Journal of Complexity 57 (2020) 101443

Contents lists available at ScienceDirect

Journal of Complexity

journal homepage: www.elsevier.com/locate/jco

Information based complexity for high dimensional sparse functions☆,☆☆

Cuize Han, Ming Yuan*

Department of Statistics, Columbia University, 1255 Amsterdam Avenue, New York, NY 10027, United States of America

ARTICLE INFO

Article history:
Received 22 April 2019
Received in revised form 25 October 2019
Accepted 27 October 2019
Available online 2 November 2019

Keywords:
Curse of dimensionality
Information based complexity
Randomized algorithms
Smoothness
Sparsity

ABSTRACT

We investigate optimal algorithms for optimizing and approximating a general high dimensional smooth and sparse function from the perspective of *information based complexity*. Our algorithms and analyses reveal several interesting characteristics for these tasks. In particular, somewhat surprisingly, we show that the optimal sample complexity for optimization or high precision approximation is independent of the ambient dimension. In addition, we show that the benefit of randomization could be substantial for these problems.

1. Introduction

High dimensional functions are ubiquitous in modern scientific and engineering applications. Dealing with them in general is intractable due to the so-called "curse-of-dimensionality". But in practical settings, the underlying function may have additional structures which, if appropriately accounted for, could help lift this barrier and allow for efficient algorithms to handle it. A canonical example of such structures is sparsity where a d-variate function f can be well-described by a much smaller number, say s ($\ll d$), albeit unknown, of variables. The special case of recovering f when it is further assumed to be linear in terms of either the input variables or a suitably chosen basis is commonly referred to as compressive sensing (see, e.g., [3,6]). Clearly the linearity assumption can be too restrictive in many applications; and there has been a lot of recent interest in broader and

☆ Research supported in part by NSF Grant DMS-1803450.
☆☆ Communicated by Erich Novak.
* Corresponding author.
E-mail address: ming.yuan@columbia.edu (M. Yuan).

https://doi.org/10.1016/j.jco.2019.101443
0885-064X/

more flexible classes of high dimensional functions. See, e.g., [17–19] for a comprehensive survey of recent progresses.

In particular, the present work was inspired by [5] and [29] who studied effective approaches for approximating a high dimensional smooth function that depends on few relevant variables. To fix ideas, we shall focus on functions that reside in Hölder space $\mathcal{H}^\alpha([0,1]^d)$ with $\alpha > 0$. Write $\alpha = \alpha_0 + \alpha_1$ where α_0 is an integer and $0 < \alpha_1 \leq 1$. Then $\mathcal{H}^\alpha([0,1]^d)$ is defined as

$$\mathcal{H}^\alpha([0,1]^d) = \{f : [0,1]^d \to \mathbb{R} \parallel D^{(l)}f(x) - D^{(l)}f(y)| \leq \max_{1\leq j\leq d} |x_i - y_i|^{\alpha_1}, \\ \forall x, y \in [0,1]^d, \quad l_1 + \cdots + l_d = \alpha_0\}$$

where $l = (l_1, \ldots, l_d) \in \mathbb{N}^d$ is a multi-index, and

$$D^{(l)}f = \frac{\partial^{l_1+\cdots+l_d} f}{\partial x_1^{l_1} \cdots \partial x_d^{l_d}}.$$

We are especially interested in functions from $\mathcal{H}^\alpha$ that are also sparse. We say a function f is supported on $S \subseteq [d]$ if and only if there exists a function $g : [0,1]^{|S|} \to \mathbb{R}$ such that

$$f(x) = g(x_S), \qquad \forall x \in [0,1]^d,$$

where $[d] = \{1, \ldots, d\}$, $|\cdot|$ stands for the cardinality of a set, and $x_S = (x_i)_{i\in S}$. Denote by $\mathrm{supp}(f)$ the smallest set on which f is supported. The *sparsity* of f can then be measured by $|\mathrm{supp}(f)|$. We are interested in the case when d is large yet $|\mathrm{supp}(f)|$ is small. More specifically, we assume that f comes from

$$\mathcal{F}_{\alpha,d,s} = \{f \in \mathcal{H}^\alpha([0,1]^d) : |\mathrm{supp}(f)| \leq s\}.$$

Our main goal is to characterize the optimal sample complexity for optimizing and approximating functions from $\mathcal{F}_{\alpha,d,s}$. In fact, we shall focus on the dependence of the complexity of optimizing and approximating f on d with both α and s fixed and known a priori. It is worth noting that assuming f has a bounded support, in a certain sense, is necessary because even if $\mathrm{supp}(f)$ is known a priori, recovering g would suffer from the curse of dimensionality in that the sample complexity increases exponentially with $|\mathrm{supp}(f)|$ and therefore even for moderate s, the task quickly becomes intractable for practical purposes.

