Sparse Harmonic Transforms: A New Class of Sublinear-time Algorithms for Learning Functions of Many Variables

Bosu Choi*   Mark A. Iwen†   Felix Krahmer‡

Abstract

In this paper we develop fast and memory efficient numerical methods for learning functions of many variables that admit sparse representations in terms of general bounded orthonormal tensor product bases. Such functions appear in many applications including, e.g., various Uncertainty Quantification (UQ) problems involving the solution of parametric PDE that are approximately sparse in Chebyshev or Legendre product bases [10, 55]. We expect that our results provide a starting point for a new line of research on sublinear-time solution techniques for UQ applications of the type above which will eventually be able to scale to significantly higher-dimensional problems than what are currently computationally feasible.

More concretely, let $\mathcal{B}$ be a finite Bounded Orthonormal Product Basis (BOPB) of cardinality $|\mathcal{B}| = N$. Herein we will develop methods that rapidly approximate any function f that is sparse in the BOPB, that is, $f : \mathcal{D} \subset \mathbb{R}^D \to \mathbb{C}$ of the form

$$f(\boldsymbol{x}) = \sum_{b \in \mathcal{S}} c_b \cdot b(\boldsymbol{x})$$

with $\mathcal{S} \subset \mathcal{B}$ of cardinality $|\mathcal{S}| = s \ll N$.

Our method adapts the CoSaMP algorithm [50] to use additional function samples from f along a randomly constructed grid $\mathcal{G} \subset \mathbb{R}^D$ with universal approximation properties in order to rapidly identify the multi-indices of the most dominant basis functions in $\mathcal{S}$ component by component during each CoSaMP iteration. It has a runtime of just $(s \log N)^{\mathcal{O}(1)}$, uses only $(s \log N)^{\mathcal{O}(1)}$ function evaluations on the fixed and nonadaptive grid $\mathcal{G}$, and requires not more than $(s \log N)^{\mathcal{O}(1)}$ bits of memory. We emphasize that nothing about $\mathcal{S}$ or any of the coefficients $c_b \in \mathbb{C}$ is assumed in advance other than that $\mathcal{S} \subset \mathcal{B}$ has $|\mathcal{S}| \leq s$. Both $\mathcal{S}$ and its related coefficients c_b will be learned from the given function evaluations by the developed method.

For $s \ll N$, the runtime $(s \log N)^{\mathcal{O}(1)}$ will be less than what is required to simply enumerate the elements of the basis $\mathcal{B}$; thus our method is the first approach applicable

---
*Department of Mathematics, Michigan State University (`choibosu@msu.edu`)

†Department of Mathematics, and Department of Computational Mathematics, Science, and Engineering(CMSE), Michigan State University (`iwenmark@msu.edu`)

‡Department of Mathematics, Technische Universität München, Germany (`felix.krahmer@tum.de`)

in a general BOPB framework that falls into the class referred to as *sublinear-time.* This and the similarly reduced sample and memory requirements set our algorithm apart from previous works based on standard compressive sensing algorithms such as basis pursuit which typically store and utilize full intermediate basis representations of size $\Omega(N)$ during the solution process.

Keywords High-Dimensional Function Approximation · Sublinear-time Algorithms · Function Learning · Sparse Approximation · Compressive Sensing

Mathematics subject classification 65T40 · 68W25

1   Introduction

One encounters the problem of multivariate function integration, approximation, interpolation, and learning from a relatively small number of function evaluations in application areas ranging from computational physics to mathematical finance. A common class of examples in the Uncertainty Quantification (UQ) literature [60, 63], for example, involves the approximation of Quantities of Interest (QoI) that are assumed to be continuous functions of a potentially large number of parameters. Consequently, uncertainty in the input parameters leads to uncertainty in the QoI outputs which, in turn, necessarily depends on how the QoI behaves as a function of the input parameters. In order to understand the uncertainty in the QoI outputs one is therefore forced to approximate the QoI as a function. This typically requires multivariate function integration and interpolation, usually via quadrature methods [17], sparse grid approaches [7], or (quasi-)Monte Carlo methods [46, 8], depending on the number of parameters (i.e., variables) on which the QoI depends. In any case, all of these approaches typically must assume that the QoI is a highly smooth function of its input parameters in order to guarantee efficiency and accuracy, though smoothness alone cannot generally save one from the curse of dimensionality [34] (i.e., from an exponential sampling and runtime dependence on the number of function variables, D).

More recently, sparsity of the quantity of interest in a given Bounded Orthonormal Product Basis (BOPB) has been identified as an appropriate model assumption for UQ problems involving solutions of parametric elliptic partial differential equations [57, 10, 6, 55, 2, 1]. This observation allows for a formulation of the QoI approximation problem in the language of compressive sensing (CS), a paradigm introduced in the signal recovery literature in the early 2000's (starting with [20, 9], cf. [23] for a comprehensive introduction to the field). Namely, when the function evaluations are performed for random choices of the input parameters, the problem is a special case of the problem of recovering a function that admits a sparse representation in a Bounded Orthogonal System from a

2

small number of function evaluations. This problem in general terms, that is, without assuming a product structure as we encounter it here, has been of interest to the CS community almost from the beginning (see, e.g., [54, 56, 44]).

Building on these general results, a number of more recent works have studied the same questions specifically for the important case of multivariate functions which exhibit sparsity in high-dimensional Chebyshev and Legendre product bases (see. e.g., [55, 10]). These methods still store and utilize full intermediate basis representations during the solution process, however. In order for the problems to still be feasible, they often make additional assumptions on the structure of the sparsity which imply that the degrees of the polynomial basis functions with large coefficients are relatively small. This has the effect of reducing the overall sampling complexity and size of the basis which, in turn, allows for faster approximation of the QoI function with less required memory. A simple example is the case where a function of a very large number of variables is assumed to actually only depend on a small subset of them [19, 28] (see also [55, 6] which achieve a similar effect in the UQ setting when $D \gg s$ via a combination of Petrov-Galerkin approximation and weighted ℓ_1 minimization techniques). Methods which assume hyperbolic cross [59, 21] or lower set [10] structures on the energetic basis function indexes provide additional examples.

The connection between UQ and BOPB-sparse function recovery established in these recent works paves the way for us to devise the first sublinear-time compressive sensing methods for general BOPB frameworks in this paper. More precisely, we are able to decouple the runtime and memory requirements necessary in order to learn a given BOPB-sparse function from the overall BOP basis size one must initially consider. In short, we develop extremely fast and memory efficient compressive sensing algorithms for such problems. Besides an improved theoretical performance, also the empirical performance of our method improves over previous approaches. In particular, the enhanced memory efficiency allows us to tackle much larger problem sizes than in previous works. We expect that the results presented in this paper will trigger follow-up works on sublinear-time solution techniques for UQ applications of the type above which will eventually be able to scale to significantly higher-dimensional problems than what are currently computationally feasible.

Though its focus is on the recovery of functions which exhibit sparsity in an arbitrary BOPB, the method developed herein is a direct descendant of previously existing sublinear-time compressive sensing algorithms developed in the mathematics and computer science communities for data stream processing and sketching applications [24, 27, 25, 41, 30]. Unlike these previous methods, however, the compressive sensing matrices we are forced to use herein are necessarily solely derived from highly structured combinations of Bounded Orthonormal System (BOS) sampling matrices (see §2.3 for

3

details). As a result, our recovery algorithm cannot make direct use of any of the group testing and random hashing techniques commonly utilized by such sublinear-time compressive sensing methods. Instead, we appeal to compressive sensing results concerning the restricted isometry constants of random sampling matrices derived from a BOS in order to develop general energy-based hashing techniques which capitalize on the tensor product basis structure of any given BOP basis $\mathcal{B}$. These new energy hashing techniques are then used to rapidly identify a given BOPB-sparse function's support set $\mathcal{S} \subset \mathcal{B}$ using the algorithms discussed in Sections 3.1 and 3.2.

Similarly, the sublinear-time compressive sensing method developed herein can also be viewed as a significantly generalized high dimensional Sparse Fourier Transform (SFT) algorithm [40, 36, 43, 12, 52, 53, 62, 48]. In particular, the support identification techniques developed for arbitrary BOP bases in Sections 3.1 and 3.2 bear a high-level resemblance to the dimension incremental support identification techniques recently proposed by both Potts and Volkmer et al. [52, 42] and Choi et al. [12, 13] for the multivariate Fourier basis (see §3.2.2 for more details). Unlike the method proposed herein, however, the aforementioned high dimensional SFT's all use the specific structure of the Fourier basis in fundamental ways which makes their results difficult to directly extend to general BOP bases. Furthermore, with the notable exception of [40, 48], none of them provide universal recovery guarantees for all Fourier compressible functions. As a result, we need to develop entirely new sublinear-time support identification methods which only depend on general BOP basis structure herein. For a more detailed discussion of how the SFT results yielded as special cases of the main result herein compare to previous SFT methods for the Fourier basis, we refer the reader to §1.3 below.

1.1 The Compressive Sensing Problem for BOPB-Sparse Functions

Toward a more exact problem formulation, let $p \in \mathbb{N}$ be any natural number and $[N] := \{0, 1, 2, \ldots, N-1\}$ for all $N \in \mathbb{N}$. The set of functions, $\{T_k : \mathcal{D} \to \mathbb{C}\}_{k \in [N]}$ forms a *Bounded Orthonormal System* (BOS) with respect to a probability measure σ over $\mathcal{D} \subset \mathbb{R}^p$ with BOS constant $K := \max_{k \in [N]} \|T_k\|_\infty \geq 1$ if $K < \infty$, and

$$\langle T_k, T_l \rangle_{(\mathcal{D}, \sigma)} := \int_{\mathcal{D}} T_k(\boldsymbol{x}) \overline{T_l(\boldsymbol{x})} d\sigma(\boldsymbol{x}) = \delta_{k,l} = \begin{cases} 1 & \text{if } k = l \\ 0 & \text{if } k \neq l \end{cases}$$

holds for all $k, l \in [N]$. Now let $\mathcal{B}_j := \{T_{j,k} : \mathcal{D}_j \to \mathbb{C}\}_{k \in [M]}$ form a BOS with respect to a probability measure ν_j on $\mathcal{D}_j \subset \mathbb{R}$, with constant $\widetilde{K}_j$ for each $j \in [D]$. Then, the BOPB functions $\mathcal{B} := \{T_{\boldsymbol{n}} : \mathcal{D} \to \mathbb{C}\}_{\boldsymbol{n} \in [M]^D}$, defined by

$$T_{\boldsymbol{n}}(\boldsymbol{x}) := \prod_{j \in [D]} T_{j;n_j}(x_j) \tag{1.1}$$

4

again form a BOS with constant

$$K := \max_{\boldsymbol{n} \in [M]^D} ||T_{\boldsymbol{n}}||_\infty = \prod_{j \in [D]} \widetilde{K}_j$$

with respect to the probability measure $\boldsymbol{\nu} := \otimes_{j \in [D]} \nu_j$ over $\mathcal{D} := \times_{j \in [D]} \mathcal{D}_j \subset \mathbb{R}^D$. Throughout this paper, we assume for simplicity that $T_{j;0} \equiv 1$ for all j. This assumption is true for the large class of orthonormal polynomials including Trigonometric polynomials, Chebyshev polynomials, Legendre polynomials, Gegenbauer polynomials, Jacobi polynomials, etc.

Herein we consider BOPB-sparse functions $f : \mathcal{D} \to \mathbb{C}$ of the form

$$f(\boldsymbol{x}) := \sum_{\boldsymbol{n} \in \mathcal{S} \subset \mathcal{I} \subseteq [M]^D} c_{\boldsymbol{n}} T_{\boldsymbol{n}}(\boldsymbol{x}) \tag{1.2}$$

where $|\mathcal{S}| = s \ll |\mathcal{I}| \leq |\mathcal{B}| = N = M^D$. Following [19, 55, 6] we will take $\mathcal{I}$ to be the subset of $[M]^D$ containing at most $d \leq D$ nonzero entries.

Considering the recovery of f using standard compressive sensing methods [20, 23] when $\mathcal{I} = [M]^D$, one can simply independently draw m'_1 points, $\mathcal{G}^E := \left\{ \boldsymbol{t}_1, \ldots, \boldsymbol{t}_{m'_1} \right\} \subset \mathcal{D}$, according to $\boldsymbol{\nu}$ and then sample f at those points to obtain

$$\boldsymbol{y}^{\mathbf{E}} = f\left(\mathcal{G}^E\right) := \left(f\left(\boldsymbol{t}_1\right), f\left(\boldsymbol{t}_2\right), \ldots, f\left(\boldsymbol{t}_{m'_1}\right) \right)^T \in \mathbb{C}^{m'_1}. \tag{1.3}$$

Our objective becomes the recovery of f using only the samples $\boldsymbol{y}^{\mathbf{E}}$.

Let the $m'_1 \times M^D$ *random sampling matrix* $\Phi \in \mathbb{C}^{m'_1 \times M^D}$ have entries given by

$$\Phi_{\ell, \boldsymbol{n}} = T_{\boldsymbol{n}}(\boldsymbol{t}_\ell). \tag{1.4}$$

We can now form the underdetermined linear system

$$\boldsymbol{y}^{\mathbf{E}} = \begin{pmatrix} f(\boldsymbol{t}_1) \\ f(\boldsymbol{t}_2) \\ \vdots \\ f\left(\boldsymbol{t}_{m'_1}\right) \end{pmatrix} = \begin{pmatrix} T_{\boldsymbol{n}_1}(\boldsymbol{t}_1) & T_{\boldsymbol{n}_2}(\boldsymbol{t}_1) & \cdots & \cdots & T_{\boldsymbol{n}_{M^D}}(\boldsymbol{t}_1) \\ T_{\boldsymbol{n}_1}(\boldsymbol{t}_2) & T_{\boldsymbol{n}_2}(\boldsymbol{t}_2) & \cdots & \cdots & T_{\boldsymbol{n}_{M^D}}(\boldsymbol{t}_2) \\ \vdots & \vdots & \ddots & & \vdots \\ T_{\boldsymbol{n}_1}(\boldsymbol{t}_{m'_1}) & T_{\boldsymbol{n}_2}(\boldsymbol{t}_{m'_1}) & \cdots & \cdots & T_{\boldsymbol{n}_{M^D}}(\boldsymbol{t}_{m'_1}) \end{pmatrix} \boldsymbol{c} = \Phi \boldsymbol{c},$$

where $\boldsymbol{c} \in \mathbb{C}^{M^D}$ contains the basis coefficients $c_{\boldsymbol{n}}$ of f, and the index vectors $\boldsymbol{n}_1, \ldots, \boldsymbol{n}_{M^D} \in [M]^D$ are ordered, e.g., lexicographically. Note that this linear system is woefully underdetermined when $m'_1 \ll M^D$. When $\boldsymbol{c}$ has only $s \ll M^D$ nonzero entries as it does here, however, the compressive sensing literature tells us that $\boldsymbol{c}$ can still be recovered using significantly fewer than M^D function evaluations as long as the normalized random sampling matrix Φ has the *Restricted Isometry Property* (RIP) of order $2s$ [23].

Definition 1 (See Definition 6.1 in [23]). *The sth restricted isometry constant δ_s of a matrix $\widetilde{\Phi} \in \mathbb{C}^{m \times N}$ is the smallest $\delta \geq 0$ such that*

$$(1 - \delta)\|\boldsymbol{c}\|_2^2 \leq \left\|\widetilde{\Phi}\boldsymbol{c}\right\|_2^2 \leq (1 + \delta)\|\boldsymbol{c}\|_2^2$$

holds for all s-sparse vectors $\boldsymbol{c} \in \mathbb{C}^N$. The matrix $\widetilde{\Phi}$ is said to satisfy the RIP of order s if $\delta_s \in (0, 1)$.

Furthermore, one can show that random sampling matrices have the restricted isometry property with high probability even when $\left|\mathcal{G}^E\right|$ is relatively small.

Theorem 1 (See Theorem 12.32 and Remark 12.33 in [23]). *Let $A \in \mathbb{C}^{m \times N}$ be the random sampling matrix associated to a BOS with constant $K \geq 1$. If, for $\delta, p \in (0, 1)$,*

$$m \geq aK^2\delta^{-2}s \cdot \max\{\log^2(4s)\log(8N)\log(9m), \log(p^{-1})\},$$

then with probability at least $1 - p$, the restricted isometry constant δ_s of $\widetilde{A} = \frac{1}{\sqrt{m}}A$ satisfies $\delta_s \leq \delta$. The constant $a > 0$ is universal.

Note that Theorem 1 effectively decouples the number of samples that one must acquire/compute in order to recover any BOPB-sparse f from the overall BOP basis size $|\mathcal{B}| = M^D$. It guarantees that a random sampling set of size $\left|\mathcal{G}^E\right| = m_1' = \mathcal{O}(K^2 \cdot s \cdot D \cdot \log^4(KMD))$ suffices. The main obstacle to reducing the sampling complexity (i.e., m_1') at this point becomes the BOS sampling constant K. To see why, consider, e.g., the cosine BOPB where for all $j \in [D]$ in (1.1) we set $T_{j;n}(x) = \sqrt{2}\cos(nx)$ for $n \geq 1$ and $T_{j;0}(x) = 1$ in (1.1). This leads to a BOS with $K = 2^{D/2}$ with respect to uniform probability measure $\boldsymbol{\nu}$ over $\mathcal{D} = [0, 2\pi]^D$. Now we can see that we still face the curse of dimensionality since $K^2 = 2^D$ even for this fairly straightforward BOPB. Nonetheless, it expresses itself in a dramatically reduced fashion: 2^D is still a vast improvement over M^D for even moderately sized $M > 2$.

As previously mentioned, to further reduce the sampling complexity from scaling like $2^{\mathcal{O}(D)}$ previous work has focussed on developing efficient methods for effectively reducing the basis size to a smaller subset of the total basis $\mathcal{B}$ (see, e.g., [10, 55]). To see how this might work in the context of our simple cosine BOPB example above, we can note that the BOPB elements in (1.1) can be rewritten as

$$T_{\boldsymbol{n}}(\boldsymbol{x}) := 2^{\|\boldsymbol{n}\|_0/2} \prod_{j=0}^{D-1} \cos(n_j x_j)$$

in that case. It now becomes obvious that limiting the basis functions to those with indexes in $\mathcal{I} := \{\boldsymbol{n} \in [M]^D \mid \|\boldsymbol{n}\|_0 \leq d \leq D\}$ leads to a reduced BOS constant of

$K = 2^{d/2} \leq 2^{D/2}$ for the resulting reduced basis, as well as to a smaller basis cardinality of size $\binom{D}{d} M^d = \mathcal{O}\left(\left(\frac{DM}{d} \right)^d \right)$.

In particular, the utility of the assumption that the s non-negligible basis indexes of f, $\mathcal{S} \subset [M]^D$, also belong to the reduced index set $\mathcal{I}$ above is supported in some UQ applications where it is known that, e.g., the solutions of some parametric PDE are not only approximately sparse in some BOP bases such as the Chebyshev or Legendre product bases, but also that most of their significant coefficients correspond to index vectors $\boldsymbol{n} \subset \mathbb{N}^D$ with relatively small (weighted) ℓ_p-norms [10, 55]. In certain simplified situations this essentially implies that $\mathcal{S} \subset \mathcal{I}$ as discussed above. As a result, we will assume throughout this paper that $\mathcal{S} \subset \mathcal{I}$ so that $N = |\mathcal{I}| = \binom{D}{d} M^d \leq M^D$.[1]

Even when $s \cdot K^2 \ll N$ so that the number of required samples m_1' is small compared to the reduced basis size $|\mathcal{I}| = N$, however, all existing standard compressive sensing approaches for recovering f still need to compute and store potentially fully populated intermediate coefficient vectors $\boldsymbol{c}' \in \mathbb{C}^N$ at some point in the process of recovering f. As a result, all existing approaches are limited in terms of the reduced basis sizes $\mathcal{I}$ they can consider by both their memory needs and runtime complexities. In this paper we develop new methods that are capable of circumventing these memory and runtime restrictions for a general class of practical BOP bases.[2] As a result, we make it possible to recover a new class of extremely high-dimensional BOPB-sparse functions which are simply too complicated to be approximated by other means. We are now prepared to discuss our main results.

1.2   Main Results

The proposed sublinear-time algorithm is a greedy pursuit method motivated by CoSaMP[50], HTP[22], and their sublinear-time predecessors [24, 27]. In particular, it is obtained from CoSaMP by replacing CoSaMP's support identification procedure with a new sublinear-time support identification procedure. See Algorithm 1 in Section 3 for pseudocode and other details. Our main result demonstrates the existence of a relatively small grid of points $\mathcal{G} \subset \mathcal{D}$ which allows Algorithm 1 to recover any given BOPB-sparse function f in sublinear-time from its evaluations on $\mathcal{G}$. We refer the reader to Section 1.4 for a detailed

---

[1]Additionally, we will occasionally assume that our total grid size $|\mathcal{G}|$ below always satisfies $|\mathcal{G}| \leq N^c$ for some absolute constant $c \geq 1$ in order to simplify some of the logarithmic factors appearing in our big-O notation. This will certainly always be the case for any standardly used (trigonometric) polynomial BOPB (such as Fourier and Chebyshev product bases) whenever $sKDM < N$.

[2]Here we note that preconditioning and well chosen sampling distributions are crucial for many BOP bases. For example, the BOS constant for the standard Legendre basis is $K = \sqrt{2M+1}^D$ which implies that a naive application of Theorem 1 may require more than M^D (or M^d) samples. However, preconditioning can effectively reduce this BOS constant to $K = \sqrt{3}^d$ in practice [56].

description of the grid set $\mathcal{G}$ and its use in Algorithm 1. The following theorem is a simplified version of Theorem 2 in Section 3.

Theorem (Main Result). *Suppose that $\{T_{\boldsymbol{n}} \mid \boldsymbol{n} \in \mathcal{I} \subseteq [M]^D\}$ is a BOS where each basis function $T_{\boldsymbol{n}}$ is defined as per (1.1), and $T_{j;0} \equiv 1$ for all $j \in [D]$. Let $\mathcal{F}_s$ be the subset of all functions $f \in \operatorname{span}\{T_{\boldsymbol{n}} \mid \boldsymbol{n} \in \mathcal{I}\}$ whose coefficient vectors are s-sparse, and let $\boldsymbol{c}_f \in \mathbb{C}^{\mathcal{I}}$ denote the s-sparse coefficient vector for each $f \in \mathcal{F}_s$. Fix $p \in (0, 1/3)$, a precision parameter $\eta > 0$, $1 \leq d \leq D$, and $K = \sup\limits_{\boldsymbol{n} \in [M]^D \ s.t. \|\boldsymbol{n}\|_0 \leq d} \|T_{\boldsymbol{n}}\|_\infty$. Then, one can randomly select a set of i.i.d. Gaussian weights $\mathcal{W} \subset \mathbb{R}$ for use in (4.1), and also randomly construct a compressive sensing grid, $\mathcal{G} \subset \mathcal{D}$, whose total cardinality $|\mathcal{G}|$ is $\mathcal{O}\left(\left(sD\mathcal{L}'K^2 + s^3DK^4\right) \max\left\{d^4 \log^4(s) \log^4(D^2M), \log^2(\frac{D}{p})\right\}\right)$, such that the following property holds $\forall f \in \mathcal{F}_s$ with probability greater than $1 - 3p$:*

> *Let $\boldsymbol{y} = f(\mathcal{G})$ consist of samples from $f \in \mathcal{F}_s$ on $\mathcal{G}$. If Algorithm 1 is granted access to $\boldsymbol{y}$, $\mathcal{G}$, and $\mathcal{W}$, then it will produce an s-sparse approximation $\boldsymbol{a} \in \mathbb{C}^{\mathcal{I}}$ s.t.*
>
> $$\|\boldsymbol{c}_f - \boldsymbol{a}\|_2 \leq C\eta,$$
>
> *where $C > 0$ is an absolute constant.*

Furthermore, the total runtime complexity of Algorithm 1 is always
$$\mathcal{O}\left(\left(\left(s^2D^2\mathcal{L}K^2 + s^5D^2K^4\right) \max\left\{d^4 \log^4(s) \log^4(D^2M), \log^2(\frac{D}{p})\right\}\right) \times \log \frac{\|\boldsymbol{c}_f\|_2}{\eta}\right).$$

Note that Algorithm 1 will run in sublinear-time whenever $s^5D^2\mathcal{L}K^4d^4 \ll |\mathcal{I}|$ (neglecting logarithmic factors). Here and in the theorem above the parameters $\mathcal{L}$ and $\mathcal{L}'$ depend on your choice of numerical method for computing the inner product between a sparse function in the span of each one-dimensional BOS $\mathcal{B}_j = \{T_{j;m} \mid m \in [M]\}$. More specifically, let $\mathcal{L}'_j$ represent the number of function evaluations one needs in order to compute all M-inner products $\{\langle g, T_{j;\tilde{n}}\rangle\}_{\tilde{n} \in [M]}$ in $\mathcal{O}(\mathcal{L})$-time for any given function $g : \mathcal{D}_j \to \mathbb{C}$ belonging to the span of $\mathcal{B}_j$ that is also s-sparse in $\mathcal{B}_j$. We then set $\mathcal{L}' := \max_{j \in [D]} \mathcal{L}'_j$. For example, if each BOS $\mathcal{B}_j$ consists of orthonormal polynomials whose degrees are all bounded above by M then quadrature rules such as Gaussian quadrature or Chebyshev quadrature give $\mathcal{L} = \mathcal{O}(M^2)$ and $\mathcal{L}' = \mathcal{O}(M)$ [17]. If each $\mathcal{B}_j$ is either the standard Fourier, sine, cosine, or Chebyschev basis then the Fast Fourier Transform (FFT) can always be used to give $\mathcal{L} = \mathcal{O}(M \log M)$ and $\mathcal{L}' = \mathcal{O}(M)$ [17].

Moreover, there are several sublinear-time sparse Fourier transforms as well as sparse harmonic transforms for other bases which could also be used to give other valid $\mathcal{L}'$ and $\mathcal{L}$ combinations [29, 5, 47, 26, 37, 38, 4, 33, 35, 39, 58, 40, 15]. These typically have $\mathcal{O}(s^c \log^{c'} M)$ runtime and sampling complexities for small positive absolute constants c and c'. As a result, one can obtain much stronger results than the main theorem above

when $s \ll M$ and every one-dimensional BOS $\mathcal{B}_j$ is either the Fourier, sine, cosine, or Chebyshev basis. The following corollary of our main theorem is obtained by using deterministic one-dimensional SFT results from [40] and [35] in order to compute all of the nonzero inner products in lines 6 – 13 of Algorithm 2. They lead to $\mathcal{L}'$ and $\mathcal{L}$ values in Section 3's Theorem 2 of size $\mathcal{O}(s^2 \log^4 M)$.

Corollary 1. *Suppose that* $\left\{ T_{\boldsymbol{n}} \mid \boldsymbol{n} \in \mathcal{I} \subseteq [M]^D \right\}$ *is a BOS where each basis function* $T_{\boldsymbol{n}}$ *is defined as per* (1.1), *and where every one-dimensional BOS* $\mathcal{B}_j$ *is either the Fourier, sine, cosine, or Chebyshev basis. Let* $\mathcal{F}_s$ *be the subset of all functions* $f \in \mathrm{span} \left\{ T_{\boldsymbol{n}} \mid \boldsymbol{n} \in \mathcal{I} \right\}$ *whose coefficient vectors are s-sparse, and let* $\boldsymbol{c}_f \in \mathbb{C}^{\mathcal{I}}$ *denote the s-sparse coefficient vector for each* $f \in \mathcal{F}_s$. *Fix* $p \in (0, 1/3)$, *a precision parameter* $\eta > 0$, $1 \leq d \leq D$, *and let* $K = \sup_{\boldsymbol{n} \in [M]^D \ s.t. \|\boldsymbol{n}\|_0 \leq d} \|T_{\boldsymbol{n}}\|_\infty$. *Then, one can randomly select a set of i.i.d. Gaussian weights* $\mathcal{W} \subset \mathbb{R}$ *for use in* (4.1), *and also randomly construct a compressive sensing grid,* $\mathcal{G} \subset \mathcal{D}$, *whose total cardinality* $|\mathcal{G}|$ *is*
$\mathcal{O}\left(s^3 D \log^4(M) K^4 \max\left\{ d^4 \log^4(s) \log^4(D^2 M), \log^2(\frac{D}{p}) \right\} \right)$, *such that the following property holds* $\forall f \in \mathcal{F}_s$ *with probability greater than* $1 - 3p$:

> *Let* $\boldsymbol{y} = f(\mathcal{G})$ *consist of samples from* $f \in \mathcal{F}_s$ *on* $\mathcal{G}$. *If Algorithm 1 is granted access to* $\boldsymbol{y}$, $\mathcal{G}$, *and* $\mathcal{W}$, *then it will produce an s-sparse approximation* $\boldsymbol{a} \in \mathbb{C}^{\mathcal{I}}$ *s.t.*
>
> $$\|\boldsymbol{c}_f - \boldsymbol{a}\|_2 \leq C\eta,$$
>
> *where* $C > 0$ *is an absolute constant.*

Furthermore, the total runtime complexity of Algorithm 1 is always
$\mathcal{O}\left(\left(s^5 D^2 \log^4(M) K^4 \max\left\{ d^4 \log^4(s) \log^4(D^2 M), \log^2(\frac{D}{p}) \right\} \right) \times \log \frac{\|\boldsymbol{c}_f\|_2}{\eta} \right)$.

