Sampling Boolean Functions over Abelian Groups and Applications

Meera Sitharam[1]* and Timothy Straney[2]

[1] CISE Department, University of Florida, Gainesville FL 32611-6120,
sitharam@cise.ufl.edu
[2] Department of Math and CS, Kent State University, Kent OH 44240,
tstraney@mcs.kent.edu

Abstract. We obtain efficient sampling methods for recovering or compressing functions over finite Abelian groups with few Fourier coefficients, i.e, functions that are (approximable by) linear combinations of few, possibly *unknown* Fourier basis functions or characters.

Furthermore, our emphasis is on efficiently and *deterministically* finding small, *uniform* sample sets, which can be used for sampling *all* functions in natural approximation classes of Boolean functions. Due to this requirement, even the simplest versions of this problem (say, when the set of approximating characters is known) require somewhat different techniques from the character theory of finite Abelian groups that are commonly used in other discrete Fourier transform applications.

We briefly discuss applications of our efficient, uniform sampling methods in computational learning theory, efficient generation of pseudorandom strings, and testing linearity; we also state highly related open problems that are not only applicable in these contexts, but are also of independent mathematical interest.

Keywords: Abelian groups, Boolean functions, computational learning theory, data compression, Diophantine equations, invariant factor decomposition, linearity testing, pseudorandom generators, VC-dimension.

1 Introduction

Complex and Boolean functions over finite Abelian groups arise in a wide variety of contexts. Assume we know the amount of information contained in a function according to some measure, say the number m of the Fourier coefficients where most of the spectrum is concentrated. The task is to find a good set of roughly m sample points at which to evaluate the function and use this information to obtain the Fourier coefficients. Furthermore we would like these sample points to be *uniformly usable* for the entire class of functions with the same spectrum concentration.

This reduces to a *nonlinear approximation* question: that of interpolation at the chosen set of sample points by m-sparse Fourier expansions for the domain group G. This is a nonlinear interpolation question since the actual set

* Supported in part by NSF Grant CCR 94-09809

of nonzero Fourier coefficients is unknown. Hence there is *no fixed linear subspace* of functions from which the interpolant is sought. For general redundant bases over the reals, with restricted orthonormality conditions, this is a difficult problem in nonlinear approximation theory, which is one of the driving forces of modern approximation theory, including Wavelet theory [16], [17], [36].

Over discrete domains where algebraic structure can be used, such nonlinear interpolation questions are sometimes called blackbox-interpolation by m-sparse Fourier expansions (with some restrictions on the set of Fourier terms) for the domain $\mathbb{Z}_p^n$. Viewing the domain as U_p^n, where U_p represents the set of complex p^{th} roots of unity, this can be alternatively stated as a question of blackbox-interpolation by m-sparse, complex valued polynomials (with some restrictions on the set of terms of the polynomial). For the case of interpolation by m-sparse, real-valued polynomials, where the domain is $\mathbb{Z}^n$, this problem has been dealt with, e.g, in [5], and [53] (which uses other information about the function in addition to point evaluation of the function); the case of m-sparse polynomials with values in $\mathbb{F}_p$, where the domain is $\mathbb{F}_p^n$, is dealt with in [10]; and for m-sparse, real valued polynomials over the cube domain or $\mathbb{Z}_2^n$, the problem is dealt with in [43].

To solve such a nonlinear interpolation problem, let us consider the obvious approaches: one could solve a linear interpolation question over the entire space of functions on G, but that would involve evaluating the function exhaustively at all elements of G, which defeats the purpose. Or alternatively, one could exhaustively search all subspaces spanned by m Fourier basis functions, and attempt to solve a linear interpolation system for each subspace until successful. Yet other methods, used in standard FFT based sampling [49], [58], [29], [43], [13], [11] - take advantage of orthonormality and use projection to adaptively search for large Fourier coefficients, by successively subdividing and pruning the domain. This in turn is done by estimating the concentration of the spectrum on subdomains using randomized sampling based on the algebraic structure of the underlying domain and Parseval's identity. Our goal however, is to *avoid* such exhaustive *or* randomized sampling and search.

Equally significantly, we wish to obtain a *uniform* sample set that works for all functions with the same Fourier concentration or support. In this context, it is important to note that *any* method - including standard FFT based methods, [49], [58] [29], [43], [13], [11] - which rely on projection to adaptively isolate large Fourier coefficients – would *also* work for functions that are approximable in the 2-norm from the space spanned by few Fourier basis functions. However, a simple result has been shown by the authors in [57], that there is *no* uniform, good sample set for such classes of functions, thus rendering such methods unsuitable for our purposes.

We prove two main results on sampling.
The first main result is found in Subsection 3.1 and develops an algorithm that takes as input a succinct description (the invariant factor decomposition) of a finite Abelian group G, and a listing of a set Q of irreducible characters or Fourier basis functions Q, and outputs a set of $|Q|$ sample points S with the following property: if any function f over G can be expressed as a linear combination of the irreducible characters in Q, then by evaluating the function at the points in S one can recover a complete description (the Fourier coefficients) of f by solving a $|Q| \times |Q|$ linear interpolation system. If the group $G = \mathbb{Z}_{n_1} \oplus \cdots \oplus$

$\mathbb{Z}_{n_k}$, where $(n_1, \ldots, n_k)$ are the invariant factors of G, then our algorithm takes $\mathcal{O}(k^2 n_1^2 |Q|^2 \log^3 |Q| \log n_1)$ steps to recover f. Since G is a finite Abelian group, the set of characters Q is isomorphic to a subset of G, but we in fact assume that Q is (isomorphic to) a subgroup of G.

The second main result is in Subsection 3.2 and develops an efficient algorithm for completely recovering (or compressing) a Boolean function f over the group $\mathbb{Z}_p^n$ (p prime) which is known to have at most m nonzero Fourier coefficients. The algorithm also produces a sample set that can be uniformly used for recovering any function with the same m nonzero Fourier cefficients as f. (Here the vector space structure of $\mathbb{F}_p^n$ is used, and the set of m Fourier basis functions or characters belong to some unknown subspace of $\mathbb{F}_p^n$). The inputs to the algorithm are just m, n and p, as well as access to the values of the function at any chosen set of m points. The algorithm runs in time $\max\{\mathcal{O}(n^4 m^2), \mathcal{O}(n^{\log_p m+1} m^2)\}$.

Finally, our results were strongly motivated by applications in computational learning theory and give efficient deterministic algorithms for learning functions in Fourier approximation classes. These approximation classes have been investigated extensively in complexity theory, especially in the context of proving complexity lower bounds which are valuable and typically hard to obtain (see e.g, [40], [41], [12], [55], [50], [48]). In Section 4 of this paper, we sketch applications in learning theory, pseudorandom number generation and linearity testing. We also sketch open problems whose solutions would not only be applicable in these contexts, but are also of independent mathematical interest. In particular, taking advantage of the Booleanness restriction relates one of these problems to a well known problem [35], [45], [14], [6], [39], [47], [4], which has other applications: that of characterizing norms and the structure of zeroes of unimodular polynomials, i.e, polynomials whose coefficients are of modulus 1.

Organization. Section 2 gives preliminaries, character theory background, and background on basic algebraic algorithms that are needed for proving the results in Section 3, Section 3 gives efficient algorithms for finding good sample sets for recovering (equivalently, compressing) functions over finite Abelian groups. Section 4 discusses applications of efficient sampling.

2 Preliminaries and basic algorithms

We refer the reader to [19], [33] for many of the required conventions and much of the notation on finite Abelian groups, character theory and to [18], [52], [51] for Fourier transforms over finite abelian groups. The Fundamental Theorem of Finitely Generated Abelian Groups (p. 159) [19] states that each finite Abelian group G is isomorphic to $\mathbb{Z}_{n_1} \oplus \cdots \oplus \mathbb{Z}_{n_k}$, where $n_i \geq 2$ for $1 \leq i \leq k$ and $n_{i+1} | n_i$ for $1 \leq i \leq k-1$. The form $\mathbb{Z}_{n_1} \oplus \cdots \oplus \mathbb{Z}_{n_k}$ is unique and is referred to as G's *invariant factor decomposition.* The subscripts $(n_1, \ldots, n_k)$ are called G's invariant factors. Since the Abelian group $\mathbb{Z}_{n_1} \oplus \cdots \oplus \mathbb{Z}_{n_k}$ is in general not a vector space we can not speak of an inner product in the sense of a vector space. Nonetheless we define a natural map $\langle \cdot, \cdot \rangle : (\mathbb{Z}_{n_1} \oplus \cdots \oplus \mathbb{Z}_{n_k}) \times (\mathbb{Z}_{n_1} \oplus \cdots \oplus \mathbb{Z}_{n_k}) \to \mathbb{Z}_{n_1}$ by $\langle (x_1, \ldots, x_k), (y_1, \ldots, y_k) \rangle = (\sum_{i=1}^{k} (n_1/n_i) x_i y_i) \bmod n_1$, where all operations are done in the integers, after which the result is computed

modulo n_1. Thus, if $Q \subseteq G$, we define $Q^\perp \equiv \{x \in G : \langle q, x\rangle = 0, \forall q \in Q\}$. Note that if Q is a subset of G, then $Q^\perp \leq G$. We use the following straightforward observation repeatedly.

