challenges.

References

[1] J. Arthur, Harmonic analysis and group representations, *Notices Amer. Math. Soc.* **47** (2000), 26–34.

[2] L. Auslander, J. R. Johnson, R. W. Johnson, Multidimensional Cooley-Tukey algorithms revisited, *Adv. Appl. Math.* **17** (1996), 477–519.

[3] L. Auslander and R. Tolimieri, Is computing with the finite Fourier transform pure or applied mathematics? *Bull. Amer. Math. Soc. (N.S.)* **1** (1979), 847–897.

[4] T. Beth, *Verfahren der schnellen Fourier-Transformation*, Teubner Studienbücher, Stuttgart, 1984.

[5] G. S. Chirikjian and A. B. Kyatkin, *Engineering Applications of Noncommutative Harmonic Analysis*, CRC Press, Boca Raton, FL, 2000.

[6] J. W. Cooley, The re-discovery of the fast Fourier transform algorithm, *Mikrochimica Acta* **III** (1987), 33–45.

[7] J. W. Cooley and J. W. Tukey, An algorithm for machine calculation of complex Fourier series, *Math. Comp.* **19** (1965), 297–301.

[8] P. Diaconis, *Group Representations in Probability and Statistics*, Inst. Math. Stat., Hayward, CA, 1988.

[9] I. Gel'fand and M. Tsetlin, Finite dimensional representations of the group of unimodular matrices, *Dokl. Akad. Nauk SSSR* **71** (1950), 825–828 (Russian).

[10] M. T. Heideman, D. H. Johnson, and C. S. Burrus, Gauss and the history of the fast Fourier transform, *Arch. Hist. Exact Sci.* **34** (1985), 265–277.

[11] G. James and A. Kerber, *The Representation Theory of the Symmetric Group*, Encyclopedia Math. Appl., vol. 16, Addison-Wesley, Reading, MA, 1981.

[12] T. Y. Lam, Representations of finite groups: A hundred years, I and II. *Notices Amer. Math. Soc.* **45** (1998), 361–372, 465–474.

[13] D. K. Maslen, The efficient computation of Fourier transforms on the symmetric group, *Math. Comp.* **67** (1998), 1121–1147.

[14] ———, Efficient computation of Fourier transforms on compact groups, *J. Fourier Anal. Appl.* **4** (1998), 19–52.

[15] D. K. Maslen and D. N. Rockmore, Generalized FFTs—a Survey of Some Recent Results, *Groups and Computation, II* (New Brunswick, NJ, 1995), DIMACS Ser. Discrete Math. Theoret. Comput. Sci., vol. 28, Amer. Math. Soc., Providence, RI, 1997, pp. 183–237.

[16] C. Rader, Discrete Fourier transforms when the number of data samples is prime, *IEEE Proc.* **56** (1968), 1107–1108.

[17] D. N. Rockmore, Some applications of generalized FFTs (an appendix w/D. Healy), *Groups and Computation, II* (New Brunswick, NJ, 1995), DIMACS Ser. Discrete Math. Theoret. Comput. Sci., vol. 28, Amer. Math. Soc., Providence, RI, 1997, pp. 329–369.

[18] P. W. Shor, Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, *SIAM J. Comput.* **26** (1997), 1484–1509.

[19] `http://www.ece.cmu.edu/~spiral/`.

[20] A. Willsky, On the algebraic structure of certain partially observable finite-state Markov processes, *Inform. Contr.* **38** (1978), 179–212.