Note that the runtime dependance achieved by the corollary above scales sublinearly with M, quadratically in D, and at most polynomially in the parameter $d \leq D$ used to determine $\mathcal{I}$. We also remind the reader that the BOS constant K for the Fourier basis is 1. As a result, the K dependence in the runtime complexity vanishes entirely when the BOPB in question is the multidimensional Fourier basis.[3] Finally, there are also sublinear-time sparse transforms for one-dimensional Legendre polynomial systems [35], though the theoretical results for sparse recovery therein require additional support restrictions beyond simple sparsity. Thus, Corollary 1 can also be extended to restricted types of Legendre-sparse functions in order to achieve sublinear-in-M runtimes. A detailed development of such results is left for future work, however.

---

[3]Though the resulting $\mathcal{O}\left(s^5 D^2 d^4 \mathrm{polylog}(MDs\|\boldsymbol{c}\|_2/\eta p) \right)$-runtime achieved by Corollary 1 for the multidimensional Fourier basis is strictly worse than the best existing noise robust and deterministic sublinear-time results for that basis [40] (except perhaps when $s^3 d^4 \ll D^2$), we emphasize that it is achieved with a different and significantly less specialized grid $\mathcal{G}$ herein.

SFT Method	Runtime Complexity	Sampling Complexity	$\mathcal{I} \subseteq [M]^D$	MB	Error Guarantees
Corollary 1	$s^5 d^4 D^2 \cdot \log^5(s)\log^8(DM)$	$s^3 d^4 D \cdot \log^4(s)\log^8(DM)$	Any	Yes	Uniform Exact
Thm 8 in [40]	$s^2 D^4 \cdot \log^4(MD)$	$s^2 D^4 \cdot \log^4(MD)$	$[M]^D$	No	Det. Uniform Best-s Term
Thm 12 in [48]	$s^2 M^2 D^3 \cdot \log(M)$	$s^2 M D^3 \cdot \log(M)$	$[M]^D$	No	Det. Uniform Best-s Term
Thm 8 in [40]	$s D^4 \cdot \log^4(MD)$	$s D^4 \cdot \log^4(MD)$	$[M]^D$	No	Nonuniform Best-s Term
Thm 3.5 [43]	$s 2^{\mathcal{O}(D^2)} D^{D+3} \cdot \log^{D+3} M$	$s 2^{\mathcal{O}(D^2)} D \cdot \log M$ *	$[M]^D$	No	Nonuniform Best-s Term
Thm 4.6 in [42]	$s^3 M D^2 \cdot \log^4(s)\log^2(DM)$ *	$s^3 M D \cdot \log^3(s)\log(DM)$ *	Any	No	Nonuniform Exact
Thm 4.7 [42] abt [52]	$s^5 D(s + M\log s) \cdot \log^3(s)\log^3(DM)$ *	$s^5 M D \cdot \log^3(s)\log^3(DM)$	Any	No	Nonuniform Exact
Alg in [52]	$s^3 D + s^2 M D \cdot \log(sM)$ †	$s^2 M D$ †	Any	No	Empirical/Noise Robust
Mod. Alg [52]	$s^3 D$ †	$s D + M D$ †	Any	No	Empirical/Noise Robust
Alg in [12]	Avg. Case: $sD \cdot \log(s)$	Avg. Case: sD	$[M]^D$	No	Empirical/Low Noise
Alg in [13]	Avg. Case: $sD \cdot \log(s)\log(M)$	Avg. Case: $sD \cdot \log(M)$	$[M]^D$	No	Empirical/Allows Noise

Figure 1: A comparison of Corollary 1 with several existing SFT results from [40, 52, 12, 43, 13, 48, 42]. To make the runtime and sampling complexity bounds easily comparable we have set all probabilities of failure p for algorithms involving randomized constructions to be $p = \left(\frac{1}{DM}\right)^c$ for a universal constant $c \geq 1$ (except for [43] which formulates its guarantees for $p = 0.02$), and have also used that $\log(M)$ and $\log(D^2 M)$ are both $\mathcal{O}\left(\log(DM)\right)$ throughout. We also note that $K = 1$ in the Fourier basis setting considered in the table. To again aid in easier table comparisons we focussed on the recovery of functions with $\|\boldsymbol{c}_f\|_2 \leq cs$ and $\eta^{-1} \leq c$ for a universal constant $c \geq 1$ (representing machine precision) to simplify the $\log\left(\frac{\|\boldsymbol{c}_f\|_2}{\eta}\right)$ term in the runtime complexity of Corollary 1. Furthermore, we used that $\log(|\mathcal{I}|) \leq d\log(DM)$ which follows from the assumption herein that $N = |\mathcal{I}| = \binom{D}{d} M^d \leq (DM)^d$. Finally, * in the table denotes that we dropped $\log|\log(\cdot)|$ factors in the complexity bounds from [42, 43] to help save space in the table, and † indicates that [52] formulates it's bounds under the assumption that $\sqrt{M} \lesssim s \lesssim M^D$ (as opposed to assuming that $s \lesssim M^D$ is just larger than some absolute constant).

1.3 A Comparison of Corollary 1 to Prior SFT Algorithms

As mentioned above, the methods developed herein can be considered as generalizations of existing SFT techniques to BOP bases. Figure 1 compares Corollary 1 in the case of the multidimensional Fourier basis to several existing SFT results for periodic functions of D-variables [40, 52, 12, 43, 13, 48, 42]. The second and third columns of the table give the worst case runtime and sampling complexities of the methods, respectively, with the exception of the rows pertaining to [12, 13] which list average case complexities for generic signals. Note that all of the quoted runtimes are sublinear in that they scale like $o(M^D)$ for sufficiently large M and D. The fourth column indicates whether each SFT paper considers improving its performance for smaller index sets $\mathcal{I} \subset [M]^D$ with $|\mathcal{I}| \ll M^D$, or not. As indicated there, the majority of previous results only considered $\mathcal{I} = [M]^D$ with the notable exceptions of [52, 42]. The fifth columns indicates if the methods allow for the easy extension of the methods and results to Mixed Bases (MB), or not. As can be seen from the table, only Corollary 1 provides guarantees for, e.g., mixed Fourier/Chebyshev product bases at the expense of increasing K.

The final column in Figure 1 summarizes the theoretical error guarantees proven about

each method in the table. There, "Best-s Term" refers to methods that prove theoretical best-s term approximation guarantees of the type considered by Cohen et al. in [16] for all periodic functions with Fourier series coefficients that decay rapidly enough. Similarly, "Exact" therein refers to methods that provide strictly weaker guarantees regarding the recovery of functions which are exactly s-sparse in Fourier (i.e., of the form of (1.2)).[4] Note that such exactly s-sparse functions correspond to multivariate trigonometric polynomials with exactly s nonzero terms in the SFT context. Both of these types of guarantees can themselves be either "Uniform" (i.e., providing sampling sets of the type discussed below in §1.4 that work *for all* functions of the given class with high probability), or "Nonuniform" (i.e., providing sampling sets that work for *any one arbitrary* function of the given class with high probability, but not necessarily for all of them). In addition, two of the results in [40, 48] provide entirely deterministic and explicit sampling constructions with no probability of failure in achieving their respective approximation guarantees. These two error guarantees are denoted with the prefix "Det." for *Det*erministic.

Finally, several other methods [52, 12, 13] provide theoretical analysis which doesn't ultimately guarantee that they can approximate an arbitrary exactly sparse function (1.2) to machine precision with high probability. All of these methods provide extensive empirical tests to demonstrate that such failures are indeed rare, however, and so are denoted in the last column of Figure 1 by the term "Empirical" along with a description of the additive errors they are observed to tolerate on their function samples ("Noise Robust" indicates good tolerance to arbitrary perturbations including random noise, "Allows Noise" indicates tolerance to random i.i.d. mean 0 noise, and "Low Noise" indicates a tolerance to small errors on the level of numerical roundoff). We refer the readers to the original papers for additional details on their respective theoretical guarantees and empirical performance.

1.4 Randomly Constructed Grids with Universal Approximation Properties

Fix a BOP basis $\mathcal{B}$ and sparsity level s. We will call any set $\mathcal{G} \subset \mathcal{D}$ a *compressive sensing grid* if and only if $\exists$ a set of weights $\mathcal{W}$ s.t. $\forall \ f : \mathcal{D} \to \mathbb{C}$ that are s-sparse in $\mathcal{B}$

Algorithm 1 with weights $\mathcal{W}$ can recover f from its evaluations on $\mathcal{G}$

is true. As mentioned above, our main results demonstrate the existence of relatively small compressive sensing grids by randomly constructing highly structured sets of points that are then shown to be compressive sensing grids with high probability. We emphasize that our use of probability in this paper is entirely constrained to (*i*) the initial choice of

---

[4]Best s-term approximation guarantees imply the exact recovery of exactly s-sparse functions.

the grid $\mathcal{G}$ given a BOP basis $\mathcal{B}$ and sparsity level s, and to (ii) the entirely independent and one-time initial choice of a set of random Gaussian weights $\mathcal{W}$ for use in (4.1) (i.e., as part of the initialization phase for Algorithm 1). Algorithm 1 is entirely deterministic once both $\mathcal{G}$ and $\mathcal{W}$ have been chosen.

The compressing sensing grids $\mathcal{G}$ utilized herein will be the union of three distinct sets of points in $\mathcal{D}$. The first set of points is the set $\mathcal{G}^E \subset \mathcal{D}$ which has already been introduced in Section 1.1 as the set of sampling points at which f is evaluated in order to obtain $\boldsymbol{y}^{\mathbf{E}}$ in (1.3). This set of points is used in Algorithm 1 in order to estimate the basis coefficients for the basis elements identified by Algorithms 2 and 3. The second and third sets included in $\mathcal{G}$, $\mathcal{G}^I \subset \mathcal{D}$ and $\mathcal{G}^P \subset \mathcal{D}$, are utilized by Algorithm 2 and Algorithm 3, respectively. Here we will summarize the main ideas behind their construction (their precise definition is given in Section 2.3 below).

The set $\mathcal{G}^I$ is the union of D grids $\mathcal{G}_j^I$, $j \in [D]$, where each $\mathcal{G}_j^I$ is designed to allow for the identification of the j-th entry of the energetic index vectors in $\mathcal{S}$ from (1.2). Each $\mathcal{G}_j^I$ set consists of all combinations of an appropriate quadrature set for the j-th entry (to allow for an approximation of one-dimensional integrals in that variable) crossed with a random sampling set for all other entries (necessary for another approximate numerical integration in these directions that singles out the j-th entry of interest). Viewed another way, each $\mathcal{G}_j^I$ set contains the points necessary to exactly integrate the functions of the j-th variable x_j one obtains from f in (1.2) after fixing the other $D - 1$ variables to be random constants, for several different collections of random constants.

Similarly, the set $\mathcal{G}^P$ is also a union of $D - 1$ grids $\mathcal{G}_j^P$, one for each $j \in [D] \setminus \{0\}$. Here the grids are designed to allow for the sequential buildup of the energetic index vectors contained in $\mathcal{S}$ from (1.2) component by component. More precisely, the samples in $\mathcal{G}_j^P$ will be used to pair the first j components of each element of $\mathcal{S}$ with its correct $(j+1)$-th component. To generate the grid $\mathcal{G}_j^P$ we generate a set W_j of independent samples of a random vector in $\mathbb{C}^j$ and a set Z_j of independent samples of a random vector in $\mathbb{C}^{D-j}$. We then choose $\mathcal{G}_j^P := W_j \times Z_j$. The underlying idea is that this construction allows one to reduce the original D-dimensional problem to a $(j+1)$-dimensional problem with energetic index vectors given by the first $j + 1$ components of the energetic index vectors $\mathcal{S}$ from the full problem. This is achieved by using the product structure, and the random construction of Z_j which allows us to approximately integrate over the last $D - j - 1$ variables of f from (1.2). Then we can use the randomness of W_j to identify the active frequencies among all combinations of the j-dimensional component vectors identified in the previous sequential step, and the $(j+1)$-th components of $\mathcal{S}$ identified using $\mathcal{G}_{j+1}^I$ above.

As we saw in Section 1.2, it turns out that $\mathcal{G} := \mathcal{G}^E \cup \mathcal{G}^I \cup \mathcal{G}^P$ will be a compressive sensing grid with high probability even when each component set is chosen to have a

relatively small cardinality. The vast majority of the remainder of this paper will be dedicated to proving this fact. We will begin in Section 2 by introducing additional notation that is used throughout the rest of the paper, and by interpreting our function evaluations on $\mathcal{G}$,

$$\boldsymbol{y} = (\boldsymbol{y}^{\mathbf{E}}, \boldsymbol{y}^{\mathbf{I}}, \boldsymbol{y}^{\mathbf{P}})^T \in \mathbb{C}^{m'_1 + m'_2 + m'_3} \tag{1.5}$$

as standard compressive sensing measurements. Next, in Section 3, Algorithm 1 is discussed in detail and the main theorem above is proven with the help of a key technical lemma (i.e., Lemma 2) that guarantees the accuracy of our proposed support identification method. Lemma 2 is then proven in Section 4. Finally, a numerical evaluation is carried out in Section 5 that demonstrates that Algorithm 1 both behaves as expected, and is robust to noisy function evaluations. The paper then concludes after a short discussion concerning future work in Section 6.

2   Preliminaries

In this section we introduce the notation that will be used in the rest of this paper as well as the problem for which we will develop our proposed algorithm. We denote by $\mathbb{N}$ the set of natural numbers, $\mathbb{R}$ the set of real numbers, and $\mathbb{C}$ the set of complex numbers. Let $[N] := \{0, 1, 2, \ldots, N - 1\}$ for $N \in \mathbb{N}$.

2.1   Notation and Preliminaries

In this paper all letters in boldface (other than probability measures such as $\boldsymbol{\nu}$) will always represent vectors. Vectors whose entries are indexed by *index vectors* in, e.g., $[M]^D$ will be assumed to have their entries ordered lexicographically for the purposes of, e.g., matrix-vector multiplications. Thus, we say either $\boldsymbol{v} \in \mathbb{C}^{[M]^D}$ or $\boldsymbol{v} \in \mathbb{C}^{M^D}$ when we want to emphasize that each entry $v_{\boldsymbol{n}}$ of $\boldsymbol{v}$ is corresponding to its index vector $\boldsymbol{n}$, or when we perform, e.g., matrix-vector multiplications, respectively. We further define the ℓ_0 pseudo-norm of a vector $\boldsymbol{v}$ by $\|\boldsymbol{v}\|_0 := |\{i : v_i \neq 0\}|$ where the index i refers to the ith entry of the vector (in lexicographical order). If $v \in \mathbb{C}$ is a scalar then we will also use the ℓ_0-notation to correspond to the indicator function defined by

$$\|v\|_0 := \begin{cases} 1 & \text{if } v \neq 0 \\ 0 & \text{if } v = 0 \end{cases}. \tag{2.1}$$

We will consider functions $f : \mathcal{D} \to \mathbb{C}$ given in a BOS product basis expansion below

13

so that

$$f(\boldsymbol{x}) := \sum_{\boldsymbol{n} \in \mathcal{I} \subseteq [M]^D} c_{\boldsymbol{n}} T_{\boldsymbol{n}}(\boldsymbol{x}) \qquad (2.2)$$

where $\boldsymbol{c} \in \mathbb{C}^{[M]^D}$. We will further assume that f is approximately sparse in this BOS product basis. That is, we will assume that there exists some index set $\mathcal{S} \subset \mathcal{I}$ for an a priori known index set $\mathcal{I} \subseteq [M]^D$ such that $\mathcal{S}$ has the property that both (i) $|\mathcal{S}| = s \ll |\mathcal{I}| \leq M^D$, and that (ii) the set of coefficients $\mathcal{C} := \{c_{\boldsymbol{n}} \mid \boldsymbol{n} \in \mathcal{S}\} \subset \mathbb{C}$ dominates f's ℓ_2-norm in the sense that

$$\sum_{\boldsymbol{n} \in \mathcal{S} \subset \mathcal{I}} |c_{\boldsymbol{n}}|^2 \gg \sum_{\boldsymbol{n} \in [M]^D \setminus \mathcal{S}} |c_{\boldsymbol{n}}|^2 =: \epsilon^2,$$

for a relatively small number $\epsilon \geq 0$. We emphasize here that absolutely nothing about $\mathcal{S}$ is known to us in advance beyond the fact that it is a subset of $\mathcal{I}$, and has cardinality at most s. We must learn the identity of its elements ourselves by sampling f.

Our analysis herein will focus on the case where $\mathcal{I}$ is given by

$$\mathcal{I} := \left\{ \boldsymbol{n} \in [M]^D \mid \|\boldsymbol{n}\|_0 \leq d \right\}$$

for some $d \leq D$ (cf. Section 1.1). Note that this includes, for $d = D$, the special case where $\mathcal{I} = [M]^D$. We will call the index vectors $\boldsymbol{n} \in \mathcal{S}$ *energetic*. Our goal is to recover $\mathcal{S}$ and the associated coefficients $\mathcal{C}$ as rapidly as possible using only evaluations/samples from f. This will, in turn, necessitate that we sample f at very few locations in $\mathcal{D}$. In this paper we will mainly focus on providing theoretical guarantees for the case where $\epsilon = 0$ (i.e., for provably recovering f that are exactly s-sparse in a BOS product basis). Numerical experiments in Section 5 demonstrate that the method also works when $\epsilon > 0$, however. We leave theoretical guarantees in the case of $\epsilon > 0$ for future consideration.

2.2 Definitions Required for Support Identification

As with most compressive sensing and sparse approximation problems we will see that identifying the function f's support $\mathcal{S}$ is the most difficult part of recovering f. As a result our proposed iterative algorithm spends the vast majority of its time in every iteration recovering as many energetic $\boldsymbol{n} = (n_0, n_1, \cdots, n_{D-1}) \in \mathcal{S}$ as it can. Only after doing so does it then approximate a sparse vector $\boldsymbol{c} \in \mathbb{C}^{[M]^D}$ containing nonzero coefficients $c_{\boldsymbol{n}}$ for each discovered $\boldsymbol{n} \in \mathcal{S}$. Here, each $\boldsymbol{n}$ will be referred to as an *index vector* of an entry in $\boldsymbol{c}$. Let supp$(\boldsymbol{v}) \subseteq [M]^D$ represent the set of index vectors whose corresponding $v_{\boldsymbol{n}}$ entries are nonzero. We introduce the following notation in order to help explain our algorithm in the subsequent sections of the paper.

14

For a given $\boldsymbol{v} \in \mathbb{C}^{[M]^D}$, $j \in [D]$, and $\widetilde{n} \in [M]$ the vector $\boldsymbol{v}_{j;\widetilde{n}} \in \mathbb{C}^{[M]^{D-1}}$ indexed by $\boldsymbol{k} \in [M]^{D-1}$ is defined by

$$\left(v_{j;\widetilde{n}}\right)_{\boldsymbol{k}} = \begin{cases} v_{\boldsymbol{n}}, & \text{if } \boldsymbol{n} = (k_0, \ldots, k_{j-1}, \widetilde{n}, k_j, \ldots, k_{D-2}) \\ 0 & \text{otherwise} \end{cases}. \tag{2.3}$$

Note that $\boldsymbol{v}_{j;\widetilde{n}}$ will only ever have at most

$$N' := \binom{D-1}{d-\|\widetilde{n}\|_0} M^{\min\{d-\|\widetilde{n}\|_0, D-1\}} \leq \left(\frac{e(D-1)M}{\max\{d-1,1\}}\right)^d \tag{2.4}$$

nonzero entries if $v_{\boldsymbol{n}} = 0$ for all $\boldsymbol{n} \in [M]^D$ with $\|\boldsymbol{n}\|_0 > d$ by assumption. Here, as throughout the remainder of the paper, we define $\binom{p}{q}$ to be 1 whenever $q \geq p$ or $q < 0$ (also recall the definition of $\|\widetilde{n}\|_0$ from (2.1) above).[5] Similarly, for a given $\boldsymbol{v} \in \mathbb{C}^{[M]^D}$, $j \in [D]$, and $\widetilde{\boldsymbol{n}} \in [M]^{j+1}$ the vector $\boldsymbol{v}_{j;(\widetilde{\boldsymbol{n}}, \cdots)} \in \mathbb{C}^{[M]^{D-j-1}}$ indexed by $\boldsymbol{k} \in [M]^{D-j-1}$ is defined by

$$\left(v_{j;(\widetilde{\boldsymbol{n}}, \cdots)}\right)_{\boldsymbol{k}} = \begin{cases} v_{\boldsymbol{n}}, & \text{if } \boldsymbol{n} = (\widetilde{\boldsymbol{n}}, \boldsymbol{k}) \\ 0 & \text{otherwise} \end{cases}. \tag{2.5}$$

The following lemma bounds the total number of nonzero entries that $\boldsymbol{v}_{j;(\widetilde{\boldsymbol{n}}, \cdots)}$ can have given that $v_{\boldsymbol{n}} = 0$ whenever $\|\boldsymbol{n}\|_0 > d$. Note that for $j = 0$, $\boldsymbol{v}_{j;\widetilde{n}} = \boldsymbol{v}_{j;(\widetilde{n}, \cdots)}$ so that (2.4) follows as a special case.

Lemma 1. *Let* $\boldsymbol{v} \in \mathbb{C}^{[M]^D}$, $j \in [D]$, *and* $\widetilde{\boldsymbol{n}} \in [M]^{j+1}$ *with* $\|\widetilde{\boldsymbol{n}}\|_0 \leq d$. *Suppose that* $v_{\boldsymbol{n}} = 0$ *whenever* $\|\boldsymbol{n}\|_0 > d$. *Then* $v_{j;(\widetilde{\boldsymbol{n}}, \cdots)}$ *can have at most*

$$\widetilde{N}_j := \binom{D-j-1}{d-\|\widetilde{\boldsymbol{n}}\|_0} M^{\min\{d-\|\widetilde{\boldsymbol{n}}\|_0, D-j-1\}} \leq \left(\frac{e(D-j-1)M}{\max\{d-j-1,1\}}\right)^d \tag{2.6}$$

nonzero entries.

Proof. Since $v_{\boldsymbol{n}} = 0$ whenever $\|\boldsymbol{n}\|_0 > d$ it must be the case that $\left(v_{j;(\widetilde{\boldsymbol{n}}, \cdots)}\right)_{\boldsymbol{n}} = 0$ whenever $\boldsymbol{n} = (\widetilde{\boldsymbol{n}}, \widetilde{\boldsymbol{n}}')$ has $\|\widetilde{\boldsymbol{n}}'\|_0 > d - \|\widetilde{\boldsymbol{n}}\|_0$, where $\widetilde{\boldsymbol{n}}' \in \mathbb{C}^{D-j-1}$. As a result, if $D-j-1 > d - \|\widetilde{\boldsymbol{n}}\|_0$ then there are at most $\binom{D-j-1}{d-\|\widetilde{\boldsymbol{n}}\|_0}$ entry combinations left in $\boldsymbol{n}$ which can be nonzero, each of which can take on M different values. If, on the other hand, $d - \|\widetilde{\boldsymbol{n}}\|_0 \geq D - j - 1$ then all of the remaining $D-j-1$ values of $\boldsymbol{n}$ can each take on M different values. $\square$

---

[5] The min $\{d - \|\widetilde{n}\|_0, D-1\}$ in the exponent of the M in (2.4) handles the case when $d = D$ and $\widetilde{n} = 0$.

Motivated by the definition of $\boldsymbol{v}_{j;\widetilde{n}}$ in (2.3) we further define

$$\mathcal{I}_{j;\widetilde{n}} := \left\{ \boldsymbol{n} \in \mathcal{I} \mid n_j = \widetilde{n} \right\} \subset [M]^D,$$

and denote the restriction matrix that projects vectors in $\mathbb{C}^{[M]^D}$ onto each $\mathcal{I}_{j;\widetilde{n}}$ (considered as a subset of $\mathbb{C}^{[M]^{D-1}}$) by $P_{j;\widetilde{n}} \in \{0,1\}^{M^{D-1} \times M^D}$. That is, we consider each $P_{j;\widetilde{n}}$ matrix to have rows indexed by $\boldsymbol{l} \in [M]^{D-1}$, columns indexed by $\boldsymbol{k} \in [M]^D$, and entries defined by

$$(P_{j;\widetilde{n}})_{\boldsymbol{l},\boldsymbol{k}} := \begin{cases} 1 & \text{if } \boldsymbol{k} = (l_0, \ldots, l_{j-1}, \widetilde{n}, l_j, \ldots, l_{D-2}) \\ 0 & \text{otherwise} \end{cases}. \tag{2.7}$$

As a result, we have that $P_{j;\widetilde{n}}\boldsymbol{v} = \boldsymbol{v}_{j;\widetilde{n}}$ for all $\boldsymbol{v} \in \mathbb{C}^{[M]^D}$.

The fast support identification strategy we will employ in this paper will effectively boil down to rapidly approximating the norms of various $\boldsymbol{c}_{j;\widetilde{n}}$ and $\boldsymbol{c}_{j;(\widetilde{\boldsymbol{n}},\cdots)}$ vectors for carefully chosen collections of $\widetilde{n} \in [M]$ and $\widetilde{\boldsymbol{n}} \in [M]^{j+1}$. This, in turn, will be done using as few evaluations of f in (2.2) as possible in order to estimate inner products and norms of other proxy functions constructed from f. As a simple example, note that $\|\boldsymbol{c}\|_2$ can be estimated by using samples from f in order to approximate $\|f\|^2_{L^2(\mathcal{D},\boldsymbol{\nu})}$ since

$$\|f\|^2_{L^2(\mathcal{D},\boldsymbol{\nu})} = \sum_{\boldsymbol{n} \in \mathcal{I} \subseteq [M]^D} |c_{\boldsymbol{n}}|^2.$$

A bit less trivially, for $j \in [D]$ and $\widetilde{n} \in [M]$ one can also define the function $\langle f, T_{j;\widetilde{n}} \rangle_{(\mathcal{D}_j, \nu_j)} :$ $\mathcal{D}'_j \to \mathbb{C}$ with domain $\mathcal{D}'_j := \times_{k \neq j} \mathcal{D}_k$ by having $\langle f, T_{j;\widetilde{n}} \rangle_{(\mathcal{D}_j, \nu_j)} (\boldsymbol{w})$ evaluate to

$$\int_{\mathcal{D}_j} f(w_0, \ldots, w_{j-1}, z, w_{j+1}, \ldots, w_{D-2}) \overline{T_{j;\widetilde{n}}(z)} \, d\nu_j(z)$$

for all $\boldsymbol{w} \in \mathcal{D}'_j$. Let $\boldsymbol{\nu}'_j := \otimes_{k \neq j} \nu_k$. It is not too difficult to see that

$$\left\| \langle f, T_{j;\widetilde{n}} \rangle_{(\mathcal{D}_j, \nu_j)} \right\|^2_{L^2(\mathcal{D}'_j, \boldsymbol{\nu}'_j)} = \sum_{\boldsymbol{n} \in \mathcal{I} \text{ s.t. } n_j = \widetilde{n}} |c_{\boldsymbol{n}}|^2 = \|\boldsymbol{c}_{j;\widetilde{n}}\|^2_2 \tag{2.8}$$

in this case. Similarly, for some $j \in [D]$ and $\widetilde{\boldsymbol{n}} \in [M]^{j+1}$ one can define the function $\langle f, T_{j;\widetilde{\boldsymbol{n}}} \rangle_{\left(\times_{i \in [j+1]} \mathcal{D}_i, \otimes_{i \in [j+1]} \nu_i \right)}$ from $\mathcal{D}''_j := \times_{k > j} \mathcal{D}_k$ into $\mathbb{C}$ by letting $\langle f, T_{j;\widetilde{\boldsymbol{n}}} \rangle_{\left(\times_{i \in [j+1]} \mathcal{D}_i, \otimes_{i \in [j+1]} \nu_i \right)}(\boldsymbol{w})$ equal

$$\int_{\times_{i \in [j+1]} \mathcal{D}_i} f(\boldsymbol{z}, w_0, \ldots, w_{D-j-2}) \prod_{k \in [j+1]} \overline{T_{k;\widetilde{n}_k}(z_k)} \, d \left(\otimes_{i \in [j+1]} \nu_i \right)(\boldsymbol{z})$$

16

for all $\boldsymbol{w} \in \mathcal{D}_j''$. Let $\boldsymbol{\nu}_j'' := \otimes_{k>j}\nu_k$. Analogously to the situation above we then have that

$$\left\|\langle f, T_{j;\widetilde{\boldsymbol{n}}}\rangle_{L^2\left(\times_{i\in[j+1]}\mathcal{D}_i, \otimes_{i\in[j+1]}\nu_i\right)}\right\|_{L^2\left(\mathcal{D}_j'', \boldsymbol{\nu}_j''\right)}^2 = \|\boldsymbol{c}_{j;(\boldsymbol{n},\cdots)}\|_2^2. \tag{2.9}$$

As we shall see below, both (2.8) and (2.9) will be implicitly utilized in order to allow the estimation of such $\|\boldsymbol{c}_{j;\widetilde{\boldsymbol{n}}}\|_2^2$ and $\|\boldsymbol{c}_{j;(\boldsymbol{n},\cdots)}\|_2^2$ norms, respectively, using just a few nonadaptive samples from f.