Observation 1. *If G is a finite Abelian group and $Q \leq G$, then $|G|/|Q^\perp| = |Q|$.*

Let $G = \mathbb{Z}_{n_1} \oplus \cdots \oplus \mathbb{Z}_{n_k}$, where $\mathbb{Z}_{n_1} \oplus \cdots \oplus \mathbb{Z}_{n_k}$ is G's invariant factor decomposition, and let ζ be the primitive ${n_1}^{th}$ root of unity of least positive amplitude, i.e. $\zeta = e^{2\pi i/n_1}$. In this paper we will assume each finite Abelian group actually *equals* its invariant factor decomposition, though in general such a group is only isomorphic to its invariant factor decomposition.

A *linear character* χ of a group G with values in $\mathbb{C}$ is a homomorphism from G into the multiplicative group of complex numbers $\mathbb{C}$, i.e. $\chi : G \to \mathbb{C}^\times$ (p. 482) [19]. Note that $\chi(0) = 1$, since χ is a homomorphism. In particular, let $G = \mathbb{Z}_{n_1} \oplus \cdots \oplus \mathbb{Z}_{n_k}$ is a finite Abelian group with invariant factors $(n_1, \ldots, n_k)$, and $x = (x_1, \ldots, x_k)$ and $y = (y_1, \ldots, y_k) \in G$ and $\zeta = e^{2\pi i/n_1}$, define $\chi_y : G \to \{\zeta^m : m \in \mathbb{Z}_{n_1}\}$ by:

$$\chi_y(x) = \zeta^{\langle x,y\rangle} = \zeta^{(\sum_{i=1}^{k} (n_1/n_i) x_i y_i) \bmod n_1}. \tag{1}$$

Then the set $\{\chi_y : y \in G\}$ is the set of characters of G.

The set of complex-valued functions on an Abelian group G is a vector space over $\mathbb{C}$ of dimension $|G|$. If f and g belong to this space, define an inner product $\langle f, g\rangle \equiv 1/|G| \sum_{x\in G} f(x)\overline{g(x)}$. Note that the set of characters $\{\chi_y : y \in G\}$ is an orthonormal basis under this inner product. For a finite Abelian group G, if $f : G \to \mathbb{C}$, then its *Fourier transform* is a function $\hat{f} : G \to \mathbb{C}$, defined by $\hat{f}(y) \equiv 1/|G| \sum_{x\in G} f(x)\overline{\chi_x(y)} = 1/|G| \sum_{x\in G} f(x)\overline{\chi_y(x)} = \langle f, \chi_y\rangle$. The support of $f : G \to \mathbb{C}$ is denoted spt $f \equiv \{x \in G : f(x) \neq 0\}$.

The following set of basic facts is used repeatedly.

Fact 2. *1. If G is a finite Abelian group, then χ is an irreducible character of G, denoted $\chi \in$ Irr(G) if and only if χ is linear, i.e. $\chi(0) = 1$. (p. 16) [33]*
2. If G is a group with commutator subgroup G', then $[G : G']$ = the number of linear characters of G. (p. 25) [33]
3. If G is an Abelian group, $|Irr(G)| = |G|$. Then the set of characters $\{\chi_y : y \in G\} =$ Irr(G).
4. If G is a finite Abelian group, $Q \leq G$ and $\chi \in Irr(G)$, then $\chi|_Q \in Irr(Q)$.
5. If G is a finite Abelian group and $Q \leq G$, then $\{\chi|_Q : \chi \in$ Irr$(G)\} = Irr(Q)$.

Fact 3. *Let G be a finite Abelian group and $Q \leq G$. If $\mathcal{T}$ is a transversal of $Q^\perp$, then $\{\chi_x|_Q : x \in \mathcal{T}\} = Irr(Q)$.*

2.1 Generating the Invariant Factor Decomposition and Solving Linear Systems over $\mathbb{Z}_k$

First, we discuss how to determine the structure of a subgroup of a finite Abelian group when this subgroup, call it Q, is given as a listing of elements. The algorithm is a component in the proof of the first main result (Theorem 13 in Section 3).

If Q is represented by a finite group *presentation*, the task of finding its invariant factor decomposition is relatively straightforward. The process involves constructing a matrix derived from the subgroup's relators and transforming this matrix into Smith Normal form, a form from which this subgroup's invariant factors can be read, see, e.g.: [54]. Unfortunately the subgroups we deal with are represented only as lists of their elements. In order to change this representation into a group presentation is a labor intensive chore we wish to avoid. Therefore we employ an algorithm which will accomplish the same objective when the input is a list of elements.

The algorithm we describe is simple, but we include it for completeness sake. Our algorithm will output elements $y_1, \ldots, y_j \in Q$ such that $Q = \langle y_1 \rangle \oplus \ldots \oplus \langle y_j \rangle$, where $\langle y_i \rangle \cong \mathbb{Z}_{m_i}$ for each i, and $(m_1, \ldots, m_j)$ are Q's invariant factors. In addition, the algorithm will run in time polynomial in $|Q|$ and the number of invariant factors of the larger group G of which Q is a subgroup. In order to develop the required algorithm, we need the following Lemma, which is a corollary of Theorem 7.12 [32].

Lemma 4. *Let P be a finite Abelian p-group. Let $C = \langle g \rangle$, where g is an element in P of maximal order. Let $x_0 = 0$. For $i \geq 1$, let $x_i \in P \setminus (C + \langle x_0, x_1, \ldots, x_{i-1} \rangle)$ with the property that $px_i \in \langle x_0, x_1, \ldots, x_{i-1} \rangle$. If k is the least index such that there exists no element $x_{k+1} \in P \setminus (C + \langle x_0, x_1, \ldots, x_k \rangle)$ with $px_{k+1} \in \langle x_0, x_1, \ldots, x_k \rangle$, then $P = C \oplus \langle x_0, x_1, \ldots, x_k \rangle$.*

The algorithm for finding the invariant factor decomposition of a subgroup Q of $G = \mathbb{Z}_{n_1} \oplus \cdots \oplus \mathbb{Z}_{n_k}$ begins by segregating all the elements of prime power order for each prime p that divides $|Q|$, thus identifying the Sylow$-p$ subgroups of Q. Lemma 4 is then used to decompose each Sylow$-p$ subgroup P into its invariant factor decomposition, $P_1 \oplus \cdots \oplus P_j$. Finally corresponding summands in each of these decompositions are grouped together to form a direct sum which is the invariant factor decomposition of Q. The complexity of this algorithm is formalized by the following theorem.

Theorem 5. *Let $G = \mathbb{Z}_{n_1} \oplus \ldots \oplus \mathbb{Z}_{n_k}$ be a finite Abelian group with invariant factors $(n_1, \ldots, n_k)$. Let Q be a subgroup of G which is given as a listing of elements. Then there exists an algorithm that runs in time $\mathcal{O}(k^2|Q|^2 \log^2 |Q|)$ and identifies elements $y_1, \ldots, y_j \in Q$ such that $Q = \langle y_1 \rangle \oplus \ldots \oplus \langle y_j \rangle$, where $\langle y_i \rangle \cong \mathbb{Z}_{m_i}$ for each i, and $(m_1, \ldots, m_j)$ are Q's invariant factors.*

Next we briefly discuss the issue of providing *one* solution to a linear system of equations over the ring $\mathbb{Z}_k$, if any exist, where k is an integer ≥ 2 which could be composite. Such an algorithm is a component in the proof of the first main result (Theorem 13) in Section 3 and is used directly in the proof of Theorem 12.

Much has been written on the subject of solving systems of linear Diophantine equations, and equations over rings $\mathbb{Z}_k$. See [21] [22]. Standard software is also available, for example in Maple [46]: linsolve mod m and msolve. The key point is that we are not interested in an algorithm that provides all solutions to the input system, it is sufficient if it provides *one*, if any solution exists.

Hence we can employ a fairly straightforward modification of the simple algorithm for solving linear diophantine systems found in (pp. 326-327)[38]. We

state the existence of this method without proof. The interested reader is referred to [24], [15], [30] and [31] for faster, but more complicated algorithms, and for example [23] for fast algorithms for sparse systems.