2.3 The Proposed Method as a Sublinear-Time Compressive Sensing Algorithm

Note that $\boldsymbol{c} \in \mathbb{C}^{[M]^D}$ from (2.2), as well as its restrictions $\boldsymbol{c}_{j;\widetilde{\boldsymbol{n}}} \in \mathbb{C}^{[M]^{D-1}}$ and $\boldsymbol{c}_{j;(\widetilde{\boldsymbol{n}},\cdots)} \in \mathbb{C}^{[M]^{D-j-1}}$ for any $\widetilde{\boldsymbol{n}} \in [M]^{j+1}$, will all be at most s-sparse under the assumption that $\epsilon = 0$. As mentioned above, this means that recovering f from a few function evaluations is essentially equivalent to recovering $\boldsymbol{c}$ using random sampling matrices. Given this, the method we propose in the next section can also be viewed as a sublinear-time compressive sensing algorithm which uses a highly structured measurement matrix A consisting of several concatenated random sampling matrices. More explicitly, the measurements $\boldsymbol{y} \in \mathbb{C}^{m_1'+m_2'+m_3'}$ utilized by Algorithm 1 below consist of function evaluations (i.e., recall (1.5)) which can be represented in the concatenated form

$$\boldsymbol{y} = \begin{bmatrix} \boldsymbol{y^E} \\ \boldsymbol{y^I} \\ \boldsymbol{y^P} \end{bmatrix} = \begin{bmatrix} \Phi\boldsymbol{c} \\ A^I\boldsymbol{c} \\ A^P\boldsymbol{c} \end{bmatrix} = \begin{bmatrix} \Phi \\ A^I \\ A^P \end{bmatrix} \boldsymbol{c} = A\boldsymbol{c} \tag{2.10}$$

for subvectors $\boldsymbol{y^E} \in \mathbb{C}^{m_1'}$, $\boldsymbol{y^I} \in \mathbb{C}^{m_2'}$, $\boldsymbol{y^P} \in \mathbb{C}^{m_3'}$ (recall §1.4, and see below for technical details), and structured sampling matrices $\Phi \in \mathbb{C}^{m_1'\times M^D}$, $A^I \in \mathbb{C}^{m_2'\times M^D}$, and $A^P \in \mathbb{C}^{m_3'\times M^D}$.

In (2.10) the matrix Φ is a standard random sampling matrix with the RIP formed as per (1.4). It and its associated samples $\boldsymbol{y^E} = \Phi\boldsymbol{c}$ are used to estimate the entries of $\boldsymbol{c}$ indexed by the index vectors contained in the identified energetic support set T in line 13 of Algorithm 1. The matrices A^I and A^P are both used for support identification.

In particular, the matrix A^I is a random sampling matrix corresponding to the sampling set $\mathcal{G}^I$, which is constructed as follows. Let $\mathcal{U}_j := \left\{u_{j,0}, \ldots, u_{j,\mathcal{L}_j'-1}\right\} \subset \mathcal{D}_j$ be the set of $\mathcal{L}_j'$ points at which one can evaluate any given $\mathcal{B}_j$-sparse function $g : \mathcal{D}_j \to \mathbb{C}$ in the span of $\mathcal{B}_j$ in order to compute all M-inner products $\left\{\langle g, T_{j;\widetilde{n}}\rangle\right\}_{\widetilde{n}\in[M]}$ in $\mathcal{O}(\mathcal{L})$-time. Also, let $I_{\geq j} : [D-1] \to \{0,1\}$ be the indicator function that is zero when $\kappa < j$, and one when

$\kappa \geq j$. For each $j \in [D]$ we will then define $\mathcal{G}_j^I \subset \mathcal{D}$ to be the set of $m\mathcal{L}_j'$ randomly generated grid points given by

$$\boldsymbol{x}_{j,\ell,k}' = ((x_{j,\ell})_0, (x_{j,\ell})_1, \ldots, (x_{j,\ell})_{j-1}, u_{j,k}, (x_{j,\ell})_j, \ldots, (x_{j,\ell})_{D-2}) \ \forall (\ell, k) \in [m] \times [\mathcal{L}_j'],$$

where each $(x_{j,\ell})_\kappa \in \mathcal{D}_{\kappa + I_{\geq j}(\kappa)}$ is an independent realization of a random variable $\sim \nu_{\kappa + I_{\geq j}(\kappa)}$ for all $\kappa \in [D-1]$. We now take $\mathcal{G}^I$ to be the union of these sets so that

$$\mathcal{G}^I := \bigcup_{j \in [D]} \mathcal{G}_j^I = \bigcup_{j \in [D]} \left\{ \boldsymbol{x}_{j,\ell,k}' \right\}_{(\ell,k) \in [m] \times [\mathcal{L}_j']}.$$

Finally, similar to (1.3), we will also define f's evaluations on $\mathcal{G}^I$ to be $\boldsymbol{y^I} \in \mathbb{C}^{m_2'}$ where

$$\boldsymbol{y^I} = f\left(\mathcal{G}^I\right) := \left(f\left(\boldsymbol{x}_{0,0,0}'\right), f\left(\boldsymbol{x}_{0,0,1}'\right), \ldots, f\left(\boldsymbol{x}_{D-1,m-1,\mathcal{L}_{D-1}'-1}'\right) \right)^T.$$

The measurements $\boldsymbol{y^I}$ are used to try to identify all of the energetic basis functions in each input dimension, i.e., the sets

$$\mathcal{N}_j' := \left\{ \widetilde{n} \in [M] \mid \exists \boldsymbol{n} \in \mathcal{S} \text{ with } n_j = \widetilde{n} \right\} \subseteq [M] \tag{2.11}$$

for each $j \in [D]$ (see Algorithm 2).

The matrix A^P is a random sampling matrix corresponding to the sampling set $\mathcal{G}^P$, whose precise definition will again be based on several different subsets for each $j \in [D] \setminus \{0\}$. For each fixed $(j, \ell, k) \in [D] \setminus \{0\} \times [m_1] \times [m_2]$ let $\boldsymbol{w}_{j,\ell} \in W_j := \times_{i \in [j+1]} \mathcal{D}_i$ and $\boldsymbol{z}_{j,k} \in Z_j := \times_{i=j+1}^{D-1} \mathcal{D}_i$ be chosen independently at random according to $\otimes_{i \in [j+1]} \nu_i$ and $\otimes_{i=j+1}^{D-1} \nu_i$, respectively.[6] We then define $\mathcal{G}_j^P \subset \mathcal{D}$ to be the set of $m_1 m_2$ randomly generated grid points given by

$$\mathcal{G}_j^P := \left\{ (\boldsymbol{w}_{j,\ell}, \boldsymbol{z}_{j,k}) \mid (\ell, k) \in [m_1] \times [m_2] \right\} \ \forall j \in [D] \setminus \{0\}.$$

As above, we now let $\mathcal{G}^P$ be the union of these sets so that

$$\mathcal{G}^P := \bigcup_{j \in [D] \setminus \{0\}} \mathcal{G}_j^P$$

and consider f's evaluations on $\mathcal{G}^P$, denoted by $\boldsymbol{y^P} \in \mathbb{C}^{m_3'}$. The resulting measurements

$$\boldsymbol{y^P} = f\left(\mathcal{G}^P\right) := \left(f\left(\boldsymbol{w}_{1,0}, \boldsymbol{z}_{1,0}\right), f\left(\boldsymbol{w}_{1,0}, \boldsymbol{z}_{1,1}\right), \ldots, f\left(\boldsymbol{w}_{D-1,m_1-1}, \boldsymbol{z}_{D-1,m_2-1}\right) \right)^T$$

---

[6]When $j = D - 1$ the vector $\boldsymbol{z}_{j,k}$ is interpreted as a null vector satisfying $(\boldsymbol{w}_{j,\ell}, \boldsymbol{z}_{j,k}) = \boldsymbol{w}_{j,\ell} \ \forall (\ell, k)$.

and its associated samples $\boldsymbol{y}^{\mathbf{P}} = A^P \boldsymbol{c}$ are then used in Algorithm 3 to help build up the estimated support set $T \subset [M]^D$ from the previously identified $\mathcal{N}'_j$-sets. See §3 below for additional details.

The matrix A^I above is built using the matrix Kronecker products $\widetilde{A}_j \otimes L_j$ for $j \in [D]$, where $\widetilde{A}_j \in \mathbb{C}^{m \times [M]^{D-1}}$ is the random sampling matrix defined in (4.2), and $L_j \in \mathbb{C}^{\mathcal{L}'_j \times [M]}$ is a sampling matrix associated with $\mathcal{U}_j \subset \mathcal{D}_j$ from Section 1.4 defined as

$$(L_j)_{q,n} := T_{j;n}(u_{j,q}), \qquad q \in [\mathcal{L}'_j] \text{ and } n \in [M]. \tag{2.12}$$

The matrix A^P, on the other hand, is constructed using $B_j \otimes C_j$ for all $j \in [D] \setminus \{0\}$ where each $B_j \in \mathbb{C}^{m_1 \times [M]^{j+1}}$ is the random sampling matrix defined below in (4.17), and each $C_j \in \mathbb{C}^{m_2 \times [M]^{D-j-1}}$ the random sampling matrix defined in (4.16). In particular, we have that

$$A^I := \begin{bmatrix} \widetilde{A}_0 \otimes L_0 \\ \vdots \\ \widetilde{A}_{D-1} \otimes L_{D-1} \end{bmatrix}, \text{ and } A^P := \begin{bmatrix} B_1 \otimes C_1 \\ \vdots \\ B_{D-1} \otimes C_{D-1} \end{bmatrix}. \tag{2.13}$$

From a set of random Gaussian weights $\mathcal{W} = \{g_\ell^k\}_{\ell \in [m], k \in [L]}$, a matrix $G \in \mathbb{C}^{L \times m}$ is defined as

$$(G)_{k,\ell} := g_\ell^k, \qquad k \in [L] \text{ and } \ell \in [m], \tag{2.14}$$

where g_ℓ^k's are the Gaussian weights from (4.1).

Briefly contrasting the proposed approach interpreted as a sublinear-time compressive sensing method via (2.10) against previously existing sublinear-time algorithms for Compressive Sensing (CS) (see, e.g., [24, 27, 25, 41, 30]), we note that no previous sublinear-time CS methods exist which utilize measurement matrices solely derived from general BOS random sampling matrices. This means that the associated recovery algorithms developed herein can not directly take advantage of the standard group testing, hashing, and error correcting code-based techniques which have been regularly employed by such methods, making the development of fast reconstruction techniques and their subsequent analysis quite challenging. Nonetheless, we will see that we can still utilize at least some of the core ideas of these methods by sublinearizing the runtime of one of their well known superlinear-time relatives, CoSaMP [50].

3   The Proposed Method

In this section we introduce and discuss our proposed method. Roughly speaking, our algorithm can be considered as a greedy pursuit algorithm (see, e.g., [18, 22, 50, 51, 64])

Algorithm 1 Sublinearized CoSaMP

1: **procedure SublinearRecoveryAlgorithm**
2: **Input:** Sampling matrices Φ, A^I, A^P (implicitly, via the samples in $\mathcal{G}$ that determine their rows), samples $\boldsymbol{y}^{\mathbf{E}} = \Phi\boldsymbol{c}$, $\boldsymbol{y}^{\mathbf{I}} = A^I\boldsymbol{c}$, $\boldsymbol{y}^{\mathbf{P}} = A^P\boldsymbol{c}$, a sparsity estimate s, and a set of i.i.d. Gaussian weights $\mathcal{W}$
3: **Output:** s-sparse approximation $\boldsymbol{a}$ of $\boldsymbol{c}$
4:   $\boldsymbol{a}^0 = \mathbf{0}$ {Initial approximation}
5:   $\boldsymbol{v}^I \leftarrow \boldsymbol{y}^{\mathbf{I}}$, $\boldsymbol{v}^P \leftarrow \boldsymbol{y}^{\mathbf{P}}$
6:   $t \leftarrow 0$
7:   **repeat**
8:      {The next line calls Algorithm 2 ... }
9:      $\mathcal{N}_j \ \forall j \in [D] \leftarrow$ **EntryIdentification**$(\boldsymbol{v}^I, \mathcal{W})$ {Support identification step # 1}
10:      {The next line calls Algorithm 3 ... }
11:      $\Omega \leftarrow$ **Pairing**$(\boldsymbol{v}^P, \mathcal{N}_j \ \forall j \in [D])$ {Support identification step # 2}
12:      $T \leftarrow \Omega \cup \operatorname{supp}(\boldsymbol{a}^t)$ {Merge supports}
13:      $\boldsymbol{b}_T \leftarrow \Phi_T^\dagger \boldsymbol{y}^{\mathbf{E}}$ {Local estimation by least-squares}
14:      $t \leftarrow t + 1$
15:      $\boldsymbol{a}^t \leftarrow (\boldsymbol{b}_T)_s$ {Prune to obtain next approximation}
16:      $\boldsymbol{v}^I \leftarrow \boldsymbol{y}^{\mathbf{I}} - A^I\boldsymbol{a}^t$, $\boldsymbol{v}^P \leftarrow \boldsymbol{y}^{\mathbf{P}} - A^P\boldsymbol{a}^t$ {Update current samples}
17:   **until** halting criterion true
18: **end procedure**

---

with a faster support identification technique that takes advantage of the structure of BOS product bases. In particular, we will focus on the CoSaMP algorithm [50] herein. Note that support identification is the most computationally expensive step of the CoSaMP algorithm. Otherwise, CoSaMP is already a sublinear-time method for any type of BOS basis one likes. Our overall strategy, therefore, will be to hijack the CoSaMP algorithm as well as its analysis by removing its superlinear-time support identification procedure and replacing it with a new sublinear-time version that still satisfies the same iteration invariant as the original. See Algorithm 1 for pseudocode of our modified CoSaMP method. Note that most its steps are identical to the original CoSaMP algorithm except for the two "Support identification" steps, and the "Update current samples" and "halting criterion" lines. Thus, our discussion will mainly focus on these three parts.

Like CoSaMP, Algorithm 1 is a greedy approximation technique which makes locally optimal choices during each iteration. In the t-th iteration, it starts with an s-sparse approximation $\boldsymbol{a}^t$ of $\boldsymbol{c}$ and then tries to approximate the at most $2s$-sparse residual vector $\boldsymbol{r} := \boldsymbol{c} - \boldsymbol{a}^t$. The two "Support identification" steps begin approximating $\boldsymbol{r}$ by finding a support set $\Omega \subset \mathcal{I}$ of cardinality at most $2s$ which contains the set $\left\{ \boldsymbol{n} \mid |r_{\boldsymbol{n}}|^2 \geq \frac{\|\boldsymbol{r}\|_2^2}{\alpha^2 s} \right\}$

(i.e., Ω contains the indices of the entries where most of the energy of $\boldsymbol{r}$ is located). These support identification steps constitute the main modification made to CoSaMP in this paper and are discussed in more detail in Sections 3.1 and 3.2 below. After support identification, in the "Merge supports" step, a new support set T of cardinality at most $3s$ is then formed from the union of Ω with the support of the current approximation $\boldsymbol{a}^t$. At this stage T should contain the overwhelming majority of the important (i.e., energetic) index vectors in $\mathcal{S}$. As a result, restricting the columns of the sampling matrix Φ to those in T (or constructing them on the fly in a low memory setting) in order to solve for $\boldsymbol{b}_T := \text{argmin}_{\boldsymbol{u} \in \mathbb{C}^{|T|}} \|\Phi_T \boldsymbol{u} - \boldsymbol{y}^{\mathbf{E}}\|_2$ should yield accurate estimates for the true coefficients of $\boldsymbol{c}$ indexed by the elements of T, $\boldsymbol{c}_T$.[7] The vector $(\boldsymbol{b}_T)_s$ restricting $\boldsymbol{b}_T$ to its s largest-magnitude elements then becomes the next approximation of $\boldsymbol{c}$, $\boldsymbol{a}^{t+1}$.

As previously mentioned, the main difference between Algorithm 1 and CoSaMP is in the support identification steps. In the proposed method support identification consists of two parts: "Entry Identification" and "Pairing". For each of these steps we use a different measurement matrix, A^I or A^P, respectively, as well as a different set of samples (either $\boldsymbol{v}^I$ or $\boldsymbol{v}^P$) from the current residual vector. Thus, we need to update a total of three estimates every iteration: $\boldsymbol{v}^I$, $\boldsymbol{v}^P$ and $\boldsymbol{a}^t$. In the next two sections we review each of the two newly proposed support identification steps in more detail.

3.1   Support Identification Step # 1: Entry Identification

For each $j \in [D]$ the entry identification algorithm (see Algorithm 2) tries to find the j-th entry of each energetic index vector $\boldsymbol{n}$ corresponding to a nonzero entry $r_{\boldsymbol{n}}$ in the $2s$-sparse residual vector $\boldsymbol{r} = \boldsymbol{c} - \boldsymbol{a}^t$. Note that for each j this gives rise to at most $2s$ index entries in $[M]$.[8] We therefore define $\mathcal{N}_j^t$ to be the resulting set $\{n_j \mid \boldsymbol{n} \in \text{supp}(\boldsymbol{r})\}$ of size at most $2s$ for each $j \in [D]$, and note that $\mathcal{N}_j^0 = \mathcal{N}_j'$ in (2.11). Note further that $\widetilde{n} \in \mathcal{N}_j^t$ if and only if $\|\boldsymbol{r}_{j;\widetilde{n}}\|_2 > 0$. As a result we can learn $\mathcal{N}_j^t$ by approximating $\|\boldsymbol{r}_{j;\widetilde{n}}\|_2$

---

[7]In practice, it suffices to approximate the least-squares solution b_T by an iterative least-squares approach such as Richardson's iteration or conjugate gradient [17] since computing the exact least squares solution can be expensive when s is large. The argument of [50] shows that it is enough to take three iterations for Richardson's iteration or conjugate gradient if the initial condition is set to $\boldsymbol{a}^t$, and if Φ has an RIP constant $\delta_{2s} < 0.025$. In fact, both of these methods have similar runtime performance.

[8]Note that we are generally assuming herein that $2s < M$. In the event that $2s \geq M$ one can proceed in at least two different ways. The first way is to not change anything, and to simply be at peace with the possibility of, e.g., occasionally returning $\mathcal{N}_j = [M]$. This is our default approach. The second way is to regroup the first $g \in \mathbb{N}$ variables of f together into a new collective "first variable", the second g variables together into a new collective "second variable", etc., for some g satisfying $M^g > 2s$. After such regrouping the algorithm can then again effectively be run as is with respect to these new collective variables.

using $\left\| \langle h, T_{j;\widetilde{n}} \rangle_{(\mathcal{D}_j, \nu_j)} \right\|^2_{L^2(\mathcal{D}'_j, \boldsymbol{\nu}'_j)}$ via (2.8) as long as we know

$$h(\boldsymbol{x}) := \sum_{\boldsymbol{n} \in \text{supp}(\boldsymbol{r})} r_{\boldsymbol{n}} T_{\boldsymbol{n}}(\boldsymbol{x}). \tag{3.1}$$

Whenever $\left\| \langle h, T_{j;\widetilde{n}} \rangle_{(\mathcal{D}_j, \nu_j)} \right\|^2_{L^2(\mathcal{D}'_j, \boldsymbol{\nu}'_j)}$ is larger than a threshold value τ (e.g., zero) for a particular choice of $j \in [D]$ and $\widetilde{n} \in [M]$, we could simply add $\widetilde{n}$ to $\mathcal{N}_j$ (our estimate of $\mathcal{N}_j^t$) in this case.

Algorithm 2 Entry identification

{The method works because $\text{median}_{k \in [L]} \left| \langle h_{j;k}, T_{j;\widetilde{n}} \rangle_{(\mathcal{D}_j, \nu_j)} \right| \approx \| \boldsymbol{r}_{j;\widetilde{n}} \|_2$. See (3.1) and (4.1) for the definition of $h_{j;k}$. For exactly s-sparse $\boldsymbol{c}$ one can use $\tau = 0$ below. More generally, one can select the largest s estimates for $\mathcal{N}_j$.}

1: **procedure** EntryIdentification
2:   **Input:** $\boldsymbol{v}^I$, and a set of i.i.d. Gaussian weights $\mathcal{W}$ for use in the $h_{j;k}$ below (see (4.1))
3:   **Output:** $\mathcal{N}_j$ for $j \in [D]$
4:     **for** $j = 0 \rightarrow D - 1$ **do**
5:       $\mathcal{N}_j \leftarrow \emptyset$
6:       **for** $\widetilde{n} = 0 \rightarrow M - 1$ **do**
7:         **if** $\text{median}_{k \in [L]} \left| \langle h_{j;k}, T_{j;\widetilde{n}} \rangle_{(\mathcal{D}_j, \nu_j)} \right| > \tau$, **then**
8:           $\mathcal{N}_j \leftarrow \{\widetilde{n}\} \cup \mathcal{N}_j$.
9:         **end if**
10:       **end for**
11:     **end for**
12: **end procedure**

Of course we don't actually know exactly what h is. However, we do have access to samples from h in each iteration in the form of $\boldsymbol{v}^I$ and $\boldsymbol{v}^P$. And, as a result, we are able to approximate $\left\| \langle h, T_{j;\widetilde{n}} \rangle_{(\mathcal{D}_j, \nu_j)} \right\|^2_{L^2(\mathcal{D}'_j, \boldsymbol{\nu}'_j)} = \| \boldsymbol{r}_{j;\widetilde{n}} \|_2$ for any $j \in [D]$ and $\widetilde{n} \in [M]$ with the estimator

$$\text{median}_k \left| \langle h_{j;k}, T_{j;\widetilde{n}} \rangle_{(\mathcal{D}_j, \nu_j)} \right|$$

defined using (4.1). In Section 4.1 we show that this estimator can be used to accurately approximate $\| \boldsymbol{r}_{j;\widetilde{n}} \|_2$ *for all $2s$-sparse residual vectors $\boldsymbol{r} \in \mathbb{C}^{[M]^D}$* using only $\mathcal{O}\left(s \cdot K^2 d\mathcal{L}' \cdot \text{polylog}(D, s, M, K) \right)$ universal samples from any given $\boldsymbol{r}$'s associated h-function in (3.1) (i.e., the samples in $\boldsymbol{v}^I$).[9] Furthermore, the estimator can always be computed in just $\mathcal{O}\left(s^2 \cdot K^2 d^2 \mathcal{L} \cdot \text{polylog}(D, s, M, K) \right)$-time.

---

[9]Recall that $\mathcal{L}'$ represents the maximum number of function evaluations one needs in order to com-

At this point it is important to note that managing to find each $\mathcal{N}_j^t$ exactly for all $j \in [D]$ still does not provide enough information to allow us to learn $\mathrm{supp}(\boldsymbol{r})$ efficiently when $d > 1$. In general the most we learn from this information is that $\mathrm{supp}(\boldsymbol{r}) \subset \left(\times_{j \in [D]} \mathcal{N}_j \right) \bigcap \mathcal{I} \subset [M]^D$. In the next "Pairing" step we address this problem by iteratively pruning the candidates in $\times_{j \in [1]} \mathcal{N}_j, \times_{j \in [2]} \mathcal{N}_j, \ldots, \times_{j \in [D]} \mathcal{N}_j$ down at each stage to the best $2s$ candidates for being a prefix of some element in $\mathrm{supp}(\boldsymbol{r})$. As we shall see, the pruning in each "Pairing" stage involves energy estimates that are computed using only the samples from h in $\boldsymbol{v}^P$. These ideas are discussed in greater detail in the next section.

3.2 Support Identification Step # 2: Pairing

Algorithm 3 Pairing

1: **procedure Pairing**
2: **Input:** $\boldsymbol{v}^P = \left\{ h(\boldsymbol{w}_{j,\ell}, \boldsymbol{z}_{j,k}) \mid j \in [D] \setminus \{0\}, \ell \in [m_1], k \in [m_2] \right\}$, $\mathcal{N}_j$ for $j \in [D]$
3: **Output:** $\mathcal{P}$
4: $\quad \mathcal{P}_0 \leftarrow \mathcal{N}_0$
5: $\quad$ **for** $j = 1 \to D - 1$ **do**
6: $\quad\quad$ {This method works because $E_{j;(\widetilde{\boldsymbol{n}}\ldots)} \approx \|\boldsymbol{r}_{j;(\widetilde{\boldsymbol{n}},\cdots)}\|_2^2$ below.}
7: $\quad\quad E_{j;(\widetilde{\boldsymbol{n}}\ldots)} \leftarrow \frac{1}{m_2} \sum_{k \in [m_2]} \left| \frac{1}{m_1} \sum_{\ell \in [m_1]} h(\boldsymbol{w}_{j,\ell}, \boldsymbol{z}_{j,k}) \overline{T_{\widetilde{\boldsymbol{n}}}(\boldsymbol{w}_{j,\ell})} \right|^2 \ \forall \widetilde{\boldsymbol{n}} \in \mathcal{P}_{j-1} \times \mathcal{N}_j.$
8: $\quad\quad$ Create $\mathcal{P}_j$ containing each $\widetilde{\boldsymbol{n}}$ whose energy estimate $E_{j;(\widetilde{\boldsymbol{n}}\ldots)}$ is in the $2s$-largest.
9: $\quad$ **end for**
10: $\quad \mathcal{P} \leftarrow \mathcal{P}_{D-1}$
11: **end procedure**

Once all the $\mathcal{N}_j \subset [M]$ have been identified for all $j \in [D]$ it remains to match them together in order learn as many of the true length-D index vectors in $\mathrm{supp}(\boldsymbol{r}) \subset [M]^D$ as possible. To achieve this we begin by attempting to identify all the prefixes of length two, $\widetilde{\boldsymbol{n}} = (\widetilde{n}_0, \widetilde{n}_1) \in \mathcal{N}_0 \times \mathcal{N}_1$, which begin at least one element in the support of $\boldsymbol{r}$. Similar to the ideas utilized above, we now note that $(\widetilde{\boldsymbol{n}}, \boldsymbol{n}') \in \mathrm{supp}(\boldsymbol{r})$ for some $\boldsymbol{n}' \in [M]^{D-2}$ if and only if $\|\boldsymbol{r}_{j;(\widetilde{\boldsymbol{n}},\cdots)}\|_2^2 > 0$. As a result, it suffices for us to use the samples from h (recall (3.1)) in $\boldsymbol{v}^P$ in order to compute $E_{j;(\widetilde{\boldsymbol{n}}\ldots)} \approx \|\boldsymbol{r}_{j;(\widetilde{\boldsymbol{n}},\cdots)}\|_2^2$ in Algorithm 3 above. The $2s$-largest estimates $E_{j;(\widetilde{\boldsymbol{n}}\ldots)}$ are then used to identify all the prefixes of length 2 which begin at least one element of $\mathrm{supp}(\boldsymbol{r})$. Of course, this same idea can then be used again to find all length-3 prefixes of elements in $\mathrm{supp}(\boldsymbol{r})$ by extending the previously identified length-2 prefixes in all possible $\mathcal{O}(s^2)$ combinations with the elements in $\mathcal{N}_2$,

---
pute $\langle g, T_{j;\widetilde{n}} \rangle$ for all $\widetilde{n} \in [M]$ in $\mathcal{O}(\mathcal{L})$-time for any given $j \in [D]$, and s-sparse $g : \mathcal{D}_j \to \mathbb{C}$ in span $\left\{ T_{j;m} \mid m \in [M] \right\}$.

$$
\begin{array}{c}
\boxed{\begin{array}{l}(0,\cdot,\cdot)\\(1,\cdot,\cdot)\\ \vdots \\(M-1,\cdot,\cdot)\end{array}} \quad \xrightarrow{\text{Alg 2.}} \quad \boxed{\begin{array}{l}(3,\cdot,\cdot)\\(4,\cdot,\cdot)\\(11,\cdot,\cdot)\end{array}}
\end{array}
$$

$$[M]\times[M]\times[M] \quad \supset \quad \mathcal{N}_0\times[M]\times[M]$$
$$|\mathcal{N}_0|=3$$

(a)

$$
\boxed{\begin{array}{l}(\cdot,0,\cdot)\\(\cdot,1,\cdot)\\ \vdots \\(\cdot,M-1,\cdot)\end{array}} \quad \xrightarrow{\text{Alg 2.}} \quad \boxed{\begin{array}{l}(\cdot,5,\cdot)\\(\cdot,7,\cdot)\end{array}}
$$

$$[M]\times[M]\times[M] \quad \supset \quad [M]\times\mathcal{N}_1\times[M]$$
$$|\mathcal{N}_1|=2$$

(b)

$$
\boxed{\begin{array}{l}(\cdot,\cdot,0)\\(\cdot,\cdot,1)\\ \vdots \\(\cdot,\cdot,M-1)\end{array}} \quad \xrightarrow{\text{Alg 2.}} \quad \boxed{\begin{array}{l}(\cdot,\cdot,6)\\(\cdot,\cdot,8)\\(\cdot,\cdot,100)\end{array}}
$$

$$[M]\times[M]\times[M] \quad \supset \quad [M]\times[M]\times\mathcal{N}_2$$
$$|\mathcal{N}_2|=3$$

(c)

$$
\boxed{\begin{array}{l}(3,5,\cdot)\\(3,7,\cdot)\\(4,5,\cdot)\\(4,7,\cdot)\\(11,5,\cdot)\\(11,7,\cdot)\end{array}} \quad \xrightarrow{\text{Alg. 3}} \quad \boxed{\begin{array}{l}(3,5,\cdot)\\(4,7,\cdot)\\(11,5,\cdot)\end{array}}
$$

$$\mathcal{N}_0\times\mathcal{N}_1\times[M] \quad \supset \quad \mathcal{P}_1\times[M]$$
$$|\mathcal{N}_0\times\mathcal{N}_1|=6 \qquad |\mathcal{P}_1|=3$$

(d)

$$
\boxed{\begin{array}{l}(3,5,6)\\(3,5,8)\\(3,5,100)\\(4,7,6)\\(4,7,8)\\(4,7,100)\\(11,5,6)\\(11,5,8)\\(11,5,100)\end{array}} \quad \xrightarrow{\text{Alg. 3}} \quad \boxed{\begin{array}{l}(3,5,6)\\(4,7,8)\\(11,5,100)\end{array}}
$$

$$\mathcal{P}_1\times\mathcal{N}_2 \quad \supset \quad \mathrm{supp}(\boldsymbol{r})$$
$$|\mathcal{P}_1\times\mathcal{N}_2|=9 \qquad |\mathrm{supp}(\boldsymbol{r})|=3$$

(e)
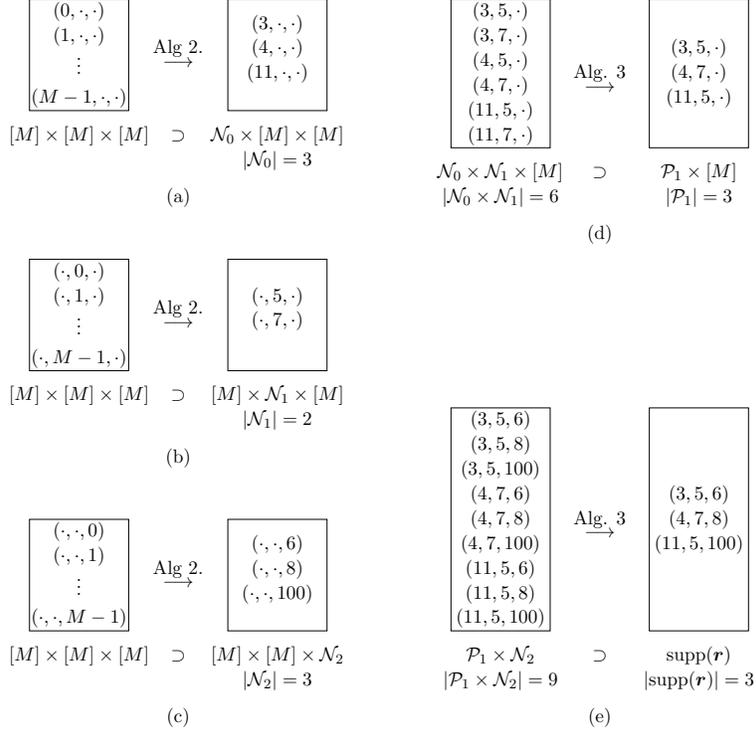
Figure 2: Description of Entry Identification ((a), (b), (c)) and Pairing ((d), (e))

and then testing the resulting length-3 prefixes' energies in order to identify the $2s$ most energetic such combinations, etc.. See Algorithm 3 above for pseudocode, §4.2 for analysis of these $E_{j;(\widetilde{\boldsymbol{n}}\ldots)}$ estimators, and §3.2.1 just below for a concrete example of the pairing process.

3.2.1   An Example of Entry Identification and Pairing to Find Support

Assume that $\boldsymbol{r} \in \mathbb{C}^{[M]^3}$ is three-sparse with a priori unknown energetic index vectors

$$\mathrm{supp}(\boldsymbol{r}) = \{(3,5,6),\ (4,7,8),\ (11,5,100)\} \subset [M]^3$$

and corresponding nonzero coefficients $r_{(3,5,6)}$, $r_{(4,7,8)}$, and $r_{(11,5,100)}$. We can further imagine that M here is significantly larger than, e.g., 100 so that computing all M^3 coefficients of $\boldsymbol{r}$ using standard numerical methods would be undesirable. In this case, Algorithm 2 aims to output the sets

$$\mathcal{N}_0 = \{3,4,11\},\ \mathcal{N}_1 = \{5,7\},\ \text{and}\ \mathcal{N}_2 = \{100,6,8\} \subset [M],$$

24

i.e., the first, second, and third entries of each index vector in the support of $\boldsymbol{r}$, respectively. These sets are described in (a), (b) and (c) of Figure 2. Note that there are $18 = 3 \times 2 \times 3$ possible index vectors which are consistent with the $\mathcal{N}_0$, $\mathcal{N}_1$, and $\mathcal{N}_2$ above. Algorithm 3 is now tasked with finding out which of these 18 possibilities are truly elements of $\text{supp}(\boldsymbol{r})$ without having to test them all individually.[10]

To identify $\text{supp}(\boldsymbol{r})$ without having to test all 18 index vectors in $\mathcal{N}_0 \times \mathcal{N}_1 \times \mathcal{N}_2$, the pairing process instead starts by estimating the energy of the $|\mathcal{N}_0| \cdot |\mathcal{N}_1| = 6$ length-2 prefixes in $\mathcal{N}_0 \times \mathcal{N}_1$ which might begin an index vector in $\text{supp}(\boldsymbol{r})$. In the ideal case these energy estimates will reveal that only 3 of these 6 possible length-2 prefixes actually have any energy,

$$\mathcal{P}_1 = \{(3,5),(4,7),(11,5)\} \subset [M]^2$$

which is illustrated in (d) of Figure 2. In its next stage visualized in (e) of Figure 2, the pairing process now continues by combining these three length-2 prefixes in $\mathcal{P}_1$ with $\mathcal{N}_2$ in order to produce $|\mathcal{P}_1| \times |\mathcal{N}_2| = 9$ final candidate elements potentially belonging to $\text{supp}(\boldsymbol{r}) \subset [M]^3$. Estimating the energy of these 9 candidates then finally reveals the true identities of the index vectors in the support of $\boldsymbol{r}$.

Note that instead of computing energy estimates for all 18 possible support candidates in $\mathcal{N}_0 \times \mathcal{N}_1 \times \mathcal{N}_2$, the pairing process allows us to determine the correct support of $\boldsymbol{r}$ using only $15 = 6 + 9 < 18$ total energy estimates in this example. Though somewhat underwhelming in this particular example, the improvement provided by Algorithm 3 becomes much more significant as the dimension D of the index vectors grows larger. When $|\text{supp}(\boldsymbol{r})| = |\mathcal{N}_j| = s$ for all $j \in [D]$, for example, $\times_{j \in [D]} \mathcal{N}_j$ will have s^D total elements. Nonetheless, Algorithm 3 will be able to identify $\text{supp}(\boldsymbol{r})$ using only $\mathcal{O}\left(s^2 D\right)$ energy estimates in the ideal setting.[11]

3.2.2 Comparison to Entry Identification and Pairing Methods in Prior Work

As previously mentioned, the support identification approach outlined above is similar in nature to the dimension incremental approaches utilized by both [52, 42] and [12, 13] in the Fourier setting.[12] For example, Algorithm 1 of [52] also works by performing D rounds of (i) entry identification to find sets called $I^{(t)}$ in [52] which are essentially

---

[10]In this simple example we can of course simply estimate the energy for all 18 possible index vectors. The three true index vectors in the support of $\boldsymbol{r}$ with nonzero energy would then be discovered and all would be well. However, this naive approach becomes spectacularly inefficient for larger $D \gg 3$.

[11]In less optimal settings one should keep in mind that Algorithm 3 only finds the most energetic entries in general, so that $\mathcal{P} \supset \left\{ \boldsymbol{n} \mid |r_{\boldsymbol{n}}|^2 \geq \frac{\|\boldsymbol{r}\|_2^2}{\alpha^2 s} \right\}$ for a given $\alpha > 1$. This is why we need to apply it iteratively.

[12]In fact, this is unsurprising given that similar dimension incremental strategies have been proposed as far back as the 1970's in work related to recovering sparse algebraic polynomials [65].

identical the $\mathcal{N}_{t-1}$ sets found in our Algorithms 1 and 2 above, followed by (ii) a pairing step to build up sets called $I^{(1,\dots,t)}$ in [52] which are essentially identical to the sets $\mathcal{P}_{t-1}$ in our Algorithm 3. As a result, the papers [52, 42] identify the support in Fourier of the Fourier sparse functions they seek to approximate via entry identification and pairing just as we do herein, at least superficially. However, there are several important differences between our algorithm's support identification strategy and the one used by [52, 42], the most crucial of which stems from the choice of the estimators used in [52, 42] versus those used herein in order to determine the $\mathcal{N}_{t-1}/I^{(t)}$ and $\mathcal{P}_{t-1}/I^{(1,\dots,t)}$ sets.

As can be seen in line 7 of Algorithm 2, we utilize a median estimator to determine our $\mathcal{N}_{t-1}$ sets herein, whereas Algorithm 1 of [52] effectively uses the magnitude of the Fourier coefficients of functions along the lines of $h_{j;k}$ from Algorithms 2 (except with $h \leftarrow f$) as an estimator to identify their $I^{(t)}$ sets. Similarly, the energy estimator we use in line 7 of Algorithm 3 to determine each $\mathcal{P}_{t-1}$ is instead replaced in Algorithm 1 of [52] by the magnitude of the Fourier coefficients of functions along the lines of $h(\boldsymbol{w}_{j,\ell}, \boldsymbol{z}_{j,k})$ from Algorithm 3 above, except, again, with h equal to the original function f. It turns out that these differences in the choices of the estimators used to determine the entry and pairing sets is pivotal for many other features of each method. In [52, 42] the use of Fourier coefficients of modified functions for estimators allows the authors to employ (multiple) rank-1 lattice techniques (see e.g. [45, 32]) in order to efficiently and accurately compute their estimators in the Fourier basis setting. Herein, our choice of different and more general estimators is essential to us being able to achieve uniform recovery guarantees for a significantly more general class of BOPB's.

Similarly, in [12, 13] the support of Fourier sparse functions is also recovered using a modified dimension incremental strategy. In particular, therein D-dimensional frequencies with nonzero Fourier coefficients are again extended element-wise in order to recover sets analogous to $\mathcal{P}_{t-1}$ above using a multidimensional phase encoding approach (see, e.g., section 3.1.1 of [29] for an example of phase encoding in the one-dimensional setting). Though fast, the resulting support identification approach in [12, 13] is potentially lossy in the sense that some elements of the $\mathcal{P}_{t-1}$ sets that the methods above might recover can be lost by the algorithms in [12, 13]. As a result, the algorithms therein may not identify all of the Fourier support of some Fourier sparse functions. To compensate for those worst case scenarios [12, 13] suggest a "tilting method" where the frequencies are projected onto alternate coordinates rotated with a certain angle which can help to recover such difficult functions, and also employs and iterative reconstruction approach in order to try to compensate for Fourier coefficients which might cancel one another out in some of the utilized projections.

A second crucial difference between the algorithms in [52, 42, 12, 13] and the approach proposed herein is the fact that we apply our support identification methods to the residual

functions h above (3.1) so that the overall approximation error reduces at a fixed rate each iteration following the iterative strategy of the CoSaMP algorithm instead of, e.g., directly applying them to the original function f itself, or iterating in some other fashion. This allows us to use the compressive sensing error analysis techniques developed in [50] to our benefit, whereas those techniques are not applicable to the algorithms as developed in [52, 42, 12, 13]. We are now prepared to begin proving the promised recovery results for our method.

3.3 A Theoretical Guarantee for Support Identification

The following lemma and theorem show that our support identification procedure (i.e., Algorithm 2, followed by Algorithm 3) always identifies the indexes of the majority of the energetic entries in $\boldsymbol{r}$. Consequently, the energy of the residual is guaranteed to decrease from iteration to iteration of Algorithm 1. We want to remind readers that $\boldsymbol{r}$ is always $2s$-sparse since $\boldsymbol{c}$ and each $\boldsymbol{a}^{t-1}$ are s-sparse in the present analysis (i.e., $\epsilon = 0$).

Lemma 2. *Suppose that $\left\{ T_{\boldsymbol{n}} \mid \boldsymbol{n} \in \mathcal{I} \subseteq [M]^D \right\}$ is a BOS where each basis function $T_{\boldsymbol{n}}$ is defined as per (1.1) , and $T_{j;0} \equiv 1$ for all $j \in [D]$. Let $\mathcal{H}_{2s}$ be the set of all functions, $h : \mathcal{D} \to \mathbb{C}$, in span $\left\{ T_{\boldsymbol{n}} \mid \boldsymbol{n} \in \mathcal{I} \right\}$ whose coefficient vectors are $2s$-sparse, and let $\boldsymbol{r}_h \in \mathbb{C}^{\mathcal{I}}$ denote the $2s$-sparse coefficient vector for each $h \in \mathcal{H}_{2s}$. Fix $p \in (0, 1/2)$, $1 \leq d \leq D$, $N = \binom{D}{d} M^d$, and $K = \sup\limits_{\boldsymbol{n} \ s.t.\|\boldsymbol{n}\|_0 \leq d} \|T_{\boldsymbol{n}}\|_\infty$. Then, one can randomly select a set of i.i.d. Gaussian weights $\mathcal{W} \subset \mathbb{R}$ for use in (4.1), and also randomly construct entry identification and pairing grids, $\mathcal{G}^I \subset \mathcal{D}$ and $\mathcal{G}^P \subset \mathcal{D}$ (recall §2.3), whose total cardinality $\left| \mathcal{G}^I \right| + \left| \mathcal{G}^P \right|$ is*
$$\mathcal{O}\left(sD\mathcal{L}'K^2 \max\{\log^2(s)\log^2(DN), \log(\tfrac{D}{p})\} + s^3 DK^4 \max\left\{\log^4(s)\log^2(N)\log^2(DN), \log^2(\tfrac{D}{p})\right\}\right),$$
such that the following property holds $\forall h \in \mathcal{H}_{2s}$ with probability greater than $1 - 2p$:

> *Let $\boldsymbol{v}_h^I \in \mathbb{C}^{m'_2}$ and $\boldsymbol{v}_h^P \in \mathbb{C}^{m'_3}$ be samples from $h \in \mathcal{H}_{2s}$ on $\mathcal{G}^I$ and $\mathcal{G}^P$, respectively. If Algorithms 2 and 3 are granted access to $\boldsymbol{v}_h^I$, $\boldsymbol{v}_h^P$, $\mathcal{G}^I$, $\mathcal{G}^P$, and $\mathcal{W}$ then they will find a set $\Omega \subset [M]^D$ of cardinality $2s$ in line 11 of Algorithm 1 such that*
> $$\|(\boldsymbol{r}_h)_{\Omega^c}\|_2 \leq 0.202\|\boldsymbol{r}_h\|_2.$$

Furthermore, the total runtime complexity of Algorithms 2 and 3 is always
$$\mathcal{O}\left(s^2 D\mathcal{L}K^2 \max\{\log^2(s)\log^2(DN), \log(\tfrac{D}{p})\} + s^5 D^2 K^4 \max\left\{\log^4(s)\log^2(N)\log^2(DN), \log^2(\tfrac{D}{p})\right\}\right).$$

In Lemma 2 above $\mathcal{L}'$ denotes the maximum number of points in $\mathcal{D}_j$ required in order to determine the value of $\langle g, T_{j;\widetilde{n}}\rangle_{(\mathcal{D}_j, \nu_j)}$ for all $\widetilde{n} \in [M]$ in $\mathcal{O}(\mathcal{L})$-time for any given s-sparse

$g : \mathcal{D}_j \to \mathbb{C}$ in span $\{T_{j;m} \mid m \in [M]\}$, maximized over all $j \in [D]$. See §1.2 for a more in depth discussion of these quantities. The reader is also referred back to §1.4 and §2.3 for a discussion of the entry identification and pairing grids, $\mathcal{G}^I$ and $\mathcal{G}^P$, mentioned in Lemma 2. The proof of Lemma 2, which is quite long and technical, is given in Section 4.3. Once Lemma 2 has been established, however, it is fairly straightforward to prove that Algorithm 1 will always rapidly recover any function of D-variables which exhibits sparsity in a tensor product basis by building on the results in [50]. We have the following theorem.

Theorem 2. *Suppose that $\{T_{\boldsymbol{n}} \mid \boldsymbol{n} \in \mathcal{I} \subseteq [M]^D\}$ is a BOS where each basis function $T_{\boldsymbol{n}}$ is defined as per (1.1), and $T_{j;0} \equiv 1$ for all $j \in [D]$. Let $\mathcal{F}_s$ be the subset of all functions $f \in \text{span} \{T_{\boldsymbol{n}} \mid \boldsymbol{n} \in \mathcal{I}\}$ whose coefficient vectors are s-sparse, and let $\boldsymbol{c}_f \in \mathbb{C}^{\mathcal{I}}$ denote the s-sparse coefficient vector for each $f \in \mathcal{F}_s$. Fix $p \in (0, 1/3)$, $1 \leq d \leq D$, $N = \binom{D}{d} M^d$, $K = \sup\limits_{\boldsymbol{n} \ s.t. \|\boldsymbol{n}\|_0 \leq d} \|T_{\boldsymbol{n}}\|_\infty$, and a precision parameter $\eta > 0$. Then, one can randomly select a set of i.i.d. Gaussian weights $\mathcal{W} \subset \mathbb{R}$ for use in (4.1), and also randomly construct a compressive sensing grid, $\mathcal{G} \subset \mathcal{D}$, whose total cardinality $|\mathcal{G}|$ is*
$$\mathcal{O}\left(sD\mathcal{L}'K^2 \max\{\log^2(s)\log^2(DN), \log(\tfrac{D}{p})\} + s^3 DK^4 \max\left\{\log^4(s)\log^2(N)\log^2(DN), \log^2(\tfrac{D}{p})\right\}\right),$$
such that the following property holds $\forall f \in \mathcal{F}_s$ with probability greater than $1 - 3p$:

> *Let $\boldsymbol{y}$ consist of samples from $f \in \mathcal{F}_s$ on $\mathcal{G}$. If Algorithm 1 is granted access to $\boldsymbol{y}$, $\mathcal{G}$, and $\mathcal{W}$, then it will produce an s-sparse approximation $\boldsymbol{a}$ such that*
> $$\|\boldsymbol{c}_f - \boldsymbol{a}\|_2 \leq C\eta.$$

> *Here $C > 0$ is an absolute constant.*

Furthermore, the total runtime complexity of Algorithm 1 is always
$$\mathcal{O}\left(\left(s^2 D^2 \mathcal{L} K^2 \max\{\log^2(s)\log^2(DN), \log(\tfrac{D}{p})\} + s^5 D^2 K^4 \max\left\{\log^4(s)\log^2(N)\log^2(DN), \log^2(\tfrac{D}{p})\right\}\right)\right.$$
$$\left. \times \log \frac{\|\boldsymbol{c}_f\|_2}{\eta}\right) \text{ when } |\mathcal{G}| \text{ is bounded as above.}$$

As above, we refer the reader back to §1.2 for a discussion of the quantities $\mathcal{L}'$ and $\mathcal{L}$ appearing in Theorem 2, as well as to §1.4 for more information on the compressive sensing grid $\mathcal{G} \subset \mathcal{D}$ mentioned therein.

Proof. In order to analyze the support identification step, we replace Lemma 4.2 in [50] by Lemma 2, and then we obtain
$$\|\boldsymbol{c}_f - \boldsymbol{a}^{t+1}\|_2 \leq 0.5\|\boldsymbol{c}_f - \boldsymbol{a}^t\|_2$$

for each iteration $t \geq 0$, which is the same as in Theorem 2.1 in [50] provided that f is exactly s-sparse and samples are not noisy. Except for the support identification step(s), Algorithm 1 agrees with CoSaMP, so that Lemmas 4.3 – 4.5 in [50] still directly apply

to Algorithm 1. After $\mathcal{O}\left(\log \frac{\|\boldsymbol{c}_f\|_2}{\eta}\right)$ iterations, we see that the s-sparse approximation $\boldsymbol{a}$ therefore satisfies

$$\|\boldsymbol{c}_f - \boldsymbol{a}\|_2 \leq C\eta.$$

Since the runtime complexity of the support identification steps and the sample update process in each iteration, the total running time arises from multiplying it with the number of iterations. The number of sample points, m_1', m, m_1 and m_2 used to define the matrices Φ, $\widetilde{A}_j$, B_j and C_j discussed in §2.3 are all chosen to ensure that these resulting measurement matrices have the RIP. Thus, the samples of f in $\boldsymbol{y}$ can be reused over as many iterations as needed. Updating the samples of each residual function for the entry identification or pairing causes an extra $\mathcal{O}(sDm_2')$ and $\mathcal{O}(sDm_3')$ computations respectively which gives rise to a D^2 factor instead of D in the first term of runtime complexity. The probability of successful recovery for all $f \in \mathcal{F}_s$ is obtained by taking the union bound over the failure probability p of Φ having $\delta_{2s} < 0.025$ via Theorem 1 together with the failure probability $2p$ of Lemma 2. $\qquad\square$

We are now prepared to begin the process of proving Lemma 2.

4   Analysis of the Support Identification

In this section, we analyze the performance of the sublinear-time support identification technique proposed in Algorithms 2 and 3. First, we show in Section 4.1 the success of the entry identification step. Indeed, Theorems 3 and 4 show under the RIP assumption that certain one-dimensional proxy functions allow us to identify the entry with large corresponding coefficients. Lemma 6 then estimates the necessary sample complexity. In Section 4.2, we analyze the pairing step showing that it works uniformly for any $2s$-sparse functions in Theorem 5. Finally, in Section 4.3, we complete the proof of the Lemma 2 providing the complete result for the proposed support identification method by combining/using the results of Sections 4.1 and 4.2.

4.1   Entry Identification

In this section we aim to find $\mathcal{N}_j^t$ containing the j-th entries of the index vectors of the nonzero transform coefficients of h for all $j \in [D]$, which is done by Algorithm 2. Define $[D]' := [D] \setminus \{j\}$.

Assume without loss of generality that the number $L \in \mathbb{N}$ of proxy functions is odd. Choose $\mathcal{X}_j := \{\boldsymbol{x}_{j,\ell}\}_{\ell \in [m]}$ where each $\boldsymbol{x}_{j,\ell}$ is chosen independently at random from $\mathcal{D}_j'$

according to $d\boldsymbol{\nu}'_j$. Also, choose $\{g_1^k, \cdots, g_m^k\}_{k \in [L]}$ where each g_ℓ^k is an i.i.d. standard Gaussian variable $\sim \mathcal{N}(0,1)$, which forms $\mathcal{W}$ introduced in Section 1.4. We define a function $h_{j;k} : \mathcal{D}_j \to \mathbb{C}$ of one variable in $\mathcal{D}_j$ as follows,

$$h_{j;k}(x) := \frac{1}{\sqrt{m}} \sum_{\ell \in [m]} g_\ell^k h([x, \boldsymbol{x}_{j,\ell}]) = \frac{1}{\sqrt{m}} \sum_{\ell \in [m]} g_\ell^k \sum_{\boldsymbol{n} \in \mathrm{supp}(\boldsymbol{r})} r_{\boldsymbol{n}} T_{j;n_j}(x) \prod_{i \in [D]'} T_{i;n_i}(x_{j,\ell})_i,$$
(4.1)

and $[x, \boldsymbol{x}_{j,\ell}]$ is the vector obtained by inserting the variable x between the entries of $\boldsymbol{x}_{j,\ell}$ indexed by $[j]$ and $\{j, j+1 \cdots, D-2\}$, i.e.,

$$[x, \boldsymbol{x}_{j,\ell}] := ((x_{j,\ell})_0, (x_{j,\ell})_1, \cdots, (x_{j,\ell})_{j-1}, \ x \ , (x_{j,\ell})_j, \cdots, (x_{j,\ell})_{D-2}).$$

For the sake of simplicity, we let $T_{\check{\boldsymbol{n}}}(\check{\boldsymbol{x}}) := \prod_{i \in [D]'} T_{i;\check{n}_i}(\check{x}_i)$ with $\check{\boldsymbol{n}} \in [M]^{D-1}$ and $\check{\boldsymbol{x}} \in \mathcal{D}'_j$.

We choose m large enough above to form the normalized random sampling matrix $\widetilde{A}_j \in \mathbb{C}^{m \times [M]^{D-1}}$ for each $j \in [D]$, defined as

$$\left(\widetilde{A}_j \right)_{\ell, \check{\boldsymbol{n}}} := \frac{1}{\sqrt{m}} T_{\check{\boldsymbol{n}}}(\boldsymbol{x}_{j,\ell}), \ \ell \in [m], \ \check{\boldsymbol{n}} \in [M]^{D-1},$$
(4.2)

so that each one has a restricted isometry constant δ_{2s} of at most δ with high probability in its restricted form. Here, the restricted form is introduced by eliminating the columns of the full $\widetilde{A}_j$ indexed by vectors $\boldsymbol{n} \notin \mathcal{I}_{j;\check{n}}$. To explain further, we denote $\widetilde{A}_j P_{j;\check{n}} \boldsymbol{r} = \widetilde{A}_j \boldsymbol{r}_{j;\check{n}}$ where $P_{j;\check{n}}$ is a restriction matrix defined in (2.7). This comes from the inner product in line 10 of Algorithm 2 which is calculated by using the evaluations of $h_{j,k}$ at $\mathcal{U}_j$ defined in Section 1.4. Then, the inner product can be written as $\left(G\widetilde{A}_j P_{j;\check{n}} \boldsymbol{r} \right)_k$ where G is defined as in (2.14). In other words, the evaluations of h at $[u_{j,k}, \boldsymbol{x}_{j,\ell}]$ from $\mathcal{G}_j^I$ in Section 1.4 are utilized to compute the inner product. Then, the matrix-vector multiplication $\widetilde{A}_j \boldsymbol{r}_{j;\check{n}}$ can be considered in its restricted form by eliminating the columns of $\widetilde{A}_j$ and elements of $\boldsymbol{r}_{j;\check{n}}$ which are zero due to their corresponding index vectors not belonging to $\mathcal{I}_{j;\check{n}}$. The resulting restricted matrix $\widetilde{A}_j$ has the size $m \times N'$ where N' is bounded above in (2.4) so that $\widetilde{A}_j$ has the restricted isometry constant δ_{2s} mentioned. The advantage of forming RIP matrices in this fashion is that it allows us to analyze the different iterations of Algorithm 1 repeatedly with the same RIP matrices. For a discussion about the number of measurements needed to ensure that $\widetilde{A}_j$ satisfies the RIP, see the following lemma (which is a simple consequence of Theorem 1).

Lemma 3. *Let $\widetilde{A}_j \in \mathbb{C}^{m \times N'}$ be the random sampling matrix as in (4.2) in its restricted form. If, for $\delta, p \in (0,1)$,*

$$m \geq aK^2 \delta^{-2} s \max \left\{ d \log^2(4s) \log(9m) \log \left(\frac{8e(D-1)DM}{d} \right), \log \left(\frac{D}{p} \right) \right\},$$

then with probability at least $1-p$, *the restricted isometry constant* δ_s *of* $\widetilde{A}_j$ *satisfies* $\delta_s \leq \delta$ *for all* $j \in [D]$. *The constant* $a > 0$ *is universal.*

As we will show, more than half of the proxy functions, $\{h_{j;k}\}_{k \in [L]}$ are guaranteed with high probability to have $\|h_{j;k}\|_{L_2(\mathcal{D}_j, \nu_j)}$ bounded above by $\|r\|_2$ up to some constant, and also $\left|\langle h_{j;k}, T_{j;\widetilde{n}} \rangle_{(\mathcal{D}_j, \nu_j)}\right|$ bounded above and below by $\|r_{j;\widetilde{n}}\|_2$ up to some constants for all $\widetilde{n} \in [M]$ and $j \in [D]$.

To show this we consider the indicator variable $E_{h,j,\widetilde{n},k}$ which is 1 if and only if all three of

1. $\|h_{j;k}\|^2_{L^2(\mathcal{D}_j, \nu_j)} \leq \alpha' \|r\|_2^2$ for the absolute constant α' defined in Lemma 4,

2. $\frac{9}{4}\|r_{j;\widetilde{n}}\|_2 \geq \left|\langle h_{j;k}, T_{j;\widetilde{n}} \rangle_{(\mathcal{D}_j, \nu_j)}\right| \geq \frac{\sqrt{23}}{12}\|r_{j;\widetilde{n}}\|_2$, and

3. the vector of Gaussian weights $g^k \in \mathbb{R}^m$ satisfying $\frac{1}{2}m \leq \|g^k\|_2^2 \leq \frac{3}{2}m$,

are simultaneously true, and 0 otherwise.

The proof will proceed as follows. Lemmas 4 and 5 together with the bound on $\|g^k\|_2$ through Bernstein's inequality imply that the probability of each $E_{h,j,\widetilde{n},k}$ being 1 is greater than 0.5. Combining this with Chernoff bound, the deviation of $\sum_{k \in [L]} E_{h,j,\widetilde{n},k}$ below its expectation shows exponential decay in its distribution. As a result, with sufficiently many proxy functions, i.e., sufficiently large L, the probability that $\sum_{k \in [L]} E_{h,j,\widetilde{n},k} < L/2$ for all $(h, j, \widetilde{n}) \in \mathcal{H} \times [D] \times [M]$ becomes very small for any finite set of functions $\mathcal{H} \subset \mathcal{H}_{2s}$ whose BOS coefficient vectors are $2s$-sparse (see Theorem 3). The number L logarithmically depends on $DM|\mathcal{H}|$. In order to get the desired properties for all $2s$-sparse functions satisfying our support assumption, the finite function set $\mathcal{H}$ is taken as $\mathcal{H}^\epsilon$ with corresponding coefficient vector set $\mathcal{R}^\epsilon$ which is an ϵ-cover over all ℓ_2-normalized $2s$-sparse vectors in $\mathbb{C}^N$ in Theorem 4. Thus, with high probability, the desired properties hold uniformly, i.e., for all functions of interest, and for all $j \in [D]$ and $\widetilde{n} \in [M]$.

The following lemma bounds the energy of the proxy functions.