Theorem 6. *There is an algorithm (a simple modification of the algorithm in (pp. 326-327)[38]) for solving a linear system over the ring $\mathbb{Z}_k$, which correctly computes atleast one solution to the system, if any exist, in time bounded by $\mathcal{O}(m^2n^2k^2\log k)$.*

3 Sampling Boolean functions with few Fourier coefficients

In this section, we give efficient algorithms for finding *uniform*, good sample sets for the following classes of Boolean and Complex valued functions.

Definition 7. Let G be a finite Abelian group. If $Q \subseteq G$, we define the classes

$$D_0^{G,Q} = \{f : G \to \mathbb{C} : \text{spt } \hat{f} \subseteq Q\},$$

$$C_0^{G,Q} = \{f : G \to \{0,1\} : \text{spt } \hat{f} \subseteq Q\}.$$

Next, we formally define the notion of a uniformity in sample sets. This is a point to be strongly emphasized since it is essential for the applications described later.

Definition 8. Let the group $G = \mathbb{Z}_{n_1} \oplus \cdots \oplus \mathbb{Z}_{n_k}$, and let C be a class of complex-valued functions over G. A subset $S \subseteq G$ is a *good sample set* for a function $f \in C$ if S is of minimal size and $f|_S$ completely specifies f in C; i.e, if $g \in C$ and $g|_S = f|_S$, then $g = f$. The set S is a *uniform* good sample set for the class C if it is of minimal size, and for all $g, f \in C$, $g|_S = f|_S$ implies $g = f$.

Before we present the main sampling results, we first recall below the standard interpolation by which a function in $D_0^{G,Q}$, can be completely recovered (i.e, a complete description of the Fourier coefficients obtained), from its values on a good sample set. See also [18], [29], etc. for basics of sampling.

First, note that if $Q = \{q_1, \ldots, q_m\}$ is any subset of a finite Abelian group G and $f \in D_0^{G,Q}$, then $f = \hat{f}(q_1)\chi_{q_1} + \cdots + \hat{f}(q_m)\chi_{q_m}$. Thus in order to recover f *exactly* one may select m elements from G, say $x_1, \ldots, x_m$, such that the set of vectors

$$\{(\chi_{q_1}(x_1), \ldots, \chi_{q_m}(x_1)), \ldots, (\chi_{q_1}(x_m), \ldots, \chi_{q_m}(x_m))\}$$

is linearly independent. In other words:

Observation 9. *A subset $S = \{x_1, \ldots, x_m\}$ of a finite Abelian group G is a uniform, good sample set for $D_0^{G,Q}$, where $Q = \{q_1, \ldots, q_m\}$, if and only if the matrix $[\chi_{q_j}(x_i)]_{\substack{i \in \{1,\ldots,|S|\} \\ j \in \{1,\ldots,|Q|\}}}$ is invertible.*

Note that such a sample set always exists: if $G = \{x_1, \ldots, x_{|G|}\}$, then the $|G| \times |Q|$ matrix, $[\chi_{q_j}(x_i)]_{\substack{i \in \{1,\ldots,|G|\} \\ j \in \{1,\ldots,|Q|\}}}$ has $|Q|$ columns, each of which represents a distinct irreducible character of G, i.e. χ_{q_j}, with $1 \leq j \leq m$, and hence linearly independent, in fact unitary (orthonormal). Therefore the row rank equals the column rank equals m, i.e. the existence problem is solved.

The following lemma gives a characterization of uniform, good sample sets that follows as a corollary to Fact 3 and will be used in the proofs of the main results.

Lemma 10. *Let G be a finite Abelian group. Let $Q = \{q_1, \ldots, q_{|Q|}\}$ be a subgroup of G and $S = \{x_1, \ldots, x_{|Q|}\}$ be a subset of G. Then $H_{S,Q} = [\chi_{x_i}(q_j)]_{i,j \in \{1,\ldots,|Q|\}} = [\chi_{q_j}(x_i)]_{i,j \in \{1,\ldots,|Q|\}}$ is invertible with inverse $1/|Q|\overline{H_{S,Q}}^{\,t}$ if and only if S is a transversal of $Q^{\perp}$.*

3.1 Known Basis Q

We begin with a theorem that follows directly from Lemma 10 which identifies uniform, good sample sets for $D_0^{G,Q}$, when Q is *either* a subgroup *or* a transversal.

Theorem 11. *Let G be a finite Abelian group.*

(i) If Q is a subgroup of G and S is any transversal of $Q^{\perp}$, then S is a uniform, good sample set for $D_0^{G,Q}$ of size $|Q|$.

(ii) If Q is a transversal in G, any subgroup S of G such that Q is a transversal of $S^{\perp}$ is a uniform, good sample set for $D_0^{G,Q}$ of size $|Q|$.

The next theorem gives an algorithm for finding transversals.

Theorem 12. *Given the invariant factor decomposition of a subgroup Q of the finite Abelian group $G = \mathbb{Z}_{n_1} \oplus \cdots \oplus \mathbb{Z}_{n_k}$, with invariant factors $(n_1, \ldots, n_k)$, a transversal S of $Q^{\perp}$ can be constructed in time bounded above by $max\{\mathcal{O}(k|Q|), \mathcal{O}(k^2 n_1^2 \log^3 |Q| \log n_1)\}$.*

Proof. Let Q be a subgroup of G such that $Q = \langle y_1 \rangle \oplus \cdots \oplus \langle y_l \rangle$, where $\langle y_i \rangle \cong \mathbb{Z}_{m_i}$ for each i, and Q has invariant factors $(m_1, \ldots, m_l)$. Let $\zeta = e^{2\pi i/n_1}$. For $1 \leq i \leq l$, let z_i be an element of G satisfying:

$$\chi_{z_i}(y_j) = \begin{cases} \zeta^{n_1/m_i} & \text{if } j = i \\ 1 & \text{if } j \in \{1, \ldots, i-1, i+1, \ldots, l\} \end{cases} \tag{2}$$

Note that if we linearly extend to Q the function χ_{z_i}, defined by the equations in (2), then χ_{z_i} is a linear character of Q and therefore belongs to $\mathrm{Irr}(Q) = \{\chi_z|_Q : z \in G\}$. That is to say there exists $z \in G$ such that $\chi_{z_i} = \chi_z|_Q$. In particular this means there exists a $z_i \in G$ satisfying system (2). In fact, if z_i is any such satisfying value, then each z in $z_i + Q^{\perp}$ will also satisfy (2), i.e. there are $|Q^{\perp}|$ values $z_i \in G$ satisfying (2). Since for each $j \in \{1, \ldots, l\}$, potentially many values of z_i satisfy (2), let us arbitrarily choose a set of satisfying values, $\{z_1, \ldots, z_l\}$, corresponding to the set $\{y_1, \ldots, y_l\}$. Define $S \equiv \{\sum_{i=1}^{l} c_i z_i : c_i \in \mathbb{Z}_{m_i}\}$. We claim that

$\{\chi_z|_Q : z \in S\} = \text{Irr}(Q)$. Since $\{\chi_z : z \in S\} \subseteq \text{Irr}(G)$, it follows from Fact 2(5) that $\{\chi_z|_Q : z \in S\} \subseteq \text{Irr}(Q)$. Let $\sum_{i=1}^{l} c_i z_i$ and $\sum_{i=1}^{l} c_i' z_i$ be elements in S such that $c_p \neq c_p'$ for some $p \in \{1, \ldots, l\}$. Then $\chi_{\sum_{i=1}^{l} c_i z_i}|_Q(y_p) = \Pi_{i=1}^{l}(\chi_{z_i}(y_p))^{c_i} = (\zeta^{n_1/m_p})^{c_p} \neq (\zeta^{n_1/m_p})^{c_p'} = \Pi_{i=1}^{l}(\chi_{z_i}(y_p))^{c_i'} = \chi_{\sum_{i=1}^{l} c_i' z_i}|_Q(y_p)$. Since there are $m_1 \cdots m_l$ distinct l-tuples of the form $(c_1, \ldots, c_l)$, where $c_i \in \mathbb{Z}_{m_i}$, it follows that $|\{\chi_z|_Q : z \in S\}| = m_1 \cdot m_2 \cdots m_l = |Q| = |\text{Irr}(Q)|$. Thus $\{\chi_z|_Q : z \in S\} = \text{Irr}(Q)$, where $|S| = |\{\chi_z|_Q : z \in S\}| = |Q|$. By Lemma 10 S is a transversal of $Q^\perp$ since $H_{S,Q}$ is a character table for Q and consequently invertible.
Now, solving System (2) is equivalent to solving:

$$\begin{bmatrix} y_{11} & \cdots & y_{1i} & \cdots & y_{1k} \\ \vdots & & \vdots & & \vdots \\ y_{i-1,1} & \cdots & y_{i-1,i} & \cdots & y_{i-1,k} \\ y_{i1} & \cdots & y_{ii} & \cdots & y_{ik} \\ y_{i+1,1} & \cdots & y_{i+1,i} & \cdots & y_{i+1,k} \\ \vdots & & \vdots & & \vdots \\ y_{l1} & \cdots & y_{li} & \cdots & y_{lk} \end{bmatrix} \begin{bmatrix} \frac{n_1}{n_1} z_{i1} \\ \vdots \\ \frac{n_1}{n_i} z_{ii} \\ \vdots \\ \frac{n_1}{n_k} z_{ik} \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ \frac{n_1}{m_i} \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \tag{3}$$

where $z_i = (z_{i1}, \ldots, z_{ik}) \in \mathbb{Z}_{n_1} \oplus \cdots \oplus \mathbb{Z}_{n_k}$ for $1 \leq i \leq l$, and $y_j = (y_{j1}, \ldots, y_{jk}) \in \mathbb{Z}_{n_1} \oplus \cdots \oplus \mathbb{Z}_{n_k}$ for $1 \leq j \leq l$.