Lemma 4. *Suppose that* $r \in \mathbb{C}^N$ *is* $2s$-*sparse, and the restricted isometry constant* δ_{2s} *of* $\widetilde{A}_j$ *satisfies* $\delta_{2s} \leq \delta$ *for all* $j \in [D]$ *where* $\delta \in (0, 7/16)$. *Then, for each* $k \in [L]$, *there exists an absolute constant* $\alpha' \in \mathbb{R}^+$ *such that*

$$\mathbb{P}\left[\|h_{j;k}\|^2_{L^2(\mathcal{D}_j, \nu_j)} \geq \alpha'\|r\|_2^2\right] \leq .025 \tag{4.3}$$

for all $j \in [D]$.

Proof. Consider the random sampling point set $\mathcal{X}_j = \{x_{j,\ell} \mid \ell \in [m]\}$ to be fixed for the moment. We begin by noting that

$$\|h_{j;k}\|^2_{L^2(\mathcal{D}_j,\nu_j)} = \int_{\mathcal{D}_j} |h_{j;k}(x)|^2 d\nu_j(x)$$

$$= \int_{\mathcal{D}_j} \left| \frac{1}{\sqrt{m}} \sum_{\ell \in [m]} g_\ell^k h([x, \boldsymbol{x}_{j,\ell}]) \right|^2 d\nu_j(x)$$

$$= \frac{1}{m} \sum_{\ell,\ell' \in [m]} g_\ell^k g_{\ell'}^k \int_{\mathcal{D}_j} h([x, \boldsymbol{x}_{j,\ell}]) \overline{h([x, \boldsymbol{x}_{j,\ell'}])} \, d\nu_j(x)$$

$$= \frac{1}{m} \sum_{\ell,\ell' \in [m]} g_\ell^k g_{\ell'}^k \sum_{\boldsymbol{n},\boldsymbol{n}' \in \mathrm{supp}(\boldsymbol{r})} r_{\boldsymbol{n}} \overline{r_{\boldsymbol{n}'}} \prod_{i \in [D]'} T_{i;n_i}(x_{j,\ell})_i \prod_{i' \in [D]'} \overline{T_{i';n'_{i'}}(x_{j,\ell'})_{i'}}$$

(4.4)

$$\times \int_{\mathcal{D}_j} T_{j;n_j}(x) \overline{T_{j;n'_j}(x)} \, d\nu_j(x)$$

$$= \frac{1}{m} \sum_{\ell,\ell' \in [m]} g_\ell^k g_{\ell'}^k \left(\sum_{\widetilde{n} \in [M]} \sum_{\substack{\boldsymbol{n},\boldsymbol{n}' \text{ s.t.} \\ n_j = n'_j = \widetilde{n}}} r_{\boldsymbol{n}} \overline{r_{\boldsymbol{n}'}} \prod_{i \in [D]'} T_{i;n_i}(x_{j,\ell})_i \prod_{i' \in [D]'} \overline{T_{i';n'_{i'}}(x_{j,\ell'})_{i'}} \right)$$

$$= \sum_{\widetilde{n} \in [M]} \sum_{\ell,\ell' \in [m]} g_\ell^k g_{\ell'}^k \left(\widetilde{A}_j \boldsymbol{r}_{j;\widetilde{n}} \right)_\ell \overline{\left(\widetilde{A}_j \boldsymbol{r}_{j;\widetilde{n}} \right)_{\ell'}},$$

(4.5)

where $\widetilde{A}_j \in \mathbb{C}^{m \times N'}$ is the restricted random sampling matrix from (4.2) and $\boldsymbol{r}_{j;\widetilde{n}} \in \mathbb{C}^{N'}$ is the restricted vector from (2.3). Thus, we can see that

$$\|h_{j;k}\|^2_{L^2(\mathcal{D}_j,\nu_j)} \;=\; \sum_{\widetilde{n} \in [M]} |\langle \widetilde{A}_j \boldsymbol{r}_{j;\widetilde{n}}, \boldsymbol{g}^k \rangle|^2 \;=\; \left\| (\boldsymbol{g}^k)^* \widetilde{A}_j R \right\|^2_2 \;=\; \left\| \left(\widetilde{A}_j R \right)^* \boldsymbol{g}^k \right\|^2_2$$

where $R \in \mathbb{C}^{N' \times M}$ is the matrix whose $\widetilde{n}^{\text{th}}$ column is $\boldsymbol{r}_{j;\widetilde{n}}$. This yields results that $\mathbb{E}_{\boldsymbol{g}^k} \left[\|h_{j;k}\|^2_{L^2(\mathcal{D}_j,\nu_j)} \right] = \|\widetilde{A}_j R\|^2_{\mathrm{F}}$.

Now observe that $\left(\widetilde{A}_j R \right)^* \boldsymbol{g}^k \sim \mathcal{N}(\boldsymbol{0}, U\Sigma^2 U^*)$, where $U\Sigma V^*$ is the SVD of $\left(\widetilde{A}_j R \right)^*$ and $\Sigma \in \mathbb{R}^{\min\{M,m\} \times \min\{M,m\}}$ is the diagonal matrix containing at most $\min\{M,m\}$ nonzero singular values, $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{\min\{M,m\}} \geq 0$, of $\widetilde{A}_j R$. Let $\widetilde{\boldsymbol{g}}^k := \Sigma V^* \boldsymbol{g}^k \sim \mathcal{N}(\boldsymbol{0}, \Sigma^2)$ and note that $\|U\widetilde{\boldsymbol{g}}^k\|^2_2 = \|\widetilde{\boldsymbol{g}}^k\|^2_2$. As a consequence, one can see that

$$\mathbb{P}\left[\|h_{j;k}\|^2_{L^2(\mathcal{D}_j,\nu_j)} \geq t \right] = \mathbb{P}\left[\|\widetilde{\boldsymbol{g}}^k\|^2_2 \geq t \right] = \mathbb{P}\left[\sum_{\ell=1}^{\min\{M,m\}} \sigma_\ell^2 X_\ell \geq t \right]$$

holds for all $t \in \mathbb{R}$, where each X_ℓ is an i.i.d χ^2 random variable. Applying the Bernstein type inequality given in Proposition 5.16 in [61] we deduce that

$$\mathbb{P}\left[\left|\|h_{j;k}\|_{L^2(\mathcal{D}_j,\nu_j)}^2 - \|\widetilde{A}_j R\|_{\mathrm{F}}^2\right| \geq t\right] \leq 2\exp\left(-a'\min\left\{\frac{t^2}{\|\boldsymbol{\sigma}\|_4^4}, \frac{t}{\|\boldsymbol{\sigma}\|_\infty^2}\right\}\right)$$

$$\leq 2\exp\left(-a'\min\left\{\frac{t^2}{\|\widetilde{A}_j R\|_F^4}, \frac{t}{\|\widetilde{A}_j R\|_F^2}\right\}\right) \qquad (4.6)$$

where $a' \in \mathbb{R}^+$ is an absolute constant, and $\boldsymbol{\sigma}$ is the vector containing the diagonal elements of Σ. An application of (4.6) with $t = \max\left\{\log 80/a', \sqrt{\log 80/a'}\right\}\|\widetilde{A}_j R\|_{\mathrm{F}}^2$ finally tells us that

$$\|h_{j;k}\|_{L^2(\mathcal{D}_j,\nu_j)}^2 \geq \left(1 + \max\left\{\frac{\log 80}{a'}, \sqrt{\frac{\log 80}{a'}}\right\}\right)\|\widetilde{A}_j R\|_{\mathrm{F}}^2$$

will hold with probability at most $1/40$.

Turning our attention to $\|\widetilde{A}_j R\|_{\mathrm{F}}^2 = \sum_{\widetilde{n}\in[M]} \|\widetilde{A}_j \boldsymbol{r}_{j;\widetilde{n}}\|_2^2$, we assert that

$$\|\widetilde{A}_j R\|_{\mathrm{F}}^2 = \sum_{\widetilde{n}\in[M]} \|\widetilde{A}_j \boldsymbol{r}_{j;\widetilde{n}}\|_2^2 \geq \frac{1}{2}\sum_{\widetilde{n}\in[M]} \|\boldsymbol{r}_{j;\widetilde{n}}\|_2^2 = \frac{1}{2}\|\boldsymbol{r}\|_2^2$$

holds since the restricted isometry constant δ_{2s} of $\widetilde{A}_j$ is assumed to be bounded above by $\frac{7}{16}$. Therefore, we finally get the desired probability estimate with $\alpha' := \frac{1}{2}\left(1 + \max\left\{\frac{\log 80}{a'}, \sqrt{\frac{\log 80}{a'}}\right\}\right)$. $\qquad \square$

The following lemma bounds the estimated inner products.

Lemma 5. *Suppose that $\boldsymbol{r} \in \mathbb{C}^N$ is 2s-sparse, and the restricted isometry constant δ_{2s} of $\widetilde{A}_j$ satisfies $\delta_{2s} \leq \delta$ for all $j \in [D]$ where $\delta \in (0, 7/16)$. Let $k \in [L]$, $j \in [D]$, and $\widetilde{n} \in [M]$. Then, there exists an absolute constant $\beta' \in \mathbb{R}^+$ such that*

$$\mathbb{P}\left[\left|\langle h_{j;k}, T_{j;\widetilde{n}}\rangle_{(\mathcal{D}_j,\nu_j)}\right| \leq \frac{\sqrt{23}}{12}\|\boldsymbol{r}_{j;\widetilde{n}}\|_2 \ \text{or} \ \left|\langle h_{j;k}, T_{j;\widetilde{n}}\rangle_{(\mathcal{D}_j,\nu_j)}\right| \geq \frac{9}{4}\|\boldsymbol{r}_{j;\widetilde{n}}\|_2\right] \leq 0.273. \quad (4.7)$$

Proof. Consider the random sampling point set $\mathcal{X}_j$ to be fixed for the time being. Recalling the definitions of $h_{j;k}$ and of $\boldsymbol{r}_{j;\widetilde{n}}$, one can see that

$$\langle h_{j;k}, T_{j;\widetilde{n}}\rangle_{(\mathcal{D}_j,\nu_j)} = \frac{1}{\sqrt{m}}\sum_{\ell\in[m]} g_\ell^k \sum_{\substack{\boldsymbol{n} \text{ s.t.} \\ n_j=\widetilde{n}}} r_{\boldsymbol{n}} \prod_{i\in[D]'} T_{i;n_i}(x_{j,\ell})_i = \sum_{\ell\in[m]} g_\ell^k \left(\widetilde{A}_j \boldsymbol{r}_{j;\widetilde{n}}\right)_\ell. \qquad (4.8)$$

33

Looking at (4.8) one can see that $\langle h_{j;k}, T_{j;\widetilde{n}} \rangle_{(\mathcal{D}_j, \nu_j)} \sim \mathcal{N}(0, \|\widetilde{A}_j \boldsymbol{r}_{j;\widetilde{n}}\|_2^2)$ and hence,

$$\mathbb{P}\left[\left|\langle h_{j;k}, T_{j;\widetilde{n}} \rangle_{(\mathcal{D}_j, \nu_j)}\right| \leq \frac{\|\widetilde{A}_j \boldsymbol{r}_{j;\widetilde{n}}\|_2}{3} \text{ or } \left|\langle h_{j;k}, T_{j;\widetilde{n}} \rangle_{(\mathcal{D}_j, \nu_j)}\right| \geq 3\|\widetilde{A}_j \boldsymbol{r}_{j;\widetilde{n}}\|_2\right] \leq 0.273 \quad (4.9)$$

holds. Combining (4.9) and the assumption on δ_{2s}, which yields $\frac{9}{16}\|\boldsymbol{r}_{j;\widetilde{n}}\|_2^2 \leq \|\widetilde{A}_j \boldsymbol{r}_{j;\widetilde{n}}\|_2^2 \leq \frac{23}{16}\|\boldsymbol{r}_{j;\widetilde{n}}\|_2^2$, establishes the desired result. $\qquad\square$

Theorem 3. *Let $\mathcal{H}$ be a finite set of functions h whose BOS coefficient vectors are $2s$-sparse, and let $\boldsymbol{r}_h \in \mathbb{C}^N$ denote the coefficient vector for each $h \in \mathcal{H}$. Suppose that the restricted isometry constant δ_{2s} of $\widetilde{A}_j$ satisfies $\delta_{2s} \leq \delta$ for all $j \in [D]$ where $\delta \in (0, 7/16)$. Furthermore, let $p \in (0,1)$, $L \in \mathbb{N}$ be odd, and $L \geq \widetilde{\gamma}\log(DM|\mathcal{H}|/p)$ hold for a sufficiently large absolute constant $\widetilde{\gamma} \in \mathbb{R}^+$. Then, $\sum_{k\in[L]} E_{h,j,\widetilde{n},k} > L/2$ simultaneously for all $(h, j, \widetilde{n}) \in \mathcal{H} \times [D] \times [M]$ with probability at least $1 - p$. That is, with probability at least $1 - p$, the following will hold simultaneously for each $(h, j, \widetilde{n}) \in \mathcal{H} \times [D] \times [M]$: All three of*

1. *$\|h_{j;k}\|_{L^2(\mathcal{D}_j, \nu_j)}^2 \leq \alpha'\|\boldsymbol{r}_h\|_2^2$ for the absolute constant α' defined in Lemma 4,*

2. *$\frac{9}{4}\|(\boldsymbol{r}_h)_{j;\widetilde{n}}\|_2 \geq \left|\langle h_{j;k}, T_{j;\widetilde{n}} \rangle_{(\mathcal{D}_j, \nu_j)}\right| \geq \frac{\sqrt{23}}{12}\|(\boldsymbol{r}_h)_{j;\widetilde{n}}\|_2$, and*

3. *the vector of Gaussian weights $\boldsymbol{g}^k \in \mathbb{R}^m$ satisfying $\frac{1}{2}m \leq \|\boldsymbol{g}^k\|_2^2 \leq \frac{3}{2}m$,*

will be simultaneously true for more than half of the $k \in [L]$.

Proof. Let $h \in \mathcal{H}$, $k \in [L]$, and $\widetilde{n} \in [M]$. The probabilities that the first and second properties fail are given in (4.3) and (4.7), respectively. For the third property, applying the Bernstein type inequality given in Proposition 5.16 in [61], one obtains

$$\mathbb{P}\left[\left|\|\boldsymbol{g}^k\|_2^2 - m\right| \geq \frac{m}{2}\right] \leq 2\mathrm{e}^{-a''m} \leq 0.03, \quad (4.10)$$

where $a'' \in \mathbb{R}^+$ is an absolute constant.

Combining (4.3), (4.7), (4.10) via a union bound now tell us that $\mathbb{P}\left[E_{h,j,\widetilde{n},k} = 0\right] \leq 328/1000$. Utilizing the Chernoff bound (see, e.g., [49, 3]) one now sees that

$$\mathbb{P}\left[\sum_{k\in[L]} E_{h,j,\widetilde{n},k} < L/2\right] = \mathbb{P}\left[\sum_{k\in[L]} (1 - E_{h,j,\widetilde{n},k}) > L/2\right] < \mathrm{e}^{-L/\bar{\gamma}} \leq \frac{p}{DM|\mathcal{H}|}$$

for an absolute constant $\bar{\gamma} \in \mathbb{R}^+$, where the last inequality follows by choosing $\widetilde{\gamma} = \bar{\gamma}$ in the assumption. Applying the union bound over all choices of $(h, j, \widetilde{n}) \in \mathcal{H} \times [D] \times [M]$ now establishes the desired result. $\qquad\square$

Theorem 4. *Let $\mathcal{H}_{2s}$ be the set of all functions h whose coefficient vectors are $2s$-sparse, and let $\boldsymbol{r}_h \in \mathbb{C}^N$ denote the coefficient vector for each $h \in \mathcal{H}_{2s}$. Suppose that the restricted isometry constant δ_{2s} of $\widetilde{A}_j$ satisfies $\delta_{2s} \leq \delta$ for all $j \in [D]$ where $\delta \in (0, 7/16)$. Furthermore, let $p \in (0,1)$, $L \in \mathbb{N}$ be odd, and assume that $L \geq \gamma' s \cdot d \cdot \log\left(\frac{DM}{\sqrt[d]{sd\ p^{1/s}}}\right)$ for sufficiently large absolute constant $\gamma' \in \mathbb{R}^+$. Then, with probability greater than $1-p$, one has $\sum_{k \in [L]} E_{h,j,\widetilde{n},k} > L/2$ simultaneously for all $(h, j, \widetilde{n}) \in \mathcal{H} \times [D] \times [M]$. Consequently, with probability greater than $1-p$, it will hold that for all choices of $(h, j, \widetilde{n}) \in \mathcal{H}_{2s} \times [D] \times [M]$ both*

1. *$\|h_{j;k}\|^2_{L^2(\mathcal{D}_j, \nu_j)} \leq (\alpha'+1)\|\boldsymbol{r}_h\|_2^2$ for the absolute constant α' defined in Lemma 4, and*

2. *$\frac{9}{2}\|(\boldsymbol{r}_h)_{j;\widetilde{n}}\|_2 \geq \left|\langle h_{j;k}, T_{j;\widetilde{n}}\rangle_{(\mathcal{D}_j, \nu_j)}\right| \geq \frac{1}{3}\|(\boldsymbol{r}_h)_{j;\widetilde{n}}\|_2$*

are true simultaneously for more than half of the $k \in [L]$.

Proof. Define $\mathcal{R}^\epsilon \subset \mathbb{C}^N$ as a finite ϵ-cover of all $2s$-sparse coefficient vectors $\boldsymbol{r} \in \mathbb{C}^N$ with $\|\boldsymbol{r}\|_2 = 1$, together with $\boldsymbol{0}$, where $N = \binom{D}{d}M^d$ and $\epsilon \in (0,1)$. Such covers exist of cardinality $|\mathcal{R}^\epsilon| \leq \left(\frac{\mathrm{e}N}{2s}\right)^{2s}\left(1 + \frac{2}{\epsilon}\right)^{2s}$ (see, e.g., Appendix C of [23]). Define $\mathcal{H}^\epsilon$ as the set of functions corresponding to the $2s$-sparse coefficient vectors in $\mathcal{R}^\epsilon$. Assume that for $\mathcal{H} = \mathcal{H}^\epsilon$ and all choices of $(h, j, \widetilde{n}) \in \mathcal{H}^\epsilon \times [D] \times [M]$, Properties $1-3$ of Theorem 3 will hold for more than half of the $k \in [L]$. By the theorem, this event will happen with probability at least $1-p$. We will now prove that under this assumption both Properties 1 and 2 above will hold as desired.

Let $\widetilde{n} \in [M]$, $j \in [D]$, consider $h \in \mathcal{H}_{2s}$ with coefficient vector $\boldsymbol{r} = \boldsymbol{r}_h$, and let $\tau := \|\boldsymbol{r}\|_2^2$. Then, there exists an $h' \in \mathcal{H}^\epsilon$ with coefficient vector $\boldsymbol{r}' \in \mathcal{R}^\epsilon$ such that both $\|\boldsymbol{r}'\|_2 = 1$ and $\|\boldsymbol{r} - \tau\boldsymbol{r}'\|_2 \leq \epsilon\tau$ hold. Finally, let $k \in [L]$ be one of the values for which Properties $1-3$ of Theorem 3 are simultaneously true for $(h', j, \widetilde{n})$. We will begin by establishing Property 1 above for h, j and k. Using (4.1) we have that

$$
\begin{aligned}
\|h_{j;k}\|_{L^2(\mathcal{D}_j, \nu_j)} &= \left\|\frac{1}{\sqrt{m}}\sum_{\ell \in [m]} g_\ell^k \sum_{\boldsymbol{n} \in \mathrm{supp}(\boldsymbol{r})} r_{\boldsymbol{n}} T_{j;n_j}(x) \prod_{i \in [D]'} T_{i;n_i}(x_{j,\ell})_i\right\|_{L^2(\mathcal{D}_j, \nu_j)} \\
&\leq \tau \left\|h'_{j;k}\right\|_{L^2(\mathcal{D}_j, \nu_j)} \quad\quad\quad\quad\quad (4.11) \\
&\quad + \left\|\frac{1}{\sqrt{m}}\sum_{\ell \in [m]} g_\ell^k \sum_{\boldsymbol{n} \in \mathrm{supp}(\boldsymbol{r})} \left(r_{\boldsymbol{n}} - \tau r'_{\boldsymbol{n}}\right) T_{j;n_j}(x) \prod_{i \in [D]'} T_{i;n_i}(x_{j,\ell})_i\right\|_{L^2(\mathcal{D}_j, \nu_j)} \\
&\leq \tau\sqrt{\alpha'}\|\boldsymbol{r}'\|_2 + \left\|\left(h - \tau h'\right)_{j;k}\right\|_{L^2(\mathcal{D}_j, \nu_j)}
\end{aligned}
$$

35

$$= \tau \sqrt{\alpha'} + \left\| \left(h - \tau h' \right)_{j;k} \right\|_{L^2(\mathcal{D}_j, \nu_j)}, \tag{4.12}$$

where the last inequality follows from the first property of Theorem 3 holding for h'.

Repeating the expansion from the proof of Lemma 4 for $\left\| \left(h - \tau h' \right)_{j;k} \right\|^2_{L^2(\mathcal{D}_j, \nu_j)}$, one obtains

$$
\begin{aligned}
\left\| \left(h - \tau h' \right)_{j;k} \right\|^2_{L^2(\mathcal{D}_j, \nu_j)} &= \sum_{\widetilde{n} \in [M]} \left| \left\langle \widetilde{A}_j \left(\boldsymbol{r} - \tau \boldsymbol{r}' \right)_{j;\widetilde{n}}, \boldsymbol{g}^k \right\rangle \right|^2 \\
&\leq \sum_{\widetilde{n} \in [M]} \left\| \widetilde{A}_j \left(\boldsymbol{r} - \tau \boldsymbol{r}' \right)_{j;\widetilde{n}} \right\|^2_2 \left\| \boldsymbol{g}^k \right\|^2_2 \\
&\leq \frac{9}{4} m \sum_{\widetilde{n} \in [M]} \left\| \left(\boldsymbol{r} - \tau \boldsymbol{r}' \right)_{j;\widetilde{n}} \right\|^2_2
\end{aligned}
$$

where the last inequality follows from the third property of Theorem 3, and $\widetilde{A}_j$ having $\delta_{2s} \leq \frac{7}{16}$. Continuing, we can see that

$$\left\| \left(h - \tau h' \right)_{j;k} \right\|^2_{L^2(\mathcal{D}_j, \nu_j)} \leq \frac{9}{4} m \left\| \left(\boldsymbol{r} - \tau \boldsymbol{r}' \right) \right\|^2_2 \leq \frac{9}{4} m \tau^2 \epsilon^2.$$

Combining this expression with (4.12) we now learn that

$$\left\| h_{j;k} \right\|_{L^2(\mathcal{D}_j, \nu_j)} \leq \tau \left(\sqrt{\alpha'} + \frac{3}{2} \epsilon \sqrt{m} \right) = \left\| \boldsymbol{r} \right\|_2 \left(\sqrt{\alpha'} + \frac{3}{2} \epsilon \sqrt{m} \right).$$

Making sure to use, e.g., an $\epsilon \leq \left(6\sqrt{\alpha' m} \right)^{-1}$ ensures property one.

Turning our attention to establishing Property 2 above for h, $\widetilde{n}$, and k, we now choose an $h' \in \mathcal{H}^\epsilon$ whose coefficient vector $\boldsymbol{r}' \in \mathcal{R}^\epsilon$ has $\left\| \boldsymbol{r}' \right\|_2 = 1$, and also satisfies

$$\left\| \boldsymbol{r}_{j;\widetilde{n}} - \tau' \boldsymbol{r}' \right\|_2 \leq \epsilon \tau' \tag{4.13}$$

for $\tau' := \left\| \boldsymbol{r}_{j;\widetilde{n}} \right\|_2$. Note that all nonzero entries of $\boldsymbol{r}'_{j;\widetilde{n}}$ agree with those of $\boldsymbol{r}'$, the latter vector only has certain additional nonzero entries in locations where $\boldsymbol{r}'_{j;\widetilde{n}}$ and also $\boldsymbol{r}_{j;\widetilde{n}}$ vanish. Consequently, replacing $\boldsymbol{r}'$ by $\boldsymbol{r}'_{j;\widetilde{n}}$ makes the left hand side of (4.13) smaller, and one obtains that

$$\left\| \boldsymbol{r}_{j;\widetilde{n}} - \tau' \boldsymbol{r}'_{j;\widetilde{n}} \cdot \right\|_2 \leq \epsilon \tau' \tag{4.14}$$

From (4.1) one can see that

$$\left| \langle h_{j;k}, T_{j;\widetilde{n}} \rangle_{(\mathcal{D}_j, \nu_j)} \right| \geq \left| \langle \tau' h'_k, T_{j;\widetilde{n}} \rangle_{(\mathcal{D}_j, \nu_j)} \right| - \left| \langle \left(h_{j;k} - \tau' h'_k \right), T_{j;\widetilde{n}} \rangle_{(\mathcal{D}_j, \nu_j)} \right|$$

36

$$\geq \frac{\sqrt{23}}{12}\tau'\|\boldsymbol{r}'_{j;\widetilde{n}}\|_2 - \left|\sum_{\ell\in[m]} g_\ell^k \left(\widetilde{A}_j \left(\boldsymbol{r} - \tau'\boldsymbol{r}'\right)_{j;\widetilde{n}}\right)_\ell\right|.$$

where the last inequality follows from the second property of Theorem 3 holding for h'. Continuing using (4.14) we have that

$$\begin{aligned}
\left|\langle h_{j;k}, T_{j;\widetilde{n}}\rangle_{(\mathcal{D}_j,\nu_j)}\right| &\geq \frac{\sqrt{23}}{12} \left(\|\boldsymbol{r}_{j;\widetilde{n}}\|_2 - \|(\tau'\boldsymbol{r}' - \boldsymbol{r})_{j;\widetilde{n}}\|_2\right) - \left|\left\langle \widetilde{A}_j \left(\boldsymbol{r} - \tau'\boldsymbol{r}'\right)_{j;\widetilde{n}}, \boldsymbol{g}^k\right\rangle\right| \\
&\geq \frac{\sqrt{23}}{12}\|\boldsymbol{r}_{j;\widetilde{n}}\|_2 - \frac{\sqrt{23}}{12}\epsilon\tau' - \left|\left\langle \widetilde{A}_j \left(\boldsymbol{r} - \tau'\boldsymbol{r}'\right)_{j;\widetilde{n}}, \boldsymbol{g}^k\right\rangle\right| \\
&\geq \frac{\sqrt{23}}{12}\|\boldsymbol{r}_{j;\widetilde{n}}\|_2 - \frac{\sqrt{23}}{12}\epsilon\tau' - \left\|\widetilde{A}_j \left(\boldsymbol{r} - \tau'\boldsymbol{r}'\right)_{j;\widetilde{n}}\right\|_2 \left\|\boldsymbol{g}^k\right\|_2 \\
&\geq \frac{\sqrt{23}}{12}\|\boldsymbol{r}_{j;\widetilde{n}}\|_2 - \frac{\sqrt{23}}{12}\epsilon\tau' - \frac{3}{2}\sqrt{m}\left\|(\boldsymbol{r} - \tau'\boldsymbol{r}')_{j;\widetilde{n}}\right\|_2 \\
&= \frac{1}{3}\|\boldsymbol{r}_{j;\widetilde{n}}\|_2 \left(\frac{\sqrt{23}}{4} - \frac{\sqrt{23}\epsilon}{4} - \frac{9}{2}\epsilon\sqrt{m}\right).
\end{aligned}$$

On the other hand,

$$\begin{aligned}
\left|\langle h_{j;k}, T_{j;\widetilde{n}}\rangle_{(\mathcal{D}_j,\nu_j)}\right| &\leq \left|\langle \tau' h'_{j;k}, T_{j;\widetilde{n}}\rangle_{(\mathcal{D}_j,\nu_j)}\right| + \left|\langle \left(h_{j;k} - \tau' h'_{j;k}\right), T_{j;\widetilde{n}}\rangle_{(\mathcal{D}_j,\nu_j)}\right| \\
&\leq \frac{9}{4}\tau'\|\boldsymbol{r}'_{j;\widetilde{n}}\|_2 + \left|\sum_{\ell\in[m]} g_\ell^k \left(\widetilde{A}_j \left(\boldsymbol{r} - \tau'\boldsymbol{r}'\right)_{j;\widetilde{n}}\right)_\ell\right|.
\end{aligned}$$

As above, we obtain using (4.14) that

$$\begin{aligned}
\left|\langle h_{j;k}, T_{j;\widetilde{n}}\rangle_{(\mathcal{D}_j,\nu_j)}\right| &\leq \frac{9}{4} \left(\|\boldsymbol{r}_{j;\widetilde{n}}\|_2 + \|(\tau'\boldsymbol{r}' - \boldsymbol{r})_{j;\widetilde{n}}\|_2\right) + \left|\left\langle \widetilde{A}_j \left(\boldsymbol{r} - \tau'\boldsymbol{r}'\right)_{j;\widetilde{n}}, \boldsymbol{g}^k\right\rangle\right| \\
&\leq \frac{9}{4}\|\boldsymbol{r}_{j;\widetilde{n}}\|_2 + \frac{9}{4}\epsilon\tau' + \left|\left\langle \widetilde{A}_j \left(\boldsymbol{r} - \tau'\boldsymbol{r}'\right)_{j;\widetilde{n}}, \boldsymbol{g}^k\right\rangle\right| \\
&\leq \frac{9}{4}\|\boldsymbol{r}_{j;\widetilde{n}}\|_2 + \frac{9}{4}\epsilon\tau' + \left\|\widetilde{A}_j \left(\boldsymbol{r} - \tau'\boldsymbol{r}'\right)_{j;\widetilde{n}}\right\|_2 \left\|\boldsymbol{g}^k\right\|_2 \\
&\leq \frac{9}{4}\|\boldsymbol{r}_{j;\widetilde{n}}\|_2 + \frac{9}{4}\epsilon\tau' + \frac{3}{2}\sqrt{m}\left\|(\boldsymbol{r} - \tau'\boldsymbol{r}')_{j;\widetilde{n}}\right\|_2 \\
&= \frac{9}{4}\|\boldsymbol{r}_{j;\widetilde{n}}\|_2 \left(1 + \epsilon + \frac{2}{3}\epsilon\sqrt{m}\right).
\end{aligned}$$

Once again, making sure to use, e.g., an $\epsilon \leq \left(6\sqrt{\alpha' m}\right)^{-1}$ will now ensure property two for h as well. $\qquad\square$

Lemma 6. *Let $\mathcal{H}_{2s}$ be the set of all functions h whose coefficient vectors are $2s$-sparse, and let $\boldsymbol{r}_h \in \mathbb{C}^N$ denote the coefficient vector for each $h \in \mathcal{H}_{2s}$. Furthermore, let $\delta \in (0, 7/16)$, $p \in (0, 1)$, $L \in \mathbb{N}$ be odd, and assume that $m \geq \widetilde{\beta}' K^2 \delta^{-2} s \max \{ d \log^2(4s) \log(9m)$ $\log \left(\frac{8e(D-1)DM}{d} \right), \log \left(\frac{2D}{p} \right) \}$ and $L \geq \gamma' s \cdot d \cdot \log \left(\frac{DM}{\sqrt[d]{sd\ (p/2)^{1/s}}} \right)$ for sufficiently large absolute constants $\widetilde{\beta}', \gamma' \in \mathbb{R}^+$. Then, with probability greater than $1 - p$, all of the following will hold for all $(h, j, \widetilde{n}) \in \mathcal{H}_{2s} \times [D] \times [M]$: Both*

1. $\|h_{j;k}\|^2_{L^2(\mathcal{D}_j, \nu_j)} \leq (\alpha' + 1) \|\boldsymbol{r}_h\|^2_2$ *for the absolute constant α' defined in Lemma 4, and*

2. $\frac{9}{2} \|(\boldsymbol{r}_h)_{j;\widetilde{n}}\|_2 \geq \left| \langle h_{j;k}, T_{j;\widetilde{n}} \rangle_{(\mathcal{D}_j, \nu_j)} \right| \geq \frac{1}{3} \|(\boldsymbol{r}_h)_{j;\widetilde{n}}\|_2$

will be simultaneously true for more than half of the $k \in [L]$.

Proof. Let A be the event that for all $(h, j, \widetilde{n}) \in \mathcal{H}_{2s} \times [D] \times [M]$, the properties 1 and 2 in Theorem 4 simultaneously hold for more than half of the $k \in [L]$, and let B be the event that the restricted isometry constant δ_{2s} of $\widetilde{A}_j$ satisfies $\delta_{2s} \leq \delta$ for all $j \in [D]$. By the Theorem 4 and Lemma 3 with properly chosen parameters including L and m, both $\mathbb{P}\left[A \mid B\right]$ and $\mathbb{P}[B]$ are greater than $1 - p/2$. We obtain, by Bayes' theorem,

$$1 - p \leq \left(1 - \frac{p}{2}\right) \left(1 - \frac{p}{2}\right) \leq \mathbb{P}\left[A \mid B\right] \mathbb{P}\left[B\right] = \mathbb{P}[A \cap B] \leq P[A],$$

which establishes the desired result. □

The results from Lemma 6 can be exploited in our algorithm as follows. In the entry identification, by taking the median over $k \in [L]$ of $\left| \langle h_{j;k}, T_{j;\widetilde{n}} \rangle_{(\mathcal{D}_j, \nu_j)} \right|$, we get a nonzero value if $\left\|(\boldsymbol{r}_h)_{j;\widetilde{n}}\right\|_2$ is nonzero and a zero value if $\left\|(\boldsymbol{r}_h)_{j;\widetilde{n}}\right\|_2$ is zero, especially due to the second property in Lemma 6 being satisfied for more the half of $k \in [L]$. Thus, we store all those $\widetilde{n}$ with nonzero median value in $\mathcal{N}_j$. On the other hand, the summation over $\widetilde{n} \in [M]$ of $\mathrm{median}_k \left| \langle h_{j;k}, T_{j;\widetilde{n}} \rangle_{(\mathcal{D}_j, \nu_j)} \right|$ can be also used for the halting criterion in our algorithm. Although $\mathcal{O}(\|\boldsymbol{c}\|_2/\eta)$ iterations guarantee the desired precision, it is not necessary to repeat the iteration if the residual $\boldsymbol{r}$ already has small energy. For any $j \in [D]$, if

$$\sum_{\widetilde{n} \in [M]} \left| \mathrm{median}_k |\langle h_{j;k}, T_{j;\widetilde{n}} \rangle_{(\mathcal{D}_j, \nu_j)}| \right|^2 \leq \frac{1}{9} \eta^2,$$

then $\|\boldsymbol{r}\|_2 \leq \eta$ by using the lower bound in the Property 2 of Lemma 6.

4.2 Pairing

In the entry identification, we can find at most $2s$ entries belonging to $\mathcal{N}_j^t := \{n_j \mid \boldsymbol{n} \in \mathrm{supp}(\boldsymbol{r})\} \subset [M]$ for each $j \in [D]$. Now, the question is how to combine the entries of each $\mathcal{N}_j \supseteq \mathcal{N}_j \cap \mathcal{N}_j^t$ in order to correctly identify the corresponding energetic elements of $\mathrm{supp}(\boldsymbol{r})$ efficiently. In order to do this, in the pairing process briefly introduced in Section 3.2, we successively build up the prefix set $\mathcal{P}_j$ of the energetic pairs for all $j \in [D] \setminus \{0\}$ such that $\mathcal{P}_j \supset \left\{ \widetilde{\boldsymbol{n}} \in \mathcal{P}_{j-1} \times \mathcal{N}_j \mid \|\boldsymbol{r}_{j;(\widetilde{\boldsymbol{n}},\cdots)}\|_2^2 \geq \frac{\|r\|_2^2}{\alpha^2 s} \right\}$ with the initialization of $\mathcal{P}_0 = \mathcal{N}_0$. The prefix set $\mathcal{P}_j$ contains only $2s$ pairs throwing out the other pairs with smaller energy for each $j \in [D] \setminus \{0\}$ so that $\mathcal{P}_{D-1} \supset \left\{ \widetilde{\boldsymbol{n}} \in \mathrm{supp}(\boldsymbol{r}) \mid |r_{\widetilde{\boldsymbol{n}}}|^2 \geq \frac{\|r\|_2^2}{\alpha^2 s} \right\}$. From (2.9), the energy $\left\|\boldsymbol{r}_{j;(\widetilde{\boldsymbol{n}},\cdots)}\right\|_2^2$ corresponding to each $\widetilde{\boldsymbol{n}} \in [M]^{j+1}$ has the following equality,

$$\left\|\boldsymbol{r}_{j;(\widetilde{\boldsymbol{n}},\cdots)}\right\|_2^2 = \left\|\langle h, T_{j;\widetilde{\boldsymbol{n}}}\rangle_{L^2\left(\times_{i\in[j+1]}\mathcal{D}_i, \otimes_{i\in[j+1]}\nu_i\right)}\right\|_{L^2\left(\mathcal{D}_j'', \boldsymbol{\nu}_j''\right)}^2. \tag{4.15}$$

The energy is estimated in Algorithm 3 by using the following estimator $E_{j;(\widetilde{\boldsymbol{n}},\cdots)}$ defined as

$$E_{j;(\widetilde{\boldsymbol{n}},\cdots)} := \frac{1}{m_2}\sum_{k\in[m_2]}\left|\frac{1}{m_1}\sum_{\ell\in[m_1]}h(\boldsymbol{w}_{j,\ell}, \boldsymbol{z}_{j,k})\overline{T_{j;\widetilde{\boldsymbol{n}}}(\boldsymbol{w}_{j,\ell})}\right|^2$$

which approximates the right hand side of (4.15) by using only a finite evaluations of h. Those sampling point sets $\mathcal{W}_j \times \mathcal{Z}_j$ for all $j \in [D] \setminus \{0\}$ are constructed from $\mathcal{W}_j := \{\boldsymbol{w}_{j,\ell}\}_{\ell\in[m_1]}$ and $\mathcal{Z}_j := \{\boldsymbol{z}_{j,k}\}_{k\in[m_2]}$ where $\boldsymbol{w}_{j,\ell}$ and $\boldsymbol{z}_{j,k}$ are chosen independently at random from $\times_{i\in[j+1]}\mathcal{D}_i$ and $\mathcal{D}_j''$ for $j \in [D-1] \setminus \{0\}$, respectively. If $j = D-1$, $\mathcal{W}_{D-1} := \{\boldsymbol{w}_{D-1,\ell}\}_{\ell\in[m_1]}$ is chosen from $\times_{i\in[D]}\mathcal{D}_i$ and $\mathcal{Z}_{D-1} = \emptyset$. Note that $\mathcal{W}_j \times \mathcal{Z}_j = \mathcal{G}_j^P$ from Section 1.4. Furthermore, the sets $\mathcal{W}_j \times \mathcal{Z}_j$ for all $j \in [D] \setminus \{0\}$ build a random sampling matrix A^P in (2.13) which explicitly expresses the samples(evaluations) of the $2s$-sparse h used in the (4.16) as $A^P \boldsymbol{r}$. The matrix A^P is broken into smaller matrices B_j and C_j for $j \in [D] \setminus \{0\}$ defined and explained in the next paragraph for the complete analysis of the pairing process in the upcoming lemmas and theorems in this section.

For all $j \in [D-1] \setminus \{0\}$, the measurement matrix $C_j \in \mathbb{C}^{m_2 \times [M]^{D-j-1}}$ is defined as

$$(C_j)_{k,\boldsymbol{n}_2} := \frac{1}{\sqrt{m_2}}T_{\boldsymbol{n}_2}''(\boldsymbol{z}_{j,k}), \quad k \in [m_2], \ \boldsymbol{n}_2 \in [M]^{D-j-1}, \tag{4.16}$$

where $T_{\boldsymbol{n}_2}''(\boldsymbol{y})$ is a partial product of the last $D-j-1$ terms of $T_{\boldsymbol{n}}(\boldsymbol{x})$ defined in (1.1), i.e.,

$$T_{\boldsymbol{n}_2}''(\boldsymbol{y}) = \prod_{i\in[D-j-1]} T_{i+j+1;n_{i+j+1}}(y_i), \quad \boldsymbol{y} \in \mathcal{D}_j''.$$

The matrix C_j can be restricted to the matrix of size $m_2 \times \widetilde{N}_j$ when C_j is applied to $\boldsymbol{v}_{j,(\widetilde{n},\cdots)}$ where $\widetilde{N}_j$ estimated in (2.6) is the cardinality of the superset of any possible $\mathrm{supp}(\boldsymbol{v}_{j;\widetilde{n}})$ with fixed $j, \widetilde{n}, D$ and d so that it satisfies RIP with sufficiently large m_2. When the $j = D - 1$, the matrix C_j is defined to be 1 since $\mathcal{Z}_{D-1} = \emptyset$. For all $j \in [D] \setminus \{0\}$, on the other hand, the matrices $B_j \in \mathbb{C}^{m_1 \times [M]^{j+1}}$ is defined as

$$(B_j)_{\ell, \boldsymbol{n}_1} := \frac{1}{\sqrt{m_1}} T'_{\boldsymbol{n}_1}(\boldsymbol{w}_{j,\ell}), \quad \ell \in [m_1], \ \boldsymbol{n}_1 \in [M]^{j+1}, \tag{4.17}$$

where $T'_{\boldsymbol{n}_1}(\boldsymbol{z})$ is a partial product of the first $j + 1$ terms of $T_{\boldsymbol{n}}(\boldsymbol{x})$ in (1.1), i.e.,

$$T'_{\boldsymbol{n}_1}(\boldsymbol{z}) = \prod_{i \in [j+1]} T_{i;n_i}(z_i), \quad \boldsymbol{z} \in \times_{i \in [j+1]} \mathcal{D}_i.$$

The matrix B_j can be restricted to the matrix of size $m_1 \times \bar{N}_j$ where $\bar{N}_j = \binom{j+1}{d} M^d$ if $j + 1 \geq d$, or M^d otherwise. The number $\bar{N}_j$ is the cardinality of the set of any possible prefix $\widetilde{n} \in [M]^{j+1}$ with fixed $j, \widetilde{n}$ and d. The sampling numbers m_1 and m_2 are chosen for all B_j and C_j with any $j \in [D] \setminus \{0\}$ to satisfy RIP with the restricted isometry constants $\widetilde{\delta}_{2s+1} \leq \widetilde{\delta}$ and $\delta'_{2s} \leq \delta'$, respectively. We again mention that $C_{D-1} = 1$.

Lemma 7. *Let $\mathcal{H}_{2s}$ be the set of all functions h whose coefficient vectors are $2s$-sparse and let $\boldsymbol{r}_h \in \mathbb{C}^N$ denote the coefficient vector for each $h \in \mathcal{H}_{2s}$. Let $j \in [D] \setminus \{0\}$, and $\widetilde{\delta}$ and δ' be chosen from $(0,1)$. Assume that B_j and C_j satisfy RIP with $\widetilde{\delta}_{2s+1} \leq \widetilde{\delta}$ and $\delta'_{2s} \leq \delta'$, respectively. Then, denoting $\boldsymbol{r} := \boldsymbol{r}_h$ for simplicity,*

$$(1 - \delta'_{2s}) \left(\max\{0, \left\| \boldsymbol{r}_{j;(\widetilde{n},\cdots)} \right\|_2 - \widetilde{\delta}_{2s+1} \|\boldsymbol{r}\|_2 \} \right)^2 \leq E_{j;(\widetilde{n},\cdots)}$$

$$\leq (1 + \delta'_{2s}) \left(\left\| \boldsymbol{r}_{j;(\widetilde{n},\cdots)} \right\|_2 + \widetilde{\delta}_{2s+1} \|\boldsymbol{r}\|_2 \right)^2. \tag{4.18}$$

for any $\widetilde{n} \in [M]^{j+1}$.

Proof. As $j \in [D] \setminus \{0\}$ is fixed, for simplicity, we use notation $\boldsymbol{w}_\ell$ and $\boldsymbol{z}_k$ for sampling points instead of $\boldsymbol{w}_{j,\ell}$ and $\boldsymbol{z}_{j,k}$ constructing the sampling matrices B_j and C_j as in (4.17) and (4.16), respectively. Fix $\widetilde{n} \in [M]^{j+1}$. Letting $\boldsymbol{n} = (\boldsymbol{n}_1, \boldsymbol{n}_2)$, $\boldsymbol{n}_1 \in [M]^{j+1}$ and $\boldsymbol{n}_2 \in [M]^{D-j-1}$, we can rewrite the energy estimate $E_{j;(\widetilde{n},\cdots)}$ as follows,

$$E_{j;(\widetilde{n},\cdots)} = \frac{1}{m_2} \sum_{k \in [m_2]} \left| \frac{1}{m_1} \sum_{\ell \in [m_1]} h(\boldsymbol{w}_\ell, \boldsymbol{z}_k) \overline{T_{j;\widetilde{n}}(\boldsymbol{w}_\ell)} \right|^2$$

$$
= \frac{1}{m_2} \sum_{k \in [m_2]} \left| \frac{1}{m_1} \sum_{\ell \in [m_1]} \sum_{\bm{n} = (\bm{n}_1, \bm{n}_2) \in \mathrm{supp}(\bm{r})} r_{\bm{n}} T_{\bm{n}}(\bm{w}_\ell, \bm{z}_k) \overline{T_{j; \widetilde{\bm{n}}}(\bm{w}_\ell)} \right|^2
$$

$$
= \frac{1}{m_2} \sum_{k \in [m_2]} \frac{1}{m_1^2} \left| \sum_{\bm{n} \in \mathrm{supp}(\bm{r})} \sum_{\ell \in [m_1]} r_{\bm{n}} T'_{\bm{n}_1}(\bm{w}_\ell) \overline{T_{j; \widetilde{\bm{n}}}(\bm{w}_\ell)} T''_{\bm{n}_2}(\bm{z}_k) \right|^2
$$

$$
=: \frac{1}{m_2} \sum_{k \in [m_2]} \frac{1}{m_1^2} \left| \sum_{\substack{\bm{n}_2 \text{ s.t. } \exists \bm{n}_1 \\ \text{with } (\bm{n}_1, \bm{n}_2) \in \mathrm{supp}(\bm{r})}} (\widetilde{r}_{\widetilde{\bm{n}}})_{\bm{n}_2} T''_{\bm{n}_2}(\bm{z}_k) \right|^2 , \tag{4.19}
$$

where

$$
(\widetilde{r}_{\widetilde{\bm{n}}})_{\bm{n}_2} := \sum_{\substack{\bm{n}_1 \text{ s.t.} \\ \bm{n} = (\bm{n}_1, \bm{n}_2) \in \mathrm{supp}(\bm{r})}} r_{\bm{n}} \sum_{\ell \in [m_1]} T'_{\bm{n}_1}(\bm{w}_\ell) \overline{T_{j; \widetilde{\bm{n}}}(\bm{w}_\ell)}. \tag{4.20}
$$

We can construct a vector $\widetilde{\bm{r}} := \big((\widetilde{r}_{\widetilde{\bm{n}}})_{\bm{n}_2} \big) \in \mathbb{C}^{\widetilde{N}_j}$ with entries $(\widetilde{r}_{\widetilde{\bm{n}}})_{\bm{n}_2}$ at $\bm{n}_2$ so that $\widetilde{\bm{r}}$ has a support whose cardinality is at most $2s$ since h is $2s$-sparse. Thus, the energy estimate $E_{j;(\widetilde{\bm{n}}, \cdots)}$ in (4.19) can be expressed as $\left\| C_j \left(\frac{\widetilde{\bm{r}}}{m_1} \right) \right\|_2^2$. Since the restricted measurement matrix $C_j \in \mathbb{C}^{m_2 \times \widetilde{N}_j}$ satisfies the RIP,

$$
(1 - \delta'_{2s}) \left\| \frac{\widetilde{\bm{r}}}{m_1} \right\|_2^2 \leq \left\| C_j \left(\frac{\widetilde{\bm{r}}}{m_1} \right) \right\|_2^2 \leq (1 + \delta'_{2s}) \left\| \frac{\widetilde{\bm{r}}}{m_1} \right\|_2^2. \tag{4.21}
$$

In order to get an upper bound and lower bound of $\left\| \frac{\widetilde{\bm{r}}}{m_1} \right\|_2$, we define $\bm{e}_{\widetilde{\bm{n}}} \in \mathbb{C}^{\bar{N}_j}$ as a standard basis vector with all 0 entries except for a 1 at $\widetilde{\bm{n}}$, and $R_j \in \mathbb{C}^{\widetilde{N}_j \times \bar{N}_j}$ as

$$
(R_j)_{\bm{n}_2, \bm{n}_1} := \begin{cases} r_{(\bm{n}_1, \bm{n}_2)} & \text{if } (\bm{n}_1, \bm{n}_2) \in \mathrm{supp}(\bm{r}) \\ 0 & \text{otherwise .} \end{cases}
$$

Note that R_j contains at most $2s$ nonzero elements since h is $2s$-sparse, and $\widetilde{\bm{n}}$ is any element in $[M]^{j+1}$. Set $\mathcal{Q} := \{ \widetilde{\bm{n}} \} \cup \{ \bm{n}_1 \in [M]^{j+1} \mid \exists \bm{n}_2 \in [M]^{D-j-1} \text{ such that } (\bm{n}_1, \bm{n}_2) \in \mathrm{supp}(\bm{r}) \}$ with a fixed $\widetilde{\bm{n}} \in [M]^{j+1}$. Thus, the cardinality of $\mathcal{Q}$ is at most $2s+1$. Orderings of indices $\bm{n}_1$ and $\bm{n}_2$ depend on the column orderings of B_j and C_j, respectively. We note that the $\widetilde{\bm{n}}$-th column of R_j is $\bm{r}_{j;(\widetilde{\bm{n}}, \cdots)}$. Both bounds of $\left\| \frac{\widetilde{\bm{r}}}{m_1} \right\|_2$ are found as follows

$$
\left\| \frac{\widetilde{\bm{r}}}{m_1} - \bm{r}_{j;(\widetilde{\bm{n}}, \cdots)} \right\|_2 = \left\| R_j (B_j)_{\mathcal{Q}}^* (B_j)_{\mathcal{Q}} \bm{e}_{\widetilde{\bm{n}}} - R_j \bm{e}_{\widetilde{\bm{n}}} \right\|_2
$$

$$
\leq \| R_j \|_{2 \to 2} \| (B_j)_{\mathcal{Q}}^* (B_j)_{\mathcal{Q}} - I \|_{2 \to 2} \| \bm{e}_{\widetilde{\bm{n}}} \|_2
$$

$$\leq \widetilde{\delta}_{2s+1}\|R_j\|_F$$
$$= \widetilde{\delta}_{2s+1}\|\boldsymbol{r}\|_2$$

and therefore,

$$\left| \left\|\frac{\widetilde{\boldsymbol{r}}}{m_1}\right\|_2 - \left\|\boldsymbol{r}_{j;(\widetilde{\boldsymbol{n}},\cdots)}\right\|_2 \right| \leq \widetilde{\delta}_{2s+1}\|\boldsymbol{r}\|_2,$$

$$\left\|\boldsymbol{r}_{j;(\widetilde{\boldsymbol{n}},\cdots)}\right\|_2 - \widetilde{\delta}_{2s+1}\|\boldsymbol{r}\|_2 \leq \left\|\frac{\widetilde{\boldsymbol{r}}}{m_1}\right\|_2 \leq \left\|\boldsymbol{r}_{j;(\widetilde{\boldsymbol{n}},\cdots)}\right\|_2 + \widetilde{\delta}_{2s+1}\|\boldsymbol{r}\|_2. \tag{4.22}$$

Combining (4.21) and (4.22), we reach the conclusion in (4.18). $\qquad\square$

Lemma 8. *Let $\mathcal{H}_{2s}$ be the set of all functions h whose coefficient vectors are $2s$-sparse and let $\boldsymbol{r}_h \in \mathbb{C}^N$ denote the coefficient vector for each $h \in \mathcal{H}_{2s}$. Let j be any integer such that $j \in [D] \setminus \{0\}$, and $\widetilde{\delta}$ and δ' be chosen from $(0,1)$. Given $\alpha > 1$, assume that the restricted B_j and C_j satisfy RIP with $\widetilde{\delta}_{2s+1} \leq \widetilde{\delta} \leq \frac{1}{\check{c}\sqrt{s}}$ for some $\check{c} > \sqrt{2}$ and $\delta'_{2s} \leq \delta' \in (0,1)$, respectively, with $\sqrt{\frac{1+\delta'}{1-\delta'}} < \frac{\check{c}}{\alpha} - 1$. Then, denoting $\boldsymbol{r} := \boldsymbol{r}_h$ for simplicity, one has the set $\mathcal{P}_j \supset \left\{\widetilde{\boldsymbol{n}} \in [M]^{j+1} \mid \|\boldsymbol{r}_{j;(\widetilde{\boldsymbol{n}},\cdots)}\|_2^2 \geq \frac{\|\boldsymbol{r}\|_2^2}{\alpha^2 s}\right\}$ of cardinality $2s$ resulting from Algorithm 3 if $\mathcal{P}_{j-1} \supset \left\{\hat{\boldsymbol{n}} \in [M]^j \mid \|\boldsymbol{r}_{j-1;(\hat{\boldsymbol{n}},\cdots)}\|_2^2 \geq \frac{\|\boldsymbol{r}\|_2^2}{\alpha^2 s}\right\}$.*

Proof. By assumption that $\mathcal{P}_{j-1} \subset \times_{i\in[j]}\mathcal{N}_i$ contains all prefixes in $\big\{\hat{\boldsymbol{n}} \in [M]^j \mid \|\boldsymbol{r}_{j-1;(\hat{\boldsymbol{n}},\cdots)}\|_2^2 \geq \frac{\|\boldsymbol{r}\|_2^2}{\alpha^2 s}\big\}$, $\mathcal{P}_{j-1} \times \mathcal{N}_j$ contains all possible prefixes $\widetilde{\boldsymbol{n}} \in [M]^{j+1}$ with $\|\boldsymbol{r}_{j;(\widetilde{\boldsymbol{n}},\cdots)}\|_2^2 \geq \frac{\|\boldsymbol{r}\|_2^2}{\alpha^2 s}$ by the definition of $\mathcal{P}_j$ and $\mathcal{N}_j$ for all $j \in [D]$. If $\left\|\boldsymbol{r}_{j;(\widetilde{\boldsymbol{n}},\cdots)}\right\|_2 = 0$, i.e., there is no $\boldsymbol{n}_2$ such that $(\widetilde{\boldsymbol{n}}, \boldsymbol{n}_2) \in \mathrm{supp}(\boldsymbol{r})$, then from Lemma 7,

$$0 \leq E_{j;(\widetilde{\boldsymbol{n}},\cdots)} \leq (1+\delta')\left(\widetilde{\delta}\|\boldsymbol{r}\|_2\right)^2 \leq (1+\delta')\left(\frac{\|\boldsymbol{r}\|_2}{\check{c}\sqrt{s}}\right)^2.$$

On the other hand, if $\left\|\boldsymbol{r}_{j;(\widetilde{\boldsymbol{n}},\cdots)}\right\|_2^2 \geq \frac{\|\boldsymbol{r}\|_2^2}{\alpha^2 s}$, i.e., there is $\boldsymbol{n}_2$ such that $(\widetilde{\boldsymbol{n}}, \boldsymbol{n}_2) \in \mathrm{supp}(\boldsymbol{r})$, then

$$(1-\delta')\left(\left\|\boldsymbol{r}_{j;(\widetilde{\boldsymbol{n}},\cdots)}\right\|_2 - \frac{\|\boldsymbol{r}\|_2}{\check{c}\sqrt{s}}\right)^2 \leq (1-\delta')\left(\left\|\boldsymbol{r}_{j;(\widetilde{\boldsymbol{n}},\cdots)}\right\|_2 - \widetilde{\delta}\|\boldsymbol{r}\|_2\right)^2$$

$$\leq E_{j;(\widetilde{\boldsymbol{n}},\cdots)} \leq (1+\delta')\left(\left\|\boldsymbol{r}_{j;(\widetilde{\boldsymbol{n}},\cdots)}\right\|_2 + \widetilde{\delta}\|\boldsymbol{r}\|_2\right)^2.$$

In order to distinguish nonzero $\left\|\boldsymbol{r}_{j;(\widetilde{\boldsymbol{n}},\cdots)}\right\|_2 \geq \frac{\|\boldsymbol{r}\|_2}{\alpha\sqrt{s}}$ from zero $\left\|\boldsymbol{r}_{j;(\widetilde{\boldsymbol{n}},\cdots)}\right\|_2$, we should have

$$(1+\delta')\left(\frac{\|\boldsymbol{r}\|_2}{\check{c}\sqrt{s}}\right)^2 < (1-\delta')\left(\frac{\|\boldsymbol{r}\|_2}{\alpha\sqrt{s}} - \frac{\|\boldsymbol{r}\|_2}{\check{c}\sqrt{s}}\right)^2$$

42

which is implied by

$$\sqrt{\frac{1+\delta'}{1-\delta'}} < \frac{\check{c}}{\alpha} - 1,$$

as in the assumption. Since estimates of zero energy and nonzero energy greater than $\frac{\|r\|_2^2}{\alpha^2 s}$ are separated, choosing $2s$ prefixes with largest estimates $E_{\widetilde{n}}$ guarantees that it contains all prefixes with energy greater than $\frac{\|r\|_2^2}{\alpha^2 s}$. $\square$

Theorem 5. *Let $\mathcal{H}_{2s}$ be the set of all functions h whose coefficient vectors are $2s$-sparse and let $r_h \in \mathbb{C}^N$ denote the coefficient vector for each $h \in \mathcal{H}_{2s}$. We assume that we have $\mathcal{N}_j$ for all $j \in [D]$. Let $\alpha > 1$, $\widetilde{\delta} \leq \frac{1}{\check{c}\sqrt{s}}$ for some $\check{c} > \sqrt{2}$ and $\delta' \in (0,1)$, satisfying $\sqrt{\frac{1-\delta'}{1+\delta'}} < \frac{\check{c}}{\alpha} - 1$, and $p \in (0,1)$. If*

$$m_1 \geq \bar{\alpha} K^2 \left(\widetilde{\delta}\right)^{-2} s \max\left\{\log^2(4s) \cdot d \cdot \log\left(\frac{\mathrm{e} M D^{1+\frac{1}{d\log^3(4s)}}}{d}\right)\log(9m_1), \log(2D/p)\right\} \quad and$$

$$m_2 \geq \bar{\beta} K^2 \left(\delta'\right)^{-2} s \max\left\{\log^2(4s) \cdot d \cdot \log\left(\frac{\mathrm{e} M D^{1+\frac{1}{d\log^3(4s)}}}{d}\right)\log(9m_2), \log(2D/p)\right\},$$

for absolute constants $\bar{\alpha}$ and $\bar{\beta}$, then denoting $r := r_h$ for simplicity, Algorithm 3 finds $\mathcal{P} \supset \left\{n \in [M]^D \mid |r_n|^2 \geq \frac{\|r\|_2^2}{\alpha^2 s}\right\}$ of $|\mathcal{P}| = 2s$ with probability at least $1 - p$.