Note that if the system:

$$\begin{bmatrix} \frac{n_1}{n_1} y_{11} & \cdots & \frac{n_1}{n_k} y_{1k} \\ \vdots & & \vdots \\ \frac{n_1}{n_1} y_{i-1,1} & \cdots & \frac{n_1}{n_k} y_{i-1,k} \\ \frac{n_1}{n_1} y_{i1} & \cdots & \frac{n_1}{n_k} y_{ik} \\ \frac{n_1}{n_1} y_{i+1,1} & \cdots & \frac{n_1}{n_k} y_{i+1,k} \\ \vdots & & \vdots \\ \frac{n_1}{n_1} y_{l1} & \cdots & \frac{n_1}{n_k} y_{lk} \end{bmatrix} \begin{bmatrix} z_{i1} \\ \vdots \\ z_{ik} \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ \frac{n_1}{m_i} \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{4}$$

is solved over $\mathbb{Z}_{n_1}$ and $(z_{i1}, \ldots, z_{ik})$ is regarded as an element of $\mathbb{Z}_{n_1}^k$, then $(z_{i1} \bmod n_1, \ldots, z_{ik} \bmod n_k) \in \mathbb{Z}_{n_1} \oplus \cdots \oplus \mathbb{Z}_{n_k}$ is a solution to system (3). Thus to find a solution to (3) we need only find a solution to (4) over $\mathbb{Z}_{n_1}$. To this end, we adapt the algorithm in Section 2.1 for solving systems of modular equations. The method we employ may not produce all solutions to the system, but it will yield atleast one, if the system has a solution. As noted earlier, each of our systems has $|Q^\perp|$ solutions, therefore the method used will always yield atleast one such for each system.

By Theorem 6 of Section 2.1, each of the l systems given by equation (2) can be solved in time $\mathcal{O}(l^2 k^2 n_1^2 \log n_1)$. Thus time $\mathcal{O}(l^3 k^2 n_1^2 \log n_1) \leq \mathcal{O}(k^2 n_1^2 \log^3 |Q| \log n_1)$ is required to solve all l systems. To enumerate S requires time $\mathcal{O}(k|Q|)$, since $|S| = |Q|$. Therefore time $\max\{\mathcal{O}(k|Q|), \mathcal{O}(k^2 n_1^2 \log^3 |Q| \log n_1)\}$ is required to construct a transversal S of $Q^\perp$. □

The final theorem of the section follows directly from Theorems 11, 12, and Observation 9.

Theorem 13. *Let $G = \mathbb{Z}_{n_1} \oplus \cdots \oplus \mathbb{Z}_{n_k}$ be a finite Abelian group with invariant factors $n_1, \ldots, n_k$. If a subgroup Q of G is specified as a listing of its elements, then there exists an algorithm which produces the elements of a uniform, good sample set for $D_0^{G,Q}$ and subsequently recovers any $f \in D_0^{G,Q}$ in time bounded by $\mathcal{O}(k^2 n_1^2 |Q|^2 \log^3 |Q| \log n_1)$.*

Proof. The algorithm begins by receiving as input a listing of the elements in a subgroup Q of G. By Theorem 5, $\mathcal{O}(k^2|Q|^2 \log^2 |Q|)$ time is required to obtain the structure of Q. Once Q's structure is known, a uniform, good sample set for $D_0^{G,Q}$, namely S, can be constructed in time $\max\{\mathcal{O}(k|Q|), \mathcal{O}(k^2 n_1^2 \log^3 |Q| \log n_1)\}$ by Theorem 12. Once S is known, $H_{S,Q}$ and ultimately $H_{S,Q}^{-1}$ can be formulated in time $\mathcal{O}(k|Q|^2)$. Then for any $f \in D_0^{G,Q}$, f is sampled on S, requiring time $\mathcal{O}(|Q|)$, and finally the product

$$H_{S,Q}^{-1} \begin{bmatrix} f(x_1) \\ \vdots \\ f(x_{|Q|}) \end{bmatrix} = \begin{bmatrix} \hat{f}(q_1) \\ \vdots \\ \hat{f}(q_{|Q|}) \end{bmatrix}$$

is computed in time $\mathcal{O}(|Q|^2)$. Thus any $f \in D_0^{G,Q}$ can be recovered in time bounded by $\mathcal{O}(k^2 n_1^2 |Q|^2 \log^3 |Q| \log n_1)$. □

3.2 Unknown Basis Q

In this section we restrict the Abelian groups under consideration to those of the form $\mathbb{Z}_p^n$, for p is prime. Since $\mathbb{Z}_p$ is a field, we denote $\mathbb{Z}_p^n$ as $\mathbb{F}_p^n$ and regard it as a vector space over $\mathbb{F}_p$.

Our second main result deals with the efficient recovery of functions in $C_0^{G,Q}$ when $G = \mathbb{F}_p^n$ and Q is an unknown subspace. The only information that is available about the function is that it has only $|Q|$ nonzero Fourier coefficients which need to be found by sampling on an order of $|Q|$ points. This is a nonlinear interpolation question since Q is unknown, and hence there is *no fixed linear space of functions* in which the interpolant is known to reside. Moreover, we would like the resulting sample set to be a uniform, good sample set for all functions in $C_0^{G,Q}$. We first assume that as in Subsection 3.1, one task is to find good sample sets for functions $f \in C_0^{G,Q}$ using only a few point evaluations of f. However, since Q is unknown, this requires more work. In addition, further work is needed to recover f since it cannot be recovered straightforwardly by solving the equations in Observation 9.

The main ingredients in the proof are the two theorems 15 and 16. Theorem 15 explains how to obtain a good sample set for $f \in C_0^{\mathbb{F}_p^n,Q}$ in time $\mathcal{O}(n^{\log_p |Q|+1}|Q|^2)$ – and Theorem 16 explains how to recover f once a good sample set for it is known in time $\mathcal{O}(n^4|Q|^2)$. Before stating these theorems, we state a simple technical lemma which is used in their proofs. The proof of the lemma is a straightforward technical exercise.

Lemma 14. *Let p be a prime number, Q be a subspace of $\mathbb{F}_p^n$.*

1. *If S is a complement of $Q^\perp$, then $f \in C_0^{\mathbb{F}_p^n, Q}$ if and only if $f(x) = f(x+y)$, $\forall x \in S$ and $\forall y \in Q^\perp$.*
2. *Let $f \in C_0^{\mathbb{F}_p^n, Q}$. Let spt $\hat{f}$ be contained in no subspace smaller than $|Q|$. Then S is a subspace of $\mathbb{F}_p^n$ of size $|Q|$ with the property that there exists no nonzero $y \in S$ such that $f(x) = f(x+y)$, $\forall x \in S$, if and only if S is a complement of $Q^\perp$.*
3. *Let $f \in C_0^{\mathbb{F}_p^n, Q}$. Let $S = \{0^n = x_1, x_2, \ldots, x_{p^m}\}$ be a complement of $Q^\perp = \{0^n = y_1, y_2, \ldots, y_{p^{n-m}}\}$. If $\{w_1, \ldots, w_{n-m}\}$ is a linearly independent set such that $W^\perp \equiv span\{w_1, \ldots, w_{n-m}\}$ is a complement of S in $\mathbb{F}_p^n$ and for each $i \in \{1, \ldots, n-m\}$, $f(x) = f(w_i + x)$, $\forall x \in S$, then for each $w \in W^\perp$, $f(y) = f(w+y)$, $\forall y \in \mathbb{F}_p^n$.*

We are now ready for the first main ingredient in the proof of of the main result of this section: Theorem 17.