Proof. Given m_1 and m_2, by Theorem 3, the probability of either B_j or C_j not satisfying $\widetilde{\delta}_{2s+1} < \widetilde{\delta}$ or $\delta'_{2s} < \delta'$ respectively is at most $\frac{p}{2D}$ for each $j \in [D] \setminus \{0\}$, and thus the union bound over all j yields the failure probability at most $\frac{p(D-1)}{D} < p$. That is, Theorem 3 ensures that B_j and C_j have RIP uniformly for all $j \in [D] \setminus \{0\}$ with probability at least $1 - p$. Repeatedly applying Lemma 8 yields the final $\mathcal{P}(= \mathcal{P}_{D-1}) \supset \left\{n \in [M]^D \mid |r_n|^2 \geq \frac{\|r\|_2^2}{\alpha^2 s}\right\}$ of cardinality $2s$ by combining the fact that $\mathcal{P}_0(= \mathcal{N}_0) \supset \left\{\widetilde{n} \in [M] \mid \|r_{(\widetilde{n},\cdots)}\|_2^2 \geq \frac{\|r\|_2^2}{\alpha^2 s}\right\}$. $\square$

4.3   Support Identification

In this section, it remains to combine the results of entry identification and pairing processes in order to give the complete support identification algorithm and to prove Lemma 2 which is the main ingredient of Theorem 2 in Section 3.3. The support identification starts with the entry identification providing $\mathcal{N}_j$, $j \in [D]$ as outputs, and in turn, the pairing takes $\mathcal{N}_j$, $j \in [D]$ as inputs and outputs $\mathcal{P}$ of cardinality $2s$ containing $\left\{n \in [M]^D \mid |r_n|^2 \geq \frac{\|r\|_2^2}{\alpha^2 s}\right\}$. Accordingly, we can get the following result.

Lemma 9. *The set $\mathcal{P}$ contains at most $2s$ index vectors satisfying*

$$\|\boldsymbol{r}_{\mathcal{P}^c}\|_2 \leq \sqrt{2s\frac{\|\boldsymbol{r}\|_2^2}{\alpha^2 s}} = \frac{\sqrt{2}\|\boldsymbol{r}\|_2}{\alpha},$$

where $\boldsymbol{r}_{\mathcal{P}^c}$ is the vector of $\boldsymbol{r}$ restricted to the complement of $\mathcal{P}$.

Proof. Note that $\boldsymbol{r}$ is $2s$-sparse and by Theorem 5 the squared magnitude of each $r_{\boldsymbol{n}}$ at $\mathcal{P}^c$ is less than $\frac{\|\boldsymbol{r}\|_2^2}{\alpha^2 s}$ so that we obtain the desired result. $\square$

Finally, we are ready to prove Lemma 2 in order to complete the analysis of support identification.

Proof of Lemma 2. By choosing $\alpha = 7$ and $\mathcal{P} = \Omega$ in Lemma 9, we obtain the desired upper bound of $\|(\boldsymbol{r}_h)_{\Omega^c}\|_2$ with probability at least $1 - 2p$ given the grids $\mathcal{G}^I$ and $\mathcal{G}^P$. The union bound of the failure probabilities p of Lemma 6 and Theorem 5 gives the desired probability. It remains to demonstrate the sampling complexity combining $|\mathcal{G}^I|$ and $|\mathcal{G}^P|$, and the runtime complexity combining Algorithms 2 and 3. The first term $|\mathcal{G}^I|$ is $m\mathcal{L}'D$ where m comes from Lemma 6, $\mathcal{L}'$ is defined as in Section 3.3 , and D, the number of changes in $j \in [D]$, therein. We emphasize that m samples are utilized repeatedly in order to implicitly construct L proxy functions combined with different Gaussian weights so that L does not affect the sampling complexity but affects the runtime complexity. The second term $|\mathcal{G}^P|$ is $m_1 m_2(D-2) + m_1$ where m_1 and m_2 are from Theorem 5, and $D - 2$ is the number of changes in $j \in [D-1] \setminus \{0\}$. We remind readers that $C_{D-1} = 1$ implied by $\mathcal{Z}_{D-1} = \emptyset$ so that m_1 samples are utilized instead of $m_1 m_2$ when $j = D - 1$. Now, we consider the runtime complexity. The first term in runtime complexity is $\mathcal{O}(mL\mathcal{L}D)$ from Algorithm 2 where mL computations are taken to implicitly construct the L proxy functions, $\mathcal{O}(\mathcal{L})$ is defined as in Section 3.3 , and D comes from the for loop from Algorithm 2. The second term in runtime complexity is $4s^2 \left(m_1 m_2(D-2) + m_1\right) + 4s^2 m_1 \sum_{j=1}^{D-1}(j+1)$ from Algorithm 3 since $4s^2$ energy estimates are calculated using the $m_1 m_2$ samples for each $j \in [D-1] \setminus \{0\}$ and m_1 samples for $j = D - 1$, and the evaluations of $T_{j;\widetilde{\boldsymbol{n}}}(\boldsymbol{w}_{j,\ell})$ are calculated for all $\widetilde{\boldsymbol{n}} \in \mathcal{P}_j$, $\forall j \in [D] \setminus \{0\}$. Here, it is assumed that it takes $\mathcal{O}(1)$ runtime to evaluate each ith component $T_{i;\widetilde{n}_i}((w_{j,\ell})_i)$ of $T_{j;\widetilde{\boldsymbol{n}}}(\boldsymbol{w}_{j,\ell})$. $\square$

5 Empirical Evaluation

In this section Algorithm 1 is evaluated numerically and compared to CoSaMP [50], its superlinear-time progenitor. Both Algorithm 1 and CoSaMP were implemented in

MATLAB for this purpose. All code used to produce the plots below is publicly available at [11].

5.1 Experimental Setup

We consider two kinds of tensor product basis functions below: Fourier and Chebyshev. In both cases each parameter, M, D, and s, is changed while the others remain fixed so that we can see how each parameter affects the runtime, sampling number, memory usage, and error of both Algorithm 1 and CoSaMP. For all experiments below $d = D$ so that $\mathcal{I} = [M]^D$. Every data point in every plot below was created using 100 different randomly generated trial signals, f, of the form

$$f(\boldsymbol{x}) = \sum_{\boldsymbol{n} \in \mathcal{S}} c_{\boldsymbol{n}} T_{\boldsymbol{n}}(\boldsymbol{x}), \tag{5.1}$$

where each function's support set, $\mathcal{S}$, contained s index vectors $\boldsymbol{n} \in [M]^D$ each of which was independently chosen uniformly at random from $[M]^D$, and where each function's coefficients $c_{\boldsymbol{n}}$ were each independently chosen uniformly at random from the unit circle in the complex plane (i.e., each $c_{\boldsymbol{n}} = \mathbb{e}^{\mathbb{i}\theta}$ where θ is chosen uniformly at random from $[0, 2\pi]$). In the Fourier setting the basis functions $T_{\boldsymbol{n}}(\boldsymbol{x})$ in (5.1) were chosen as per (5.2), and in the Chebyshev setting as per (5.3).

Below a *trial* will always refer to the execution of Algorithm 1 and/or CoSaMP on a particular randomly generated trial function f in (5.1). A *failed trail* will refer to any trial where either CoSaMP or Algorithm 1 failed to recover the correct support set $\mathcal{S}$ for f. Herein the parameters of both Algorithm 1 and CoSaMP were tuned to keep the number of failed trials down to less than 10 out of the total 100 used to create every data point in every plot. Finally, in all of our plots Algorithm 1 is graphed with red, and CoSaMP with blue.

5.2 Experiments with the Fourier Basis for $\mathcal{D} = [0, 1]^D$

In this section we consider the Fourier tensor product basis

$$T_{\boldsymbol{n}}(\boldsymbol{x}) := \prod_{j=0}^{D-1} \mathbb{e}^{2\pi \mathbb{i} n_j x_j} \tag{5.2}$$

whose orthogonality measure is the Lebesgue measure on $\mathcal{D} = [0, 1]^D$. In Figures 3, 4, 5 and 6, results are shown for approximating Fourier-sparse trial functions (5.1) using noise-less samples $\boldsymbol{y}$. In Figures 3 and 6a, the parameter M changes over the set $\{10, 20, 40, 80\}$
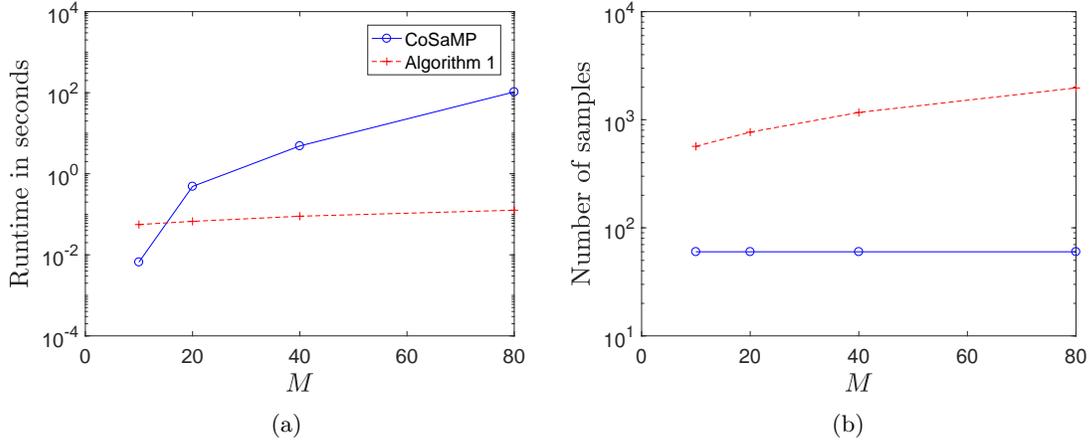
Figure 3: Fourier basis, $M \in \{10, 20, 40, 80\}$, $D = 4$, $s = 5$
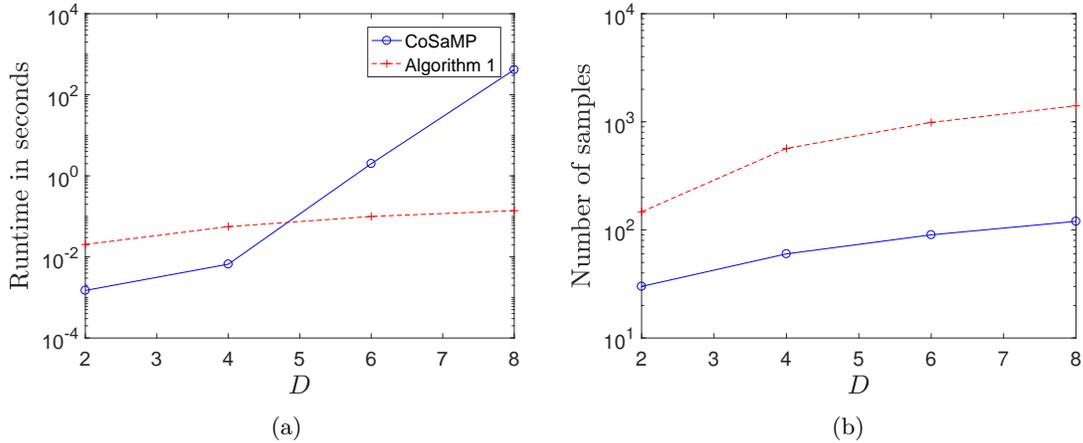
Figure 4: Fourier basis, $M = 10$, $D = \{2, 4, 6, 8\}$, $s = 5$

while $D = 4$ and $s = 5$ are held constant. In Figure 3a, the average runtime (in seconds) is shown as M changes. The average here is calculated over all 100 trials at each data point excluding any failed trials. As we can see, the runtime of Algorithm 1 grows very slowly as M grows, whereas the runtime grows fairly quickly for CoSaMP since its measurement matrix's size increases significantly as M grows. Figure 3b shows the number of samples used by both CoSaMP and Algorithm 1. We can see that Algorithm 1 requires more samples due mainly to its support identification's pairing step. On the other hand, we
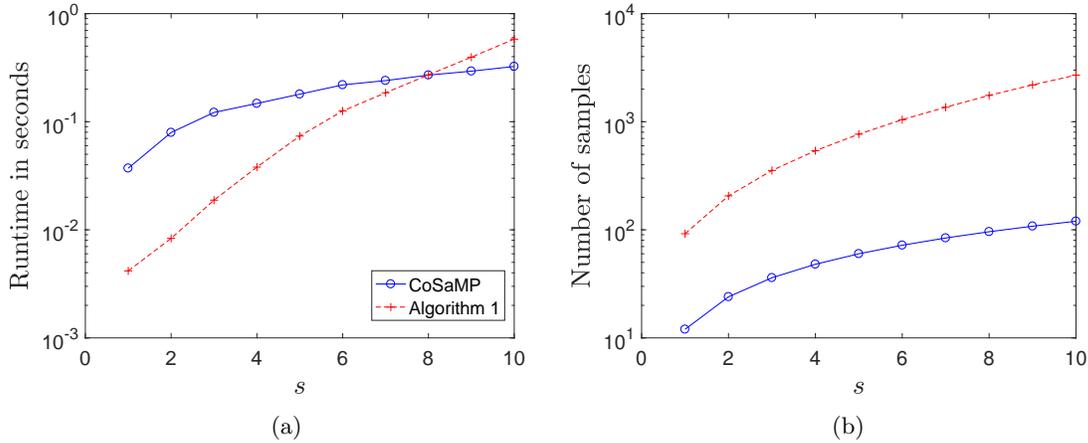
Figure 5: Fourier basis, $M = 20, \ D = 4, \ s = \{1, 2, 3, \cdots, 10\}$
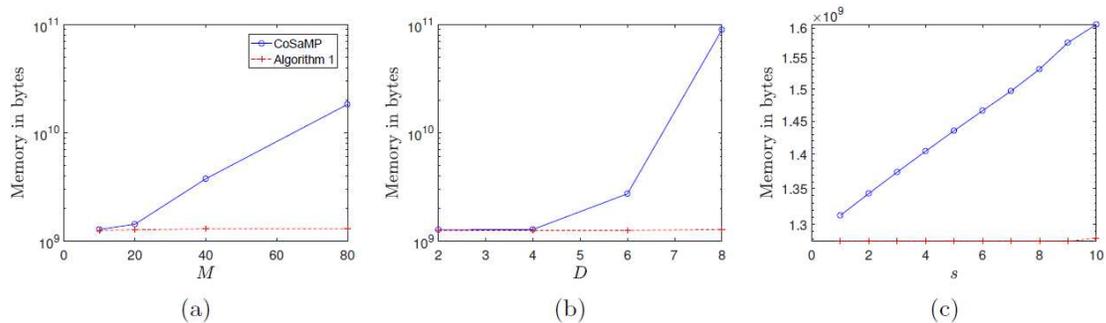
Figure 6: Fourier basis, memory usage when varying M, D and s

can see in Figure 6a that the memory usage of CoSaMP grows very rapidly compared to the slow growth of Algorithm 1's memory usage. This exemplifies the tradeoff between Algorithm 1 and CoSaMP – Algorithm 1 uses more samples than CoSaMP in order to reduce its runtime complexity and memory usage for large D and M. Finally, both methods produce outputs whose average errors (over the trials where they don't fail) are on the order of 10^{-15} to 10^{-14}, which is also observed in all other experiments in Sections 5.2, 5.3 and 5.4.

In Figures 4 and 6b, the number of dimensions, D, changes while both $M = 10$ and $s = 5$ are held fixed. Here, we can clearly see the advantage of Algorithm 1 for functions of many variables. The runtime and memory usage of CoSaMP blow up quickly

47
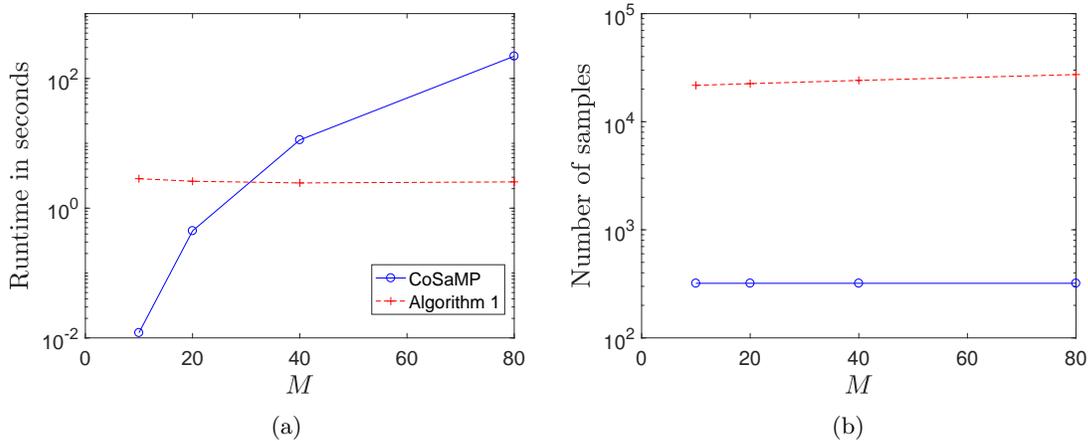
Figure 7: Chebyshev basis, $M \in \{10, 20, 40, 80\}$, $D = 4$, $s = 5$

as D increases due to the gigantic matrix-vector multiplies it requires to identify support. Algorithm 1, on the other hand, shows much slower growth in runtime and memory usage. When $D = 10$, for example, CoSaMP requires terabytes of memory whereas Algorithm 1 requires only a few gigabytes. In Figures 5 and 6c, s varies in $\{1, 2, 3, \cdots, 10\}$ while $M = 20$ and $D = 4$ are fixed. Since Algorithm 1 has $\tilde{\mathcal{O}}(s^5)$ scaling[13] in runtime due to its pairing step, it suffers as sparsity increases more quickly than CoSaMP does. Note that the crossover point is around $s = 8$, so that Algorithm 1 appears to be slower than CoSaMP for all $s > 8$ when $M = 20$ and $D = 4$. Though "only polynomial in s", it is clear from these experiments that the runtime scaling of Algorithm 3 in s needs to be improved before the methods proposed herein can become truly useful in practice.

5.3 Experiments with the Chebyshev Basis for $\mathcal{D} = [-1, 1]^D$

In this section we consider the Chebyshev tensor product basis

$$T_{\boldsymbol{n}}(\boldsymbol{x}) := 2^{\frac{1}{2}\|\boldsymbol{n}\|_0} \prod_{j=0}^{D-1} \cos\left(n_j \arccos(x_j)\right) \tag{5.3}$$

whose orthogonality measure is $d\boldsymbol{\nu} = \otimes_{j \in [D]} \frac{dx_j}{\pi \sqrt{1 - x_j^2}}$ on $\mathcal{D} = [-1, 1]^D$. Runtime and sampling complexity graphs are provided in Figures 7, 8 and 9 as M, D, and s vary,

---

[13]The $\tilde{\mathcal{O}}$ complexity notation here neglects all logarithmic factors while simultaneously holding D, K, d, η, $\|\mathbf{c}_f\|_2$ and $\mathcal{L}$ constant.
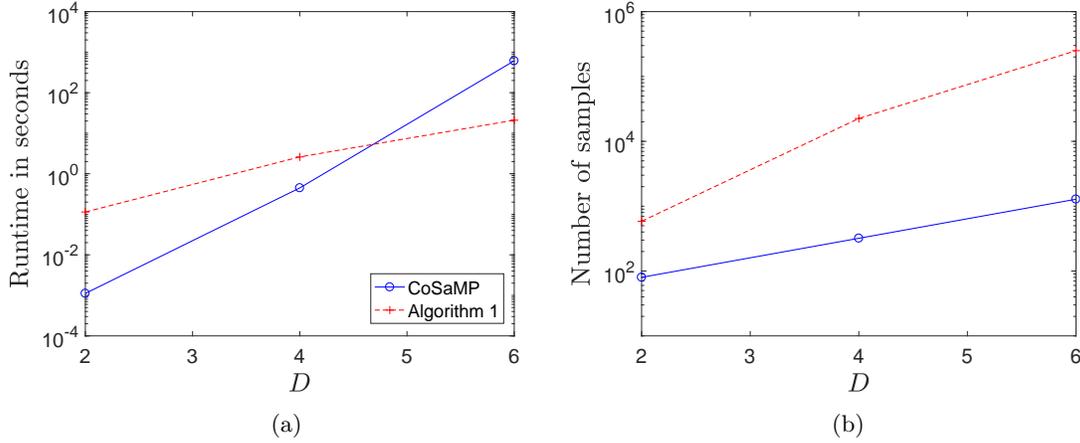
Figure 8: Chebyshev basis, $M = 20$, $D = \{2, 4, 6\}$, $s = 5$
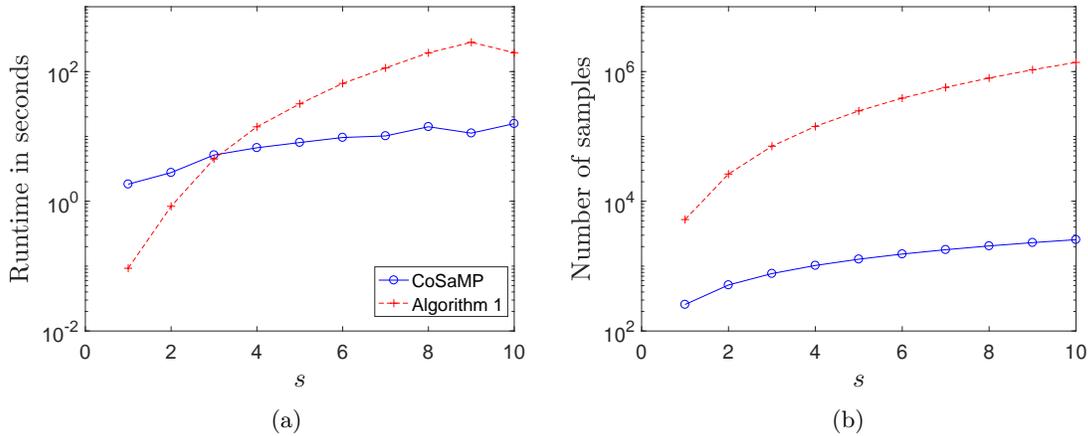
Figure 9: Chebyshev basis, $M = 10$, $D = 6$, $s = \{1, 2, 3, \cdots, 10\}$

respectively. In Figure 10, memory usage is also graphed for each M, D and s variation. Since this Chebyshev product basis has a BOS constant of $K = 2^{D/2}$, both CoSaMP and Algorithm 1 suffer from a mild exponential growth in sampling, runtime, and memory complexity as D increases (recall that $d = D$ for these experiments). This leads to markedly different overall performance for the Chebyshev basis than what is observed for the Fourier basis where $K = 1$. A reduction in performance from the Fourier case for both methods is clearly visible, e.g., in Figure 8. Nonetheless, Algorithm 1 demonstrates

49
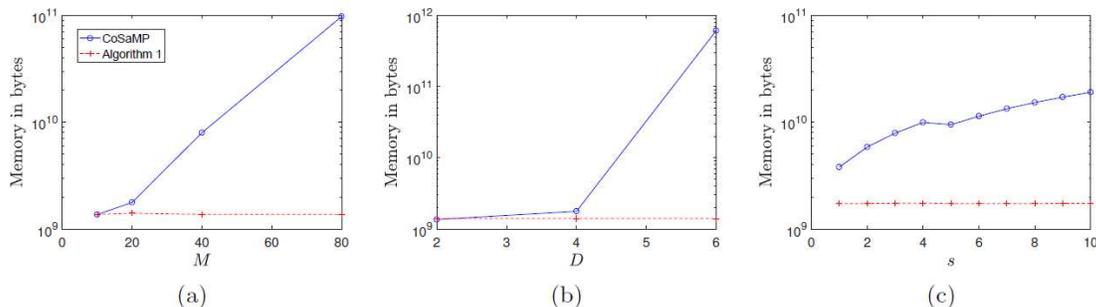
Figure 10: Chebyshev basis, memory usage when varying M, D, and s

the expected reduced runtime and sampling complexity dependence on M and D over CoSaMP in Figures 7 and 8, as well as a striking reduction in its required memory usage over CoSaMP in Figure 10 even when its runtime complexity is worse in Figure 9a. Unfortunately, the $\tilde{\mathcal{O}}(s^5)$ runtime dependance of Algorithm 3 on sparsity is again clear in Figure 9a leading to a crossover point of Algorithm 1 with CoSaMP at only $s = 3$ when $M = 10$ and $D = 6$. This again clearly marks the pairing process of Algorithm 3 as being in need of improvement.

5.4 Experiments for Larger Ranges of Sparsity s and Dimension D

Figures 11 and 12 explore the performance of Algorithm 1 on Fourier sparse functions for larger ranges of D and s, respectively. In Figure 11, a function of $D = 75$ variables can be recovered in just a few seconds when it is sufficiently sparse in the Fourier basis. It is worth pointing out here that when $D = 75$ the BOS in question contains $20^{75} \sim 10^{97}$ basis functions, significantly more than the approximately 10^{82} atoms estimated to be in the observable universe. We would like to emphasize that Algorithm 1 is solving problems in this setting that are simply too large to be solved efficiently, if at all, using standard superlinear-time compressive sensing approaches due to their memory requirements when dealing with such extremely large bases. Figure 12 also shows that functions with larger Fourier sparsities, s, than previously considered (up to $s = 160$) can be be recovered in about an hour or less from a BOS of size $40^5 = 102,400,000$.

In Figures 13 and 14 we consider the functions which are sparse in the Chebyshev product basis. Again, due to the larger BOS constant of the Chebyshev basis, the D and s ranges that our method can deal efficiently are smaller than in the Fourier case. When D is 12 or s is 20 in Figures 13 and 14, respectively, for example, it takes a few hours for Algorithm 1 to finish running. We again remind the readers that standard superlinear-
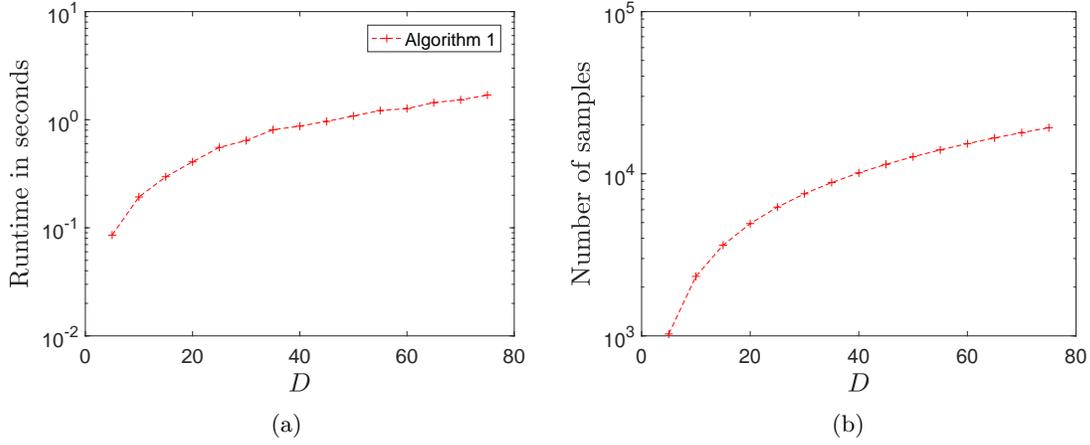
Figure 11: Fourier basis, $M = 20$, $D \in \{5, 10, 15, 20, \cdots, 75\}$, $s = 5$
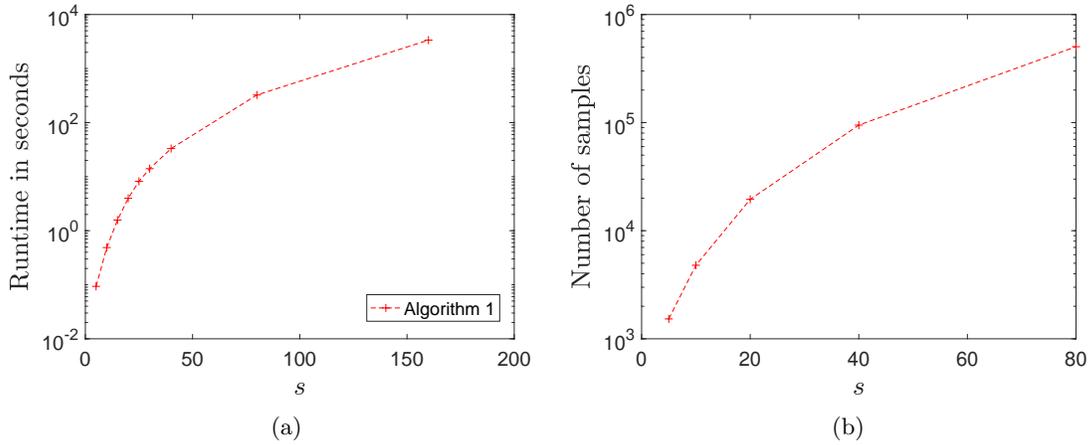
Figure 12: Fourier basis, $M = 40$, $D = 5$, $s \in \{5, 10, 20, 40, 80\}$

time compressive sensing methods cannot solve with such high dimensional problems at all, however, on anything less than a world class supercomputer due to their memory requirements. In the Figure 13 experiments the Chebyshev BOS contains $20^{12} \sim 10^{15}$ basis functions when $D = 12$. In the Figure 14 experiments the BOS contains just over 100 million basis functions.
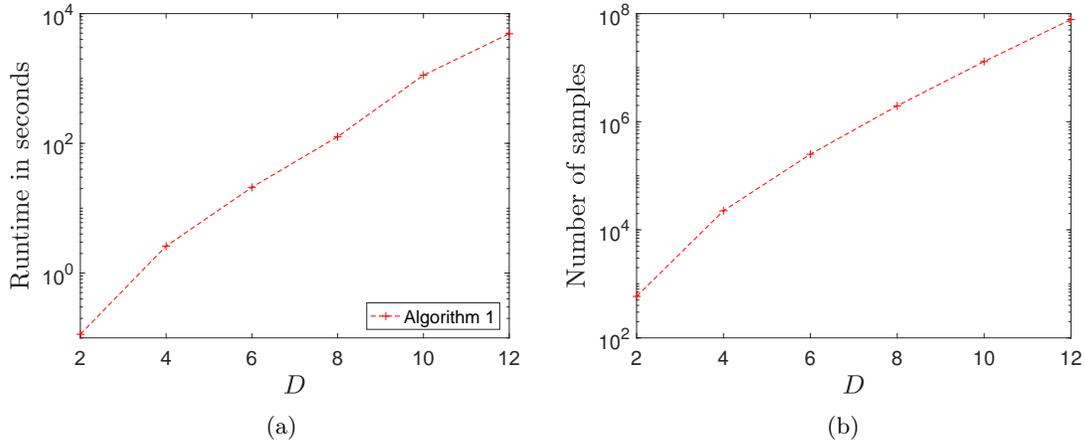
Figure 13: Chebyshev basis, $M = 20$, $D \in \{2, 4, 6, \cdots, 12\}$, $s = 5$
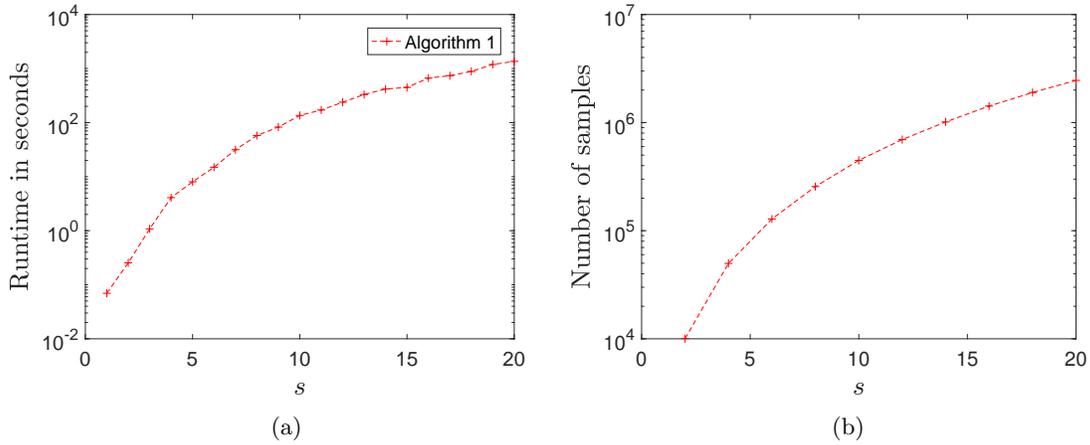
Figure 14: Chebyshev basis, $M = 40$, $D = 5$, $s \in \{2, 4, 6, \cdots, 20\}$

5.5 Recovery of Functions from Noisy Measurements

In Figures 15 and 16 we further consider exactly sparse trial functions (5.1) whose function evaluations are contaminated with Gaussian noise. That is, we provide Algorithm 1 with noisy samples

$$\boldsymbol{y}' \,=\, \boldsymbol{y} + \boldsymbol{g}' \,=\, \boldsymbol{y} + \sigma \frac{\|\boldsymbol{y}\|^2}{\|\boldsymbol{g}\|_2} \boldsymbol{g}$$

52
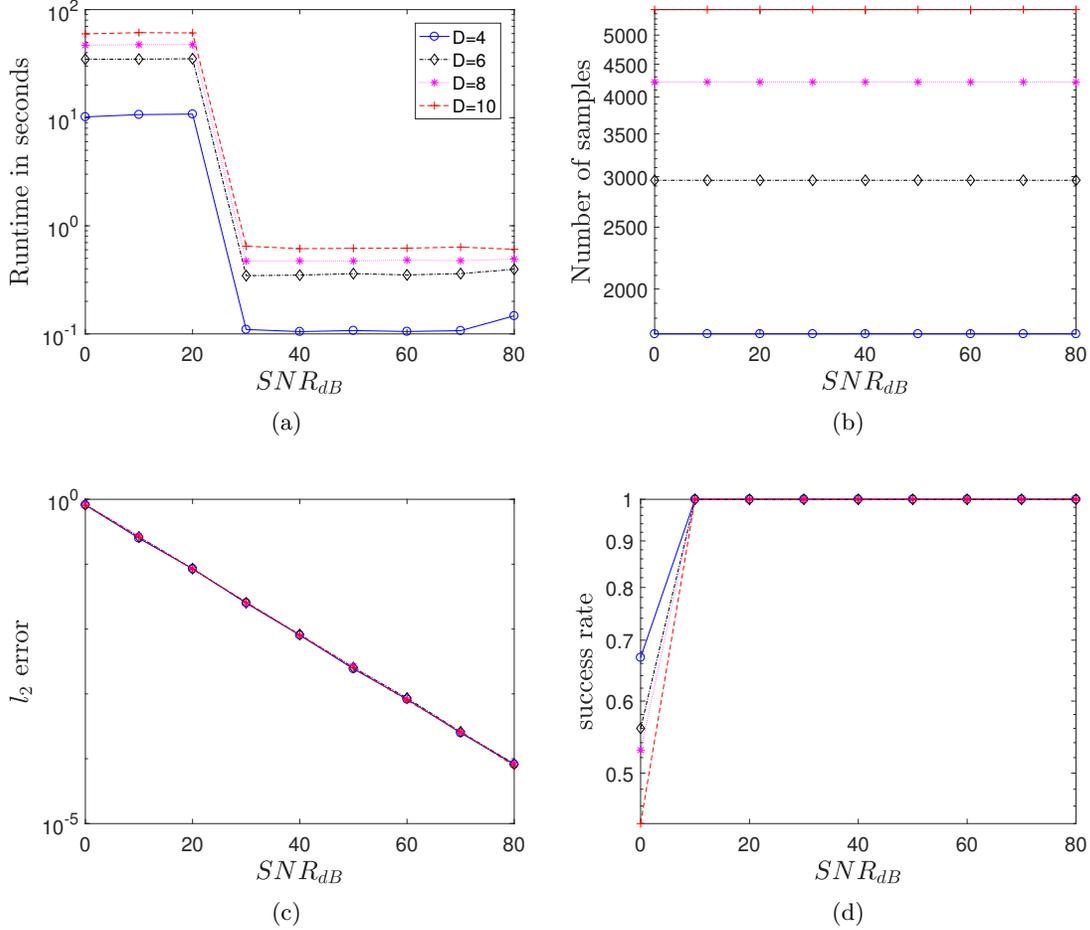
Figure 15: Algorithm 1, Fourier basis, $M = 10$, $D \in \{4, 6, 8, 10\}$, $s = 5$, $\mathrm{SNR}_{\mathrm{dB}} \in \{0, 10, 20, \cdots, 80\}$

where $\boldsymbol{y}$ contains noiseless samples from each f as per (1.5), $\boldsymbol{g} \sim \mathcal{N}(\boldsymbol{0}, I)$, and $\sigma \in \mathbb{R}^+$ is used to control the Signal to Noise Ratio (SNR) defined herein by

$$\mathrm{SNR}_{\mathrm{dB}} := 10 \log_{10} \left(\frac{\|\boldsymbol{y}\|_2^2}{\|\boldsymbol{g}'\|_2^2} \right) = -10 \log_{10} \left(\sigma^2 \right).$$

Figures 15 and 16 show the performance of Algorithm 1 for the Fourier and Chebyshev product bases, respectively, as SNR varies. Figure 15a shows the average runtime for each $D \in \{2, 4, 6, 8\}$ as $\mathrm{SNR}_{\mathrm{dB}}$ changes. When $\mathrm{SNR}_{\mathrm{dB}}$ is close to 0 (which means that
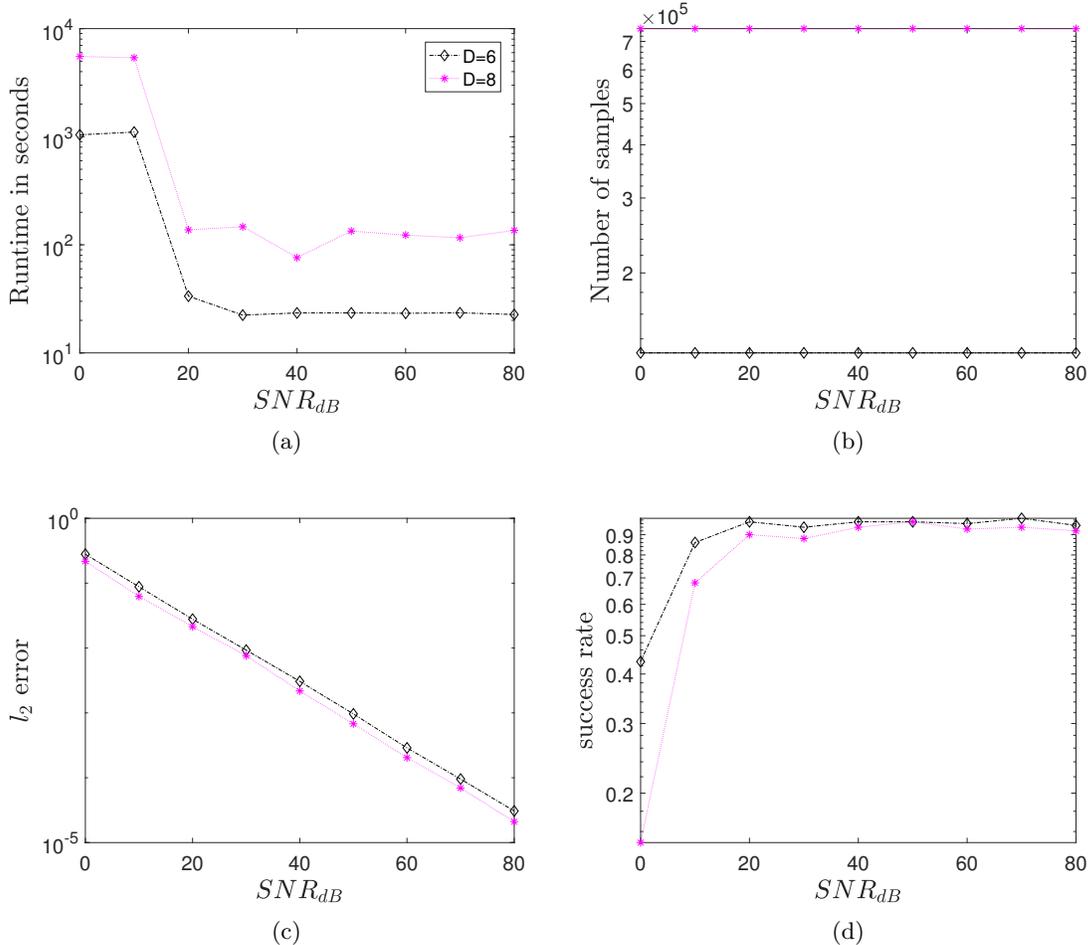
Figure 16: Algorithm 1, Chebyshev basis, $M = 10$, $D \in \{6, 8\}$, $s = 5$, SNR$_{\mathrm{dB}} \in \{0, 10, 20, \cdots, 80\}$

the ℓ_2-norm of noise vector is the same as the ℓ_2-norm of sample vector), the runtime gets larger due to Algorithm 1 using a larger number of overall iterations. The runtime also increases mildly as D increases in line with our previous observations. The sampling number in Figure 15b is set to be three times larger than the sampling number used in the noiseless cases. Figure 16 shows the results of Algorithm 1 applied to functions which are sparse in the Chebyshev product basis. Similar to the Fourier case, the runtime grows as the noise level gets worse in Figure 16a. Also, larger D results in the larger runtime as previously discussed. In Figure 16b, the sampling number is also set by tripling the sampling number used in noiseless cases.

As above, in both Figures 15 and 16 the average ℓ_2-error is computed by only considering the successful trials where every element of f's support, $\mathcal{S}$, is found. Here, however, the percentage of successful trials falls below 90% for lower SNR values. The success rates (i.e., the percentage of successful trials at each data point) are therefore plotted in Figures 15d and 16d. Both figures show that a smaller SNR_{dB} results in a smaller success rate, as one might expect. As SNR_{dB} increases, however, the ℓ_2-error decreases linearly for the successful trials.

5.6 Some Additional Implementational Details

In the line 13 of Algorithm 1 solving the least square problem can be accelerated by the iterative algorithms such as the Richardson method or the conjugate gradient method when the size of the matrix Φ_T is large [50]. For our range of relatively low sparsities, however, there was not much difference in the runtime between using such iterative least square solving algorithms and simply multiplying $\boldsymbol{y}^{\mathbf{E}}$ by the Moore-Penrose inverse, $\Phi_T^\dagger :=$ $(\Phi_T^* \Phi_T)^{-1} \Phi_T^*$. Thus, we simply form and use the Moore-Penrose inverse for both CoSaMP and Algorithm 1 in our implementations below.

Similarly, in our CoSaMP implementation the conjugate transpose of the measurement matrix, Φ, of size $m \times M^D$ is simply directly multiplied by the updated sample vector $\boldsymbol{v}$ in each iteration in order to obtain the signal proxy used for CoSaMP's support identification procedure (recall that $d = D$ in all experiments below so that $\mathcal{I} = [M]^D$). It is important to note that this matrix-vector multiplication can generally be done more efficiently if, e.g., one instead uses nonuniform FFT techniques [31] to evaluate $\Phi^* \boldsymbol{y}$ for the types of high-dimensional Fourier and Chebyshev basis functions considered below. However, such techniques are again not actually faster than a naive direct matrix multiply for the ranges of relatively low sparsities we consider in the experiments herein.[14] Furthermore, such nonuniform FFT techniques will still exhibit exponential runtime and memory dependence on D in the high-dimensional setting even for larger sparsity levels. Thus, nonuniform FFTs were not utilized in our MATLAB implementation of CoSaMP.

Again, we remind the reader that all the MATLAB codes used to produce the plots above is publicly available [11]. We invite the interested reader to download it and reproduce the plots herein at their leisure.

---

[14] CoSaMP always uses only $m = \mathcal{O}(s \cdot D \log M)$ samples in the experiments herein which means that its measurement matrix's conjugate transpose, $\Phi^* \in \mathbb{C}^{M^D \times m}$, can be naively multiplied by vectors in only $\mathcal{O}(s \cdot D \log M \cdot M^D)$-time. When s is small this is comparable to the $\mathcal{O}(D \log M \cdot M^D)$ runtime complexity of a (nonuniform) FFT.

6 Future Work

In this paper we develop a sublinear-time compressive sensing algorithm for rapidly learning functions of many variables that admit sparse representations in arbitrary Bounded Orthonormal Product (BOP) bases. Our results are universal in the sense that we give randomized constructions for highly structured grids which are proven to allow for the swift recovery *of all* functions which are sufficiently sparse in a given BOP basis, with high probability. This is the first method of its kind for general BOP bases. As a result, there is much work to do before these preliminary results reach their full potential.

First, and perhaps most obviously, the theoretical guarantees developed herein only apply to exactly BOPB-sparse functions despite the fact that our numerical experiments suggest that the algorithm also works for nearly BOPB-sparse functions. As a result, it should be possible to extend the main theorem herein to obtain best s-term approximation guarantees in the sense of Cohen et al. [16] for more arbitrary functions while simultaneously improving its complexity bounds (see, e.g., [14] for preliminary results in this direction). Specific complexity improvements that should be considered include an attempt at reducing the current cubic-in-s sampling complexity of our main theoretical result. This necessitates that a better pairing method be developed in Section 3 that requires fewer function evaluations.

Additionally, different dimension matching techniques for improving the pairing and entry identification steps of our proposed support identification method could be considered. Both steps are currently dimension incremental in the sense that all entries of the energetic index vectors are found one dimension at a time, and then extended into longer prefixes one dimension at a time. However, there is nothing stopping either of these steps from being generalized so that several short (potentially overlapping) prefixes of energetic index vectors are found in parallel and then merged/combined in a different order. This process could even be made adaptive to help eliminate interference between different index vector prefixes during a modified pairing phase's energy estimations. For example, any time a set of energetic prefixes differs from all the others under consideration in its jth entry one could compute an inner product in the jth dimension of h in a fashion similar to our current entry identification step in order to better isolate those prefixes' energy estimates from the others. Such methods could potentially lead to more accurate pairing results in noisy conditions.

Finally, there is also a good deal of improvement possible in the numerical implementation of the methods developed herein. In particular, Algorithm 1 was implemented in a generic fashion in our Section 5 experiments. For better results the implementation should be tuned to the particular BOPB being considered. It is also important to note that Algorithm 1 is inherently embarrassingly parallel in nature. In particular, all of the

energetic index vector entry sets $\mathcal{N}_j$ can be computed in parallel. Similarly, during the pairing step of support identification several prefixes can be grown simultaneously from, e.g., both the front and end of the index vector until the energetic prefixes and suffixes meet. Upon meeting in the middle, one additional energy estimate could then be done to correctly pair the proper prefixes and suffixes together.

7 Acknowledgements

References

[1] B. Adcock. Infinite-dimensional ℓ^1 minimization and function approximation from pointwise data. *Constructive Approximation*, 45(3):345–390, 2017.

[2] B. Adcock, S. Brugiapaglia, and C. G. Webster. *Compressed Sensing Approaches for Polynomial Approximation of High-Dimensional Functions*, pages 93–124. Springer International Publishing, Cham, 2017. ISBN 978-3-319-69802-1. doi: 10.1007/978-3-319-69802-1_3. URL https://doi.org/10.1007/978-3-319-69802-1_3.

[3] R. Arratia and L. Gordon. Tutorial on large deviations for the binomial distribution. *Bulletin of mathematical biology*, 51(1):125–131, 1989.

[4] J. Bailey, M. A. Iwen, and C. V. Spencer. On the design of deterministic matrices for fast recovery of Fourier compressible functions. *SIAM Journal on Matrix Analysis and Applications*, 33(1):263–289, 2012.

[5] S. Bittens, R. Zhang, and M. A. Iwen. A deterministic sparse FFT for functions with structured Fourier sparsity. *Advances in Computational Mathematics*, to appear.

[6] J.-L. Bouchot, H. Rauhut, and C. Schwab. Multi-level Compressed Sensing Petrov-Galerkin discretization of high-dimensional parametric PDEs. *ArXiv e-prints*, Jan. 2017.

[7] H.-J. Bungartz and M. Griebel. Sparse grids. *Acta numerica*, 13:147–269, 2004.

[8] R. E. Caflisch. Monte Carlo and quasi-Monte Carlo methods. *Acta numerica*, 7:1–49, 1998.

[9] E. J. Candeès, J. Romberg, and T. Tao. Stable signal recovery from incomplete and inaccurate measurements. *Commun. Pur. Appl. Math.*, 59:1207–1223, 2006.

[10] A. Chkifa, N. Dexter, H. Tran, and C. G. Webster. Polynomial approximation via compressed sensing of high-dimensional functions on lower sets. *arXiv preprint arXiv:1602.05823*, 2016.

[11] B. Choi and M. Iwen. SHT: Sparse harmonic transforms for learning functions of many variables. `https://math.msu.edu/~markiwen/Code.html`, Aug. 2018.

[12] B. Choi, A. Christlieb, and Y. Wang. Multi-dimensional sublinear sparse Fourier algorithm. *arXiv preprint arXiv:1606.07407*, 2016.

[13] B. Choi, A. Christlieb, and Y. Wang. Multiscale high-dimensional sparse Fourier algorithms for noisy data. *arXiv preprint arXiv:1907.03692*, 2019.

[14] B. Choi, M. Iwen, and T. Volkmer. Sparse harmonic transforms ii: Best *s*-term approximation guarantees for bounded orthonormal product bases in sublinear-time. *arXiv:1909.09564*, 2019.

[15] A. Christlieb, D. Lawlor, and Y. Wang. A multiscale sub-linear time Fourier algorithm for noisy data. *Applied and Computational Harmonic Analysis*, 40:553 – 574, 2016.

[16] A. Cohen, W. Dahmen, and R. DeVore. Compressed sensing and best *k*-term approximation. *Journal of the American mathematical society*, 22(1):211–231, 2009.

[17] G. Dahlquist and Å. Björck. *Numerical Methods in Scientific Computing: Volume 1*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2008. ISBN 0898716446, 9780898716443.

[18] I. Daubechies, M. Defrise, and C. De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on pure and applied mathematics*, 57(11):1413–1457, 2004.

[19] R. DeVore, G. Petrova, and P. Wojtaszczyk. Approximation of functions of few variables in high dimensions. *Constructive Approximation*, 33(1):125–143, 2011.

[20] D. L. Donoho. Compressed sensing. *IEEE Transactions on information theory*, 52 (4):1289–1306, 2006.

[21] D. Dũng, V. N. Temlyakov, and T. Ullrich. Hyperbolic cross approximation. *arXiv preprint arXiv:1601.03978*, 2016.

[22] S. Foucart. Hard thresholding pursuit: an algorithm for compressive sensing. *SIAM Journal on Numerical Analysis*, 49(6):2543–2563, 2011.

[23] S. Foucart and H. Rauhut. *A mathematical introduction to compressive sensing*. Springer, 2013.

[24] A. Gilbert, M. Strauss, J. Tropp, and R. Vershynin. Sublinear approximation of compressible signals. *Proc. SPIE Intell. Integrated Microsystems (IIM)*, page 623, 2006.

[25] A. Gilbert, Y. Li, E. Porat, and M. Strauss. Approximate sparse recovery: Optimizing time and measurements. *SIAM Journal on Computing*, 41(2):436–453, 2012. doi: 10.1137/100816705. URL https://doi.org/10.1137/100816705.

[26] A. C. Gilbert, S. Muthukrishnan, and M. Strauss. Improved time bounds for near-optimal sparse Fourier representations. In *Proceedings of SPIE*, volume 5914, page 59141A, 2005.

[27] A. C. Gilbert, M. J. Strauss, J. A. Tropp, and R. Vershynin. One sketch for all: Fast algorithms for compressed sensing. In *Proceedings of the Thirty-ninth Annual ACM Symposium on Theory of Computing*, STOC '07, pages 237–246, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-631-8. doi: 10.1145/1250790.1250824. URL http://doi.acm.org/10.1145/1250790.1250824.

[28] A. C. Gilbert, M. A. Iwen, and M. J. Strauss. Group testing and sparse signal recovery. In *42nd Asilomar Conference on Signals, Systems, and Computers*, 2008.

[29] A. C. Gilbert, P. Indyk, M. Iwen, and L. Schmidt. Recent developments in the sparse Fourier transform: a compressed Fourier transform for big data. *IEEE Signal Processing Magazine*, 31(5):91–100, 2014.

[30] A. C. Gilbert, Y. Li, E. Porat, and M. J. Strauss. For-all sparse recovery in near-optimal time. *ACM Trans. Algorithms*, 13(3):32:1–32:26, Mar. 2017. ISSN 1549-6325. doi: 10.1145/3039872. URL http://doi.acm.org/10.1145/3039872.

[31] L. Greengard and J.-Y. Lee. Accelerating the nonuniform fast Fourier transform. *SIAM review*, 46(3):443–454, 2004.

[32] C. Gross, M. A. Iwen, L. Kämmerer, and T. Volkmer. A deterministic algorithm for constructing multiple rank-1 lattices of near-optimal size. *arXiv:2003.09753*, 2020.

[33] H. Hassanieh, P. Indyk, D. Katabi, and E. Price. Simple and practical algorithm for sparse Fourier transform. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*, pages 1183–1194. Society for Industrial and Applied Mathematics, 2012.

[34] A. Hinrichs, E. Novak, M. Ullrich, and H. Woźniakowski. The curse of dimensionality for numerical integration of smooth functions. *Mathematics of Computation*, 83(290): 2853–2863, 2014.

[35] X. Hu, M. Iwen, and H. Kim. Rapidly computing sparse Legendre expansions via sparse Fourier transforms. *Numerical Algorithms*, pages 1–31, 2015.

[36] P. Indyk and M. Kapralov. Sparse Fourier transform in any constant dimension with nearly-optimal sample complexity in sublinear time. 2014.

[37] M. Iwen, A. Gilbert, and M. Strauss. Empirical evaluation of a sub-linear time sparse DFT algorithm. *Communications in Mathematical Sciences*, 5(4):981–998, 2007.

[38] M. A. Iwen. A deterministic sub-linear time sparse Fourier algorithm via non-adaptive compressed sensing methods. In *Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 20–29. Society for Industrial and Applied Mathematics, 2008.

[39] M. A. Iwen. Combinatorial sublinear-time Fourier algorithms. *Foundations of Computational Mathematics*, 10(3):303–338, 2010.

[40] M. A. Iwen. Improved approximation guarantees for sublinear-time Fourier algorithms. *Applied And Computational Harmonic Analysis*, 34(1):57–82, 2013.

[41] M. A. Iwen. Compressed sensing with sparse binary matrices: Instance optimal error guarantees in near-optimal time. *Journal of Complexity*, 30(1):1–15, 2014.

[42] L. Kämmerer, D. Potts, and T. Volkmer. High-dimensional sparse FFT based on sampling along multiple rank-1 lattices. *arXiv preprint arXiv:1711.05152*, 2017.

[43] M. Kapralov. Sparse fourier transform in any constant dimension with nearly-optimal sample complexity in sublinear time. *arXiv 1604.00845*, 2016.

[44] F. Krahmer and R. Ward. Stable and robust sampling strategies for compressive imaging. *IEEE Trans. Image Proc.*, 23(2):612–622, 2014. doi: 10.1109/TIP.2013. 2288004.

[45] F. Y. Kuo, G. Migliorati, F. Nobile, and D. Nuyens. Function integration, reconstruction and approximation using rank-1 lattices. *arXiv:1908.01178*, 2019.

[46] G. Leobacher and F. Pillichshammer. *Introduction to Quasi-Monte Carlo Integration and Applications*. Compact Textbooks in Mathematics. Springer International Publishing, 2014. ISBN 9783319034256. URL `https://books.google.com/books?id=OM-EBAAAQBAJ`.

[47] S. Merhi, R. Zhang, M. A. Iwen, and A. Christlieb. A new class of fully discrete sparse Fourier transforms: Faster stable implementations with guaran-

tees. *Journal of Fourier Analysis and Applications*, `https://doi.org/10.1007/s00041-018-9616-4`, 2018.

[48] L. Morotti. Explicit universal sampling sets in finite vector spaces. *Applied and Computational Harmonic Analysis*, 43(2):354–369, 2017.

[49] R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.

[50] D. Needell and J. A. Tropp. CoSaMP: Iterative signal recovery from incomplete and inaccurate samples. *Applied and Computational Harmonic Analysis*, 26(3):301–321, 2009.

[51] D. Needell and R. Vershynin. Signal recovery from incomplete and inaccurate measurements via regularized orthogonal matching pursuit. *IEEE Journal of selected topics in signal processing*, 4(2):310–316, 2010.

[52] D. Potts and T. Volkmer. Sparse high-dimensional FFT based on rank-1 lattice sampling. *Applied and Computational Harmonic Analysis*, 41(3):713–748, 2016.

[53] D. Potts and T. Volkmer. Multivariate sparse FFT based on rank-1 Chebyshev lattice sampling. In *Sampling Theory and Applications (SampTA), 2017 International Conference on*, pages 504–508. IEEE, 2017.

[54] H. Rauhut. Random sampling of sparse trigonometric polynomials. *Appl. Comp. Harmon. Anal.*, 22(1):16–42, 2007. ISSN 1063-5203.

[55] H. Rauhut and C. Schwab. Compressive sensing Petrov-Galerkin approximation of high-dimensional parametric operator equations. *Mathematics of Computation*, 86 (304):661–700, 2017.

[56] H. Rauhut and R. Ward. Sparse Legendre expansions via ℓ_1-minimization. *J. Approx. Theory*, 164(5):517–533, 2012.

[57] C. Schwab and R. A. Todor. Karhunen–Loève approximation of random fields by generalized fast multipole methods. *Journal of Computational Physics*, 217(1):100–122, 2006.

[58] I. Segal and M. Iwen. Improved sparse Fourier approximation results: Faster implementations and stronger guarantees. *Numerical Algorithms*, 63:239 – 263, 2013.

[59] J. Shen and L.-L. Wang. Sparse spectral approximations of high-dimensional problems based on hyperbolic cross. *SIAM Journal on Numerical Analysis*, 48(3):1087–1109, 2010.

[60] R. C. Smith. *Uncertainty Quantification: Theory, Implementation, and Applications.*

Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2013. ISBN 161197321X, 9781611973211.

[61] R. Vershynin. Introduction to the non-asymptotic analysis of random matrices. *arXiv:1011.3027v7*, 2011.

[62] T. Volkmer. Multivariate approximation and high-dimensional sparse FFT based on rank-1 lattice sampling. *Dissertation (PhD thesis), Faculty of Mathematics, Technische Universität Chemnitz (Chemnitz University of Technology)*, 2017.

[63] D. Xiu. *Numerical Methods for Stochastic Computations: A Spectral Method Approach*. Princeton University Press, Princeton, NJ, USA, 2010. ISBN 0691142122, 9780691142128.

[64] T. Zhang. Sparse recovery with orthogonal matching pursuit under RIP. *IEEE Transactions on Information Theory*, 57(9):6215–6221, 2011.

[65] R. Zippel. Probabilistic algorithms for sparse polynomials. In *International symposium on symbolic and algebraic manipulation*, pages 216–226. Springer, 1979.