Theorem 15. *Let p be a prime number, Q be a subspace of $\mathbb{F}_p^n$ and $f \in C_0^{\mathbb{F}_p^n, Q}$ such that Q is the smallest subspace containing spt $\hat{f}$. If Q is unknown, but $|Q|$ is known, then a uniform, good sample set for $C_0^{\mathbb{F}_p^n, Q}$ can be found in time $\mathcal{O}(n^{\log_p |Q|+1}|Q|^2)$.*

Proof. If $|Q| = 1$ (i.e. $Q = \{0^n\}$), then we let $S = \{0^n\}$. Thus S is a transversal of $Q^\perp$ and consequently, by Theorem 11 and Observation 9, a good sample set for $f \in C_0^{\mathbb{F}_p^n, Q}$. Clearly, in this case S can be found within the required time. Therefore we assume $1 < |Q| = p^m \leq p^n$.

We begin by finding a subspace S of size $|Q|$ with the property that there exists no nonzero $y \in S$ such that $f(x) = f(x+y)$, $\forall x \in S$. In order to do so let $\{e_1, \ldots, e_n\}$ be the standard basis for $\mathbb{F}_p^n$, where e_i is the n-tuple whose only nonzero coordinate is the i^{th} with value 1. Since $|Q^\perp| < p^n$, $Q^\perp$ has a nonzero complement. Furthermore one of these complements must be spanned by m elements in the set $\{e_1, \ldots, e_n\}$. So we sequentially generate the subsets of $\{e_1, \ldots, e_n\}$ of size m until one is found such that its span (a subspace of size p^m), call it S, has the property that there exists no nonzero $y \in S$ such that $f(x) = f(x+y)$, $\forall x \in S$. Since, as noted above, some subset of $\{e_1, \ldots, e_n\}$ of size m spans a complement of $Q^\perp$, Lemma 14 (2) guarantees that the search for S will successfully terminate. The entire operation requires time $\mathcal{O}(n^{m+1}|Q|^2) = \mathcal{O}(n^{\log_p |Q|+1}|Q|^2)$ and produces a set S which is a complement of $Q^\perp$. As such, S is also a transversal of $Q^\perp$ and consequently a good sample set for $f \in C_0^{\mathbb{F}_p^n, Q}$. □

Next we proceed to the second ingredient in the proof of the main result of this section: Theorem 17. We show how to recover f given the sample set S without knowing Q, i.e, when a complement S of $Q^\perp$ is known. The proof uses several constructions. First a basis for S is assembled using Algorithm A below. With this set a basis for a complement of S is constructed using Algorithm B. In Algorithm C the set S and the basis for the constructed complement

are then used to find a basis for a set $W^{\perp}$, where $W^{\perp}$ is a complement of S and $f \in C_0^{\mathbb{F}_p^n, W}$. Using $W^{\perp}$'s basis, W is easily constructed and subsequently the invertible matrix $H_{S,W}$, as found in the standard interpolation system in Observation 9 is used to identify f's Fourier coefficients.

Theorem 16. *Let p be a prime. Let Q be a subspace of $\mathbb{F}_p^n$ and $S = \{0^n = x_1, x_2 \ldots, x_{p^m}\}$ be a complement of $Q^{\perp}$. Let $f \in C_0^{\mathbb{F}_p^n, Q}$. If S is known, but Q is not, then there exists an algorithm which recovers f exactly in time $\mathcal{O}(n^4|Q|^2)$.*

Proof. We begin by noting that $|S| = |Q|$. Hence if $|S| = 1$, then spt $\hat{f} \subseteq Q = \{0^n\}$. Since we now know Q, it is possible to recover f by solving the interpolation system in Observation 9 This recovery requires time $\mathcal{O}(|Q|)$ and the algorithm purported to exist most certainly does in this case. So we continue under the assumption that $|S| \geq 2$.

The construction of the overall algorithm begins with the creation of Algorithm A, an algorithm that forms a basis, B_S for S.

Algorithm A: Begin by inserting the nonzero elements of S into a queue. Suppose x_2 is at the top of the queue. Set $B_S = \{x_2\}$ and delete the nonzero elements of $\langle x_2 \rangle$ from the queue. Now suppose x_3 is at the top of the queue. Since $|\langle x_3 \rangle| = p$ and $x_3 \notin \langle x_2 \rangle$, it follows that $\langle x_2 \rangle \cap \langle x_3 \rangle = \langle 0 \rangle$. Thus $\langle x_2 \rangle \oplus \langle x_3 \rangle \leq S$. Set $B_S = \{x_2, x_3\}$ and delete the nonzero elements of $(\langle x_2 \rangle \oplus \langle x_3 \rangle) \setminus \langle x_2 \rangle$ from the queue. The algorithm continues in this fashion until we have $\langle x_2 \rangle \oplus \langle x_3 \rangle \oplus \cdots \oplus \langle x_{m+1} \rangle = S$ and $B_S = \{x_2, \ldots, x_{m+1}\}$, at which stage the algorithm terminates. Since each nonzero element has order p, the algorithm must reach this final stage. Furthermore the algorithm will execute in time $\mathcal{O}(n|Q|^2)$. Without loss of generality assume the basis produced by this algorithm is in fact $\{x_2, \ldots, x_{m+1}\} = B_S$.

Algorithm B is used to construct a basis for a complement to S in $\mathbb{F}_p^n$.

Algorithm B: Begin with the standard basis, $\{e_1, \ldots, e_n\}$ for $\mathbb{F}_p^n$, where e_i is the n-tuple whose every entry is 0, except the i^{th} entry, which is 1. If dim $S = n$, $\langle 0^n \rangle$ is a complement of S. If dim $S = m < n$, for each i, regard each of $x_2, \ldots, x_{m+1}, e_i$ as $n \times 1$ column vectors. Then the system:

$$[x_2 \cdots x_{m+1}\ e_i] \begin{bmatrix} c_1 \\ \vdots \\ c_{m+1} \end{bmatrix} = \mathbf{0}$$

can be solved over $\mathbb{F}_p$ in time at most $\mathcal{O}(n^3)$. If the solution to the above system is trivial, then the vectors $x_2, \ldots, x_{m+1}, e_i$ are linearly independent. In this case set B equal to $\{e_i\}$. Otherwise, if no such e_i exists, we continue as above solving the systems:

$$[x_2 \cdots x_{m+1}\ e_i\ e_{i+1}] \begin{bmatrix} c_1 \\ \vdots \\ c_{m+2} \end{bmatrix} = \mathbf{0}$$

and so on, eventually constructing a set of $n - m$ linearly independent standard basis vectors. Represent the set B just constructed by $\{b_1, \ldots, b_{n-m}\}$. Then span

B is a complement of S. The entire process requires at most time $\mathcal{O}(n^4)$.

The following claims justify Algorithm C. Their proofs are straightforward.

Claim 1: For each $i \in \{1, \ldots, n-m\}$ there exist elements $w_i \in S^c$ and $x_{b_i} \in S$ such that $w_i = b_i - x_{b_i}$ and $f(x) = f(w_i + x) = f((b_i - x_{b_i}) + x)$, $\forall x \in S$.

Claim 2: $W' \equiv \{w_1, \ldots, w_{n-m}\}$ is a linearly independent set.

Claim 3: span W' is a complement of S in $\mathbb{F}_p^n$.

With the above claims in place we are ready for Algorithm C.

Algorithm C: This algorithm has as its input $B = \{b_1, \ldots, b_{n-m}\}$, the output of Algorithm B and the given set $S = \{0^n = x_1, x_2, \ldots, x_{p^m}\}$. For each $i \in \{1, \ldots, n-m\}$ and for each $j \in \{1, \ldots, p^m\}$, set w_{ij} equal to $b_i - x_j$. Next compute $f(x)$ and $f(w_{ij} + x)$ for each $x \in S$ until $f(x) \neq f(w_{ij} + x)$ or the list of elements in S is exhausted. If $f(x) = f(w_{ij} + x)$ for every $x \in S$, place w_{ij} in the set W'. By Claim 1 such a w_{ij} exists for each $i \in \{1, \ldots, n-m\}$. Once such a w_{ij} is found, increment i and repeat the process until $i > n - m$. At the conclusion of the algorithm W' will contain $n - m$ linearly independent vectors in $\mathbb{F}_p^n$ (by Claim 2) such that span W' is a complement of S in $\mathbb{F}_p^n$ (by Claim 3). Dropping the second subscript of each element in W', we have $W' = \{w_1, \ldots, w_{n-m}\}$. The entire algorithm requires time at most $\mathcal{O}(n^2|Q|)$.

Let $W^\perp =$ span W' and let $W = (W^\perp)^\perp$. Use Algorithm D to compute W.

Algorithm D: Consider each of $w_1, \ldots, w_{n-m}$ to be a $1 \times n$ row vector, then solve the system:

$$\begin{bmatrix} w_1 \\ \vdots \\ w_{n-m} \end{bmatrix} \begin{bmatrix} c_1 \\ \vdots \\ c_n \end{bmatrix} = \mathbf{0}$$

Since the rank of $\begin{bmatrix} w_1 \\ \vdots \\ w_{n-m} \end{bmatrix}$ is $n - m$, the dimension of the solution set is m. In particular, the solution set is W and this space can be found and listed in time at most $\max\{\mathcal{O}(n^3), \mathcal{O}(n|Q|)\}$.

Since we now have a linearly independent set, $W' = \{w_1, \ldots, w_{n-m}\}$ (by Algorithm C) such that $W^\perp =$ span W' is a complement of S (by Claim 3) and for each $i \in \{1, \ldots, n-m\}$, $f(x) = f(w_i + x)$, $\forall x \in S$ (by Algorithm C), it follows directly from Lemma 14 (3) that for each $w \in W^\perp$, $f(x) = f(w + x)$, $\forall x \in S$. Thus by Lemma 14(1), $f \in C_0^{\mathbb{F}_p^n, W}$. Finally in order to recover f exactly we apply Algorithm E.

Algorithm E: This algorithm uses as input the subspace $W = \{z_1, \ldots, z_{p^m}\}$, which is the output of Algorithm D, as well as the given subspace, $S = \{0^n =$

$x_1, x_2, \ldots, x_{p^m}\}$ as its input. The algorithm then proceeds to solve the system:

$$H_{S,W} \begin{bmatrix} \hat{f}(z_1) \\ \vdots \\ \hat{f}(z_{p^m}) \end{bmatrix} = \begin{bmatrix} f(x_1) \\ \vdots \\ f(x_{p^m}) \end{bmatrix},$$

where $H_{S,W} = [\chi_{z_j}(x_i)]_{i,j \in \{1,\ldots,p^m\}}$ is invertible since S is a transversal of $W^\perp$, and to recover f exactly, i.e. $f = \sum_{j=1}^{p^m} \hat{f}(z_j)\chi_{z_j}$. The algorithm requires at most $\mathcal{O}(n|Q|^2)$ time.

In conclusion, note that Algorithms A-E can be accomplished in time at most $\mathcal{O}(n^4|Q|^2)$. Thus f can be exactly recovered in time polynomial in n and $|Q|$. □

We can now state the main theorem of this section.

Theorem 17. *Let p be a prime number, Q be a subspace of $\mathbb{F}_p^n$ and $f \in C_0^{\mathbb{F}_p^n, Q}$ such that Q is the smallest subspace containing spt $\hat{f}$. If Q is unknown, but $|Q|$ is known, then a uniform, good sample set for $C_0^{\mathbb{F}_p^n, Q}$ can be found and f can be recovered exactly in time $\max\{\mathcal{O}(n^4|Q|^2), \mathcal{O}(n^{\log_p |Q|+1}|Q|^2)\}$.*

Proof. The proof follows from Theorem 15 – which explains how to obtain a good sample set for $f \in C_0^{\mathbb{F}_p^n, Q}$ in time $\mathcal{O}(n^{\log_p |Q|+1}|Q|^2)$ – and Theorem 16, which explains how to recover f once a good sample set for it is known in time $\mathcal{O}(n^4|Q|^2)$. □

4 Applications

4.1 Learning and Pseudorandom generation

Background on Learning theory The results presented in this paper apply directly to computational learning theory, a well-established area of research, see [37], which concerns itself with computational models of learning. In our context, computational learning can be viewed as a study of how a student can *learn* a function in a given class, i.e, reach a stage where it can predict the value of a function on a random input, by using an algorithm (during an initial learning phase) that makes only a few *queries* to a teacher who provides values of f. The set of queries is sometimes called the *training set*. The construction of the algorithm uses knowledge of the class being learnt. The quantity "few" is in terms of the size of f's domain and the size of the support of its Fourier transform. Many learning algorithms use the so called PAC (*Probably Almost Correct*) learning model. This model is probabilistic in that the student has no control on the queries made to the teacher who has access to the values of f. The queries are randomly chosen from the domain over an arbitrary distribution D. A learning algorithm in this model outputs a *hypothesis* for f that agrees with f on all but a small, bounded measure of the domain, i.e, the hypothesis has bounded error, ϵ where the error is measured according to D (thus *Almost Correct*); and this error

bound this bound can be relied upon with a high probability that is bounded below, $1-\delta$. (thus Probably Almost Correct) [37]. The efficiency of the algorithm is based on the number of queries, as a (preferably polynomial) function of the size of the domain of f, the error ϵ and the degree of confidence $1-\delta$. Another well studied model called Angluin's exact learning model [1], where the queries are chosen by the student, and there is no hypothesis error. It is permitted for the algorithm to be randomized, so that the zero-error hypothesis is found with high probability, e.g, [43]. In [57] the authors formalized a natural, deterministic new model of learning called the AAC or Always Approximately Correct model, which is more general than in that it admits a degree of hypothesis error, but remains deterministic and allows the queries to be chosen by the student. Moreover, it introduced the notion of uniform, good training sets of inputs, as the basis of deterministic learning algorithms. These sample/training sets are good for *all* functions in the class being learnt.

Several previous learning algorithms for Boolean functions have exploited the (Fourier) spectral properties of functions in the class. These include the learning algorithms for AC^0 functions in [44], [20], [55], for decision trees in [43], for DNF formulae in [34], for monotone Boolean functions in [9], and recently, the authors considered other approximation classes of Boolean functions in [57]. All of these results deal with Boolean functions on the cube, or the Abelian group $\mathbb{Z}_2^n$. Extensions of some of these results for functions on other Abelian groups were given by [8]. These algorithms, in effect, deal with classes of Boolean functions f that are approximable by some linear combination g of few Fourier basis functions, with respect to a chosen norm, and the algorithms obtain such an approximation g as a hypothesis.

In some cases the set of Fourier basis functions that define the approximating class is fixed (and known to the learner), as in [44], [55], and [20], and in others, its size is fixed (and known to the learner), but the set itself is variable and left for the learner to decipher, as in [43] and [34].

Most of these algorithms (except [55], parts of [43], and [57]) use the PAC learning model, hence the algorithms are not deterministic, and they do not provide uniform, good sample sets.

Another important concept used in Learning theory is the Vapnik-Chervonenkis Dimension (VC-Dimension) [37] is a purely combinatorial measure of the complexity of a class of functions to be learned. This number is used to characterize the size of the sample set needed to learn a function, and is thereby an important tool used to decide whether a given learning algorithm is optimal or not. In our context, the VC-dimension of a class can be defined as follows. For any class $\mathcal{C}$ of Boolean functions over a domain X and any $T = \{x_1, \ldots, x_m\} \subseteq X$ define $\Pi_{\mathcal{C}}(T) = \{(f(x_1), \ldots, f(x_m)) : f \in \mathcal{C}\}$. If $\Pi_{\mathcal{C}}(T) = \{0,1\}^{|T|}$, then we say T is *shattered* by $\mathcal{C}$. Thus T is shattered by $\mathcal{C}$ if $\mathcal{C}$ realizes all possible dichotomies of T. The *VC-dimension* of $\mathcal{C}$, denoted $VCD(\mathcal{C})$, is the cardinality of the largest set T shattered by $\mathcal{C}$. If arbitrarily large finite sets can be shattered by $\mathcal{C}$, then $VCD(\mathcal{C}) = \infty$.

How results in this paper apply to Learning In this paper, we have *already* shown how to efficiently and deterministically learn functions in certain classes,

moreover, using uniform, good sample sets. In particular, Theorems 13 and 17 of this paper *directly* provides a significant extension of the learning results in [57] to classes of functions over arbitrary finite Abelian groups.

Furthermore the classes, e.g. $C_0^{G,Q}$, (Q known or unknown subgroup), are of the type which have been historically of interest in complexity theory, especially in the context of proving threshold and communication complexity lower bounds which are typically hard to obtain (e.g. [40], [41], [42] [48], [50], [25], [27], [26], and [28], [56]). See [55] for the strong relationship between learning and proving lower bounds.

The following result gives bounds on the VC-dimension of the classes discussed in this paper and also show that many of the sample sets obtained in Section 3 are not only uniform and good, but are in fact, optimal in size.

Theorem 18. *If Q is a subset of a finite Abelian group G, then $VCD(C_0^{G,Q}) \leq |Q|$. Let G be a finite Abelian group and Q be a subgroup of G, then $VCD(C_0^{G,Q}) = |Q|$.*

Furthermore, using the results of this paper, one can get deterministic learning results with uniform training/sample sets for classes beyond those discussed in the previous sections of this paper. Let us define the approximation class of functions approximable from the span of Fourier basis functions in a set Q, where the approximation is within ϵ in the ∞ norm. More formally, $C_\epsilon^{G,Q,\infty} = \{f : G \to \{0,1\} : \exists g \in D_0^{G,Q} \text{ with } \|f - g\|_\infty \leq \epsilon\}$. Most threshold complexity classes (mentioned above) are approximation classes of this type.

Suppose G is a finite Abelian group and Q is a subset of G containing 0. If $\epsilon \geq 1/2$, it is not hard to see that the constant function $1/2$ i.e, $1/2\chi_0$, approximates all Boolean functions to within $1/2$ in the max norm, and thus $C_\epsilon^{G,Q,\infty}$ consists of all Boolean functions over G, which is an intractable class. If $\epsilon < 1/2$ and $Q = \{q_1, \ldots, q_{|Q|}\}$ is a subgroup of G, it can be shown that $C_\epsilon^{G,Q,\infty} = C_0^{G,Q}$. This leaves us with the case where $\epsilon < 1/2$ and Q is an arbitrary subset of G.

As described in the Introduction, standard sampling methods from the DFT or FFT literature [49], [58] [43], [29], [13], [11] would treat this class as being similar to the class $C_\epsilon^{G,Q,2}$ (approximation in the 2-norm as opposed to the ∞-norm). These methods would adaptively (in case Q is unknown) use projections and domain pruning to pick out the largest $|Q|$ Fourier coefficients. It is however shown in [57] that there are *no* uniform, good, sample sets for $C_\epsilon^{G,Q,2}$. Thus none of these methods will work $C_\infty^{G,Q,2}$ although there *are* uniform, good sample sets for this class, as shown in the following theorem which is an extension of a result in [57], proved using a Booleanized, duality theorem from approximation theory (distinct from duality in group theory), to the case where Q is a transversal, yielding a uniform, AAC learning result.

Theorem 19. *Let G be a finite Abelian group. If Q is a transversal of some subgroup, call it $S^\perp$ of G, then S is a uniform training set (a set which can be used to learn any member of $C_\epsilon^{G,Q,\infty}$ within a given error bound) of size $|Q|$ for the class $C_\epsilon^{G,Q,\infty}$. In addition, $C_\epsilon^{G,Q,\infty}$ is AAC learnable in time $\mathcal{O}(|Q|^2)$ with hypothesis error bounded by $\mathcal{O}(|Q|\epsilon^2)$ in the 2-norm.*

Pseudorandom Generators Randomized computations are popular for dealing with deterministically intractable algorithmic problems. Since randomness is an expensive resource, it is desirable to reduce the number of truly random numbers or bits used by a randomized computation R. To achieve this, one requires an efficient pseudorandom generator which takes as input a small random "seed" of very few random bits, and outputs a longer pseudorandom string of n numbers. It is essential that the randomized computation R cannot distinguish (with respect to chosen moments/measures) the resulting pseudorandom string from a truly random string of n bits. It is useful if the same set of pseudorandom strings can be *uniformly* used for *all* randomized computations in a complexity class C.

By extending the results in [55] connecting learning, pseudorandom generation, and lower bounds, it can be shown that if the function embodying the computation R comes from the class $C_0^{G,Q}$ or the class $C_\epsilon^{G,Q,\infty}$ (defined in subsection 4.1), then an efficient pseudorandom generator is available that uses a random seed of size at most $\log|Q|$ truly random bits. The idea of this result is again based on a Booleanized duality theorem from approximation theory showing that the uniform sample sets described in Section 3 and the previous subsection serve as sets of pseudorandom strings for *all* computations (functions) from the corresponding approximation classes.

Open Problems The problem of learning $C_\epsilon^{G,Q,\infty}$ with uniform training/sample sets, or finding uniform sets of pseudorandom strings for computations in $C_\epsilon^{G,Q,\infty}$, for arbitrary (known or unknown) *subsets* Q of an abelian group G, is still open and it is highly related to the following two simply-stated problems of independent mathematical interest.

- Taking $G = \mathbb{Z}_2^n$, given a set $|Q| = 2^p$ rows of the character table of G (i.e, the $2^n \times 2^n$ Hadamard matrix H_n), find a set of columns S such that the submatrix $H_{S,Q}$ is invertible, and has small inverse max-norm, say bounded by $O(2^p)$, for all n.
- What can be said about the structure of the supports Q of the Fourier transforms of *Boolean* functions over an abelian group $G = Z_p^n$, given that $|Q|$ is significantly smaller compared to $|G|$. Utilizing the fact that the Fourier basis functions over $G = \mathbb{Z}_p^n$ can be viewed as complex polynomials over U_p^n, where U_p are the complex p^{th} roots of unity, and abelian group duality, the above question can be phrased as follows. What can be said about the structure of the support set (nonzeroes) Q of a unimodular polynomials over U_p^n? This general problem is well-known [45], [14], [6], [39], [47], [4], and has several applications.

4.2 Linearity Testing

It is useful to be able to determine whether or not a map from one group to another is a homomorphism without testing the map on every possible pair of inputs. The following results show that this test can be executed expeditiously if the maps in question belong to the class of functions discussed in this paper.

The phrase "linearity testing" has been used by the complexity community to mean probabilistic/statistical testing used to determine if a function is a homomorphism into the *multiplicative* group of complex numbers, i.e, to determine if the function is a character. These linearity tests [3] have been used extensively for program checking, probabilistically checkable proofs and showing nonapproximability of NP-Complete problems (see, for example, [7], [2]). These testing algorithms use a constant number of random queries to the function, and the emphasis is on the two sided confidence error bounds of the output hypothesis called the completeness and soundness measures. Our treatment of linearity testing has a different emphasis in that the algorithm is deterministic and correct. The goal is to minimize the *query complexity*, i.e, to use the minimum number of queries.

Let G and H be groups. A function $f : G \to H$ is said to be *linear* if for every $x, y \in G$, $f(x+y) = f(x) + f(y)$. Furthermore, if $S \subseteq G$, then $f|_S$ is said to be linear if for every $u, v \in S$, $f(u+v) = f(u) + f(v)$. Note: Here $u+v$ need not belong to S. For functions that fail to be linear there is a measure to gauge the degree of failure. Let G and H be groups and $f : G \to H$. Then $Err(f) \equiv \Pr_{x,y \in G}\{f(x+y) \neq f(x)+f(y)\}$. If $S \subseteq G$, define $Err(f|_S) \equiv \Pr_{u,v \in S}\{f(u+v) \neq f(u) + f(v)\}$. Note: Here $u+v$ need not belong to S.
As the following theorem describes, a test for linearity applied to a function, f in $D_0^{G,Q}$ or an algorithm that computes $Err(f)$ does not need to evaluate f on its entire domain.

Theorem 20. *Let G be a finite Abelian group and Q be a subgroup of G. Let $f \in D_0^{G,Q}$. If S is a transversal of $Q^\perp$, then $f|_S$ is linear if and only if f is linear; also $Err(f) = Err(f|_S)$. Let $G = \mathbb{Z}_{n_1} \oplus \ldots \oplus \mathbb{Z}_{n_k}$ be a finite Abelian group with invariant factors $(n_1, \ldots, n_k)$. Let Q be a subgroup of G given as a listing of elements. Let $f \in D_0^{G,Q}$. Then f can be tested for linearity, and $Err(f)$ can be computed in time at most $\mathcal{O}(k^2 n_1^2 |Q|^2 \log^3 |Q| \log n_1)$.*

Acknowledgments: The authors wish to thank Mark Lewis, Per Enflo and Chuck Gartland for the technical advice they so graciously gave during innumerable conversations.

References

1. Angluin, D.: Learning Regular Sets from Queries and Counterexamples. Information and Computation **75(2)**, 87-106 (1987)
2. Arora, S., Lund, C., Motwani, R., Sudan, M., Szegedy, M., Proof Verification and the Intractability of Approximation Problems. Proceedings of 33^{rd} IEEE Symposium on Foundations of Computer Science, 14-23, (1992)
3. M. Bellare, D. Coppersmith, J. Hastad, M. Kiwi, M. Sudan: Linearity Testing in Characteristic 2. Proceedings of the 36^{th} IEEE Symposium on Foundations of Computer Science, (1995)
4. Benke, G.: On the maximum modulus for a certain class of unimodular trigonometric polynomials. Recent Advances in Fourier Analysis and Applications, J.S. Byrnes, and J.F. Byrnes eds, Kluwer Academic Publishers, 83-100 (1990)
5. Ben-Or, M., Tiwari, P.: A Deterministic Algorithm for Sparse Multivariate Polynomial Interpolation. Proceedings of the 20^{th} Annual ACM Symposium on Theory of Computing, 301-309 (1988)

6. Björck, G.: Functions of modulus 1 on $\mathbb{Z}_n$ whose Fourier transforms have constant modulus, and cyclic n-roots. Recent Advances in Fourier Analysis and Applications, J.S. Byrnes, and J.F. Byrnes eds, Kluwer Academic Publishers, 131-140 (1990)
7. Blum, M., Luby, M., Rubinfeld, R.: Self-testing/Correcting with Applications to Numerical Problems. Proceedings of the 22^{nd} Annual ACM Symposium on Theory of Computing, 73-83, (1990)
8. Boneh, Dan: Learning Using Group Representations. Proceedings of COLT 1995, 418-426 (1995)
9. Bshouty, N. H., Tamon, C.: On the Fourier Spectrum of Monotone Functions. Proceedings of the 27^{th} Annual ACM Symposium on Theory of Computing, 219-399 (May 1995)
10. Bshouty, N., Mansour, Y.: Simple Learning Algorithms for Decision Trees and Multivariate Polynomials. Proceedings of the 36^{th} IEEE Symposium on the Foundations of Computer Science, 304-311 (1995)
11. Brigham, O.E.: The fast Fourier transform and applications. Prentice Hall (1988)
12. Bruck, J., Smolensky, R.: Polynomial threshold functions, AC^0 functions, and spectral norms. 31^{st} Ann. IEEE Symp. Foundations of CS (FOCS), 632-641 (1990)
13. Butzer, P.L.: Fourier analysis and approximation. Academic Press (1971).
14. Byrnes, J.S.: On polynomials with coefficients of modulus one. Bull. London Math. Soc. 12, 171-176 (1961)
15. Chou, T.-W.J., Collins, G.E.: Algorithms for the Solution of Systems of Linear Diophantine Equations. SIAM Journal of Computing **11**, 687-708 (1982)
16. Chui, C.K.: Wavelet analysis and applications (series) vols 1 and 2; An introduction to Wavelets and A tutorial in Theory and Applications. Academic Press (1992)
17. DeVore, R.A.: Nonlinear Approximation. Acta Numerica, pp. 51-150, (1998)
18. Dym, H., McKean, H. P.: Fourier Series and Integrals. Probability and Mathematical Statistics Series, Academic Press (1972)
19. Dummit, D.S., Foote, R.S.: Abstract Algebra, 1st ed. Englewood Cliffs, NJ: Prentice-Hall (1991)
20. Furst, M., Jackson, J., Smith, S.: Improved Learning of AC^0 Functions. 4^{th} Conference on Computational Learning Theory, 317-325 (1991)
21. Gathen, J.v-z., Gerhard, J.: Modern Computer Algebra. Cambridge University Press (1999)
22. Geddes, K.O., Czapor, S.R., Labahn, G.: Algorithms for Computer Algebra, Kluwer academic (1992)
23. M. Giesbrecht: Efficient Parallel Solution of Sparse Systems of Linear Diophantine Equations, Proceedings of the ACM International Symposium on Parallel Symbolic Computation (PASCO'97), ACM Press, 1-10, (1997)
24. M. Giesbrecht: Fast Computation of the Smith form of an integer matrix. Proceedings of the International Symposium on Symbolic and Algebraic Computation (ISSAC) ACM Press, 110-118, (1995)
25. Goldman, M., Hastad, J., Razborov, A. A.: Majority Gates vs. General Weighted Threshold Gates. Proceedings of the 32^{nd} IEEE Symposium on the Foundations of Computer Science, (1991)
26. Gotsman, C., Linial, N.: Equivalence of Two Problems on the Cube - A Note. Journal of Combinatorial Theory, Ser. A **61**, 142-146 (1992)
27. Grolmusz, V.: Harmonic Analysis, Real Approximation and Communication Complexity of Boolean Functions. Manuscript (1994)
28. Hajnal, A., Maass, W., Pudlák, P., Szegedy, M., Turán, G.: Threshold Circuits of Bounded Depth. Proceedings of the 28^{th} IEEE Symposium on Foundations of Computer Science, 99-110 (1987)
29. Higgins, J.R.: Sampling theory in Fourier and Signal analysis. Clarendon press; Oxford University Press (1996)

30. Iliopoulos, Costas S.: Worst-case Complexity Bounds on Algorithms for Computing the Canonical Structure of Finite Abelian Groups and the Hermite and Smith Normal Forms of an Integer Matrix. SIAM Journal of Computing **18(4)**, 658-669 (August, 1989)
31. Iliopoulos, Costas S.: Worst-case Complexity Bounds on Algorithms for Computing the Canonical Structure of Infinite Abelian Groups and Solving Systems of Linear Diophantine Equations. SIAM Journal of Computing **18(4)**, 670-678 (August, 1989)
32. Isaacs, I.M.: Algebra, A Graduate Course. Pacific Grove, CA: Brooks/Cole Publishing Co. (1994)
33. Isaacs, I.M.: Character Theory of Finite Groups. San Diego London : Academic Press, Inc. (1976)
34. Jackson, J.: An Efficient Membership Query Algorithm for Learning DNF with respect to the Uniform Distribution. Proceedings of the 35^{th} IEEE Symposium on Foundations of Computer Science, 42-53 (1994)
35. Kahane, J-P., Sur les polynomes a coefficients unimodulaires. Bull. London Math. Soc. 12, 321-342 (1960)
36. Kahane, J-P., Lemarie-Rieusset, P-G., Fourier series and Wavelets. Gordon and Breach Publishers (1995)
37. Kearns, Michael J., Vazirani, Umesh V.: An Introduction to Computational Learning Theory. Cambridge, Massachusetts: The MIT Press (1994)
38. Knuth, Donald E.: The Art of Computer Programming, Vol. 2, Seminumerical Algorithms, 2nd ed. Reading, Massachusetts: Addison-Wesley (1981)
39. Körner, T.: On a polynomial of J.S. Byrnes. Bull. London Math. Soc., 12, 219-224 (1980)
40. M. Krause, P. Pudlák: On the Power of Depth 2 Circuits with Threshold and Modulo Gates. Proceedings of the 26^{th} Symposium on Theory of Computing, 48-58, (1994)
41. M. Krause, P. Pudlák: On Computing Boolean Functions by Sparse Real Polynomials. Proceedings of the 36^{th} IEEE Symposium on Foundations of Computer Science, 682-691, (1995)
42. M. Krause, S. Waack: Variation Ranks of Communication Matrices and Lower Bounds for Depth Two Circuits having Symmetric Gates and Unbounded Fan-in. Proceedings of the 32^{nd} IEEE Symposium on Foundations of Computer Science, 777-782, (1991)
43. Kushilevitz, E., Mansour, Y.: Learning Decision Trees Using the Fourier Transform. Proceedings of the 32^{nd} IEEE Symposium on Foundations of Computer Science, 455-464 (1991)
44. Linial, N., Mansour, Y., Nisan, N.: Constant Depth Circuits, Fourier Transforms, and Learnability. Proceedings of the 30^{th} IEEE Symposium on Foundations of Computer Science, 574-579 (1989)
45. Littlewood, J. E., On the mean values of certain trigonometric polynomials. J. London Math. Soc. 36, 307-334 (1961)
46. Maple, A symbolic computation software package. Waterloo Maple Incorporated
47. Newman, D.J., Giroux, A.: Properties on the unit circle of polynomials with unimodular coefficients. Recent Advances in Fourier Analysis and Applications, J.S. Byrnes, and J.F. Byrnes eds, Kluwer Academic Publishers, 79-81 (1990)
48. Nisan, N., Szegedy, M.: On the Degree of Boolean Functions as Real Polynomials. Proceedings of the 24^{th} Annual ACM Symposium on Theory of Computing, 462-467 (1992)
49. Nussbaumer, H.J.: Fast Fourier Transform and Convolutional Algorithms. Springer-Verlag, Berlin, 1982.
50. Paturi, R.: On the Degree of Polynomials that Approximate Symmetric Boolean Functions. Proceedings of 24^{th} Annual ACM Symposium on Theory of Computing,

468-474 (1992)
51. Rudin, W., Fourier analysis on groups. Wiley Classics Library (1990)
52. Terras, A., Fourier analysis on finite groups and applications. Cambridge University Press (1999)
53. Schapire, R., Sellie, L.: Learning Sparse Multivariate Polynomials over a Field with Queries and Counterexamples. Proceedings of the 6th Workshop on Computational Learning Theory, 17-26 (1993)
54. Sims, Charles C.: Computation with Finitely Presented Groups. Cambridge University Press (1994)
55. Sitharam, M.: Pseudorandom Generators and Learning Algorithms for AC^0. Proceedings of ACM Symposium on Theory of Computing (STOC), 478-488 (1994); Computational Complexity (5), 248-266, (1995)
56. Sitharam, M.: Evaluating spectral norms for constant depth circuits with symmetric gates. Computational Complexity (6) 167-189 (1995)
57. Sitharam, M., Straney, T.: Derandomized Learning of Boolean Functions. Proceedings of 8^{th} International Workshop, ALT 97. Lecture Notes in Artificial Intelligence, Vol. 1316. Berlin Heidelberg New York: Springer (1997)
58. Walker, J.S.: Fast Fourier Transforms. Studies in Advanced Mathematics series, CRC press (1991)