1.1. Information based complexity

In particular, we shall take the approach of information based complexity. The general framework of information based complexity was formalized in early 1980s by [25,26,30] among others. Earlier developments in the area focused on the dependence of the complexities on the accuracy with the dimensionality considered fixed. Since the pioneering work of [31], more emphasis has been put on high dimensional problems and characterizing the role of d. See [32] for a review of history of the field and the three volume monograph by [17–19] and the references therein for recent progresses. Existing studies of information based complexity for high dimensional problems are often in the framework of the so-called weighted spaces, first introduced by [21]. Our work here complements them by examining the effect of sparsity in high dimensional problems.

Suppose that an element $f \in \mathcal{F}_{\alpha,d,s}$ is not known but we can make point queries to an oracle that takes an $x \in [0,1]^d$ as input and returns the function value $f(x)$. The information about f we gather from n point queries can then be represented by

$$I_n^{\mathrm{ran}} = \{N : \mathcal{F}_{\alpha,d,s} \mapsto \mathbb{R}^n | N(f) = (f(z_1), f(z_2(f(z_1))), \ldots, f(z_n(f(z_1), \ldots, f(z_{n-1})))), \\ z_1 \text{ is a random variable in } [0,1]^d, \\ z_i \text{ is a random function that maps from } \mathbb{R}^{i-1} \mapsto [0,1]^d, i = 2, \ldots, n\}.$$

Note that in general, the queries may depend on those made earlier and therefore are adaptive in nature. The superscript of I_n^{ran} signifies the fact that the query points z_is are allowed to be random.

When they follow degenerative distributions the query scheme becomes deterministic, and we shall denote the corresponding information set by

$$I_n^{\det} = \{N : \mathcal{F}_{\alpha,d,s} \mapsto \mathbb{R}^n | N(f) = (f(z_1), f(z_2(f(z_1))), \ldots, f(z_n(f(z_1), \ldots, f(z_{n-1})))),$$
$$z_1 \in [0,1]^d,\ z_i : \mathbb{R}^{i-1} \mapsto [0,1]^d, i = 2, \ldots, n\}.$$

An algorithm based on n point queries can be written as $S_n := \phi \circ N$ for some $N \in I_n^{\text{ran}}$ (resp. $I_n^{\det}$), and a function ϕ that maps from $\mathbb{R}^n$ to $\mathbb{R}$ for optimization, and from $\mathbb{R}^n$ to $C([0,1]^d)$ for approximation. The accuracy of an algorithm S_n at f can be measured by:

$$\Delta(S_n, f; \text{OPT}) := |S_n(f) - \min_x f(x)|$$

for optimization, and

$$\Delta(S_n, f; \text{APP}) := \|S_n(f) - f\|_{C([0,1]^d)}$$

for approximation. Note that there is no loss of generality to consider only finding the minimum of the function for the problem of optimization because the function class $\mathcal{F}_{\alpha,d,s}$ is symmetric, that is $f \in \mathcal{F}_{\alpha,d,s}$ if and only if $-f \in \mathcal{F}_{\alpha,d,s}$. Of special interest are the optimal algorithms that can achieve a given level of accuracy with the smallest number of point queries, or the so-called information based complexity. Denote by

$$n_\varepsilon^{\det}(\mathcal{F}_{\alpha,d,s}, \text{OPT}) = \min\{n \in \mathbb{N} : \exists S_n = \phi \circ N \text{ for some } \phi : \mathbb{R}^n \to \mathbb{R} \text{ and } N \in I_n^{\det}$$
$$\text{such that } \Delta(S_n, f; \text{OPT}) \le \varepsilon, \quad \forall f \in \mathcal{F}_{\alpha,d,s}\},$$

and

$$n_\varepsilon^{\det}(\mathcal{F}_{\alpha,d,s}, \text{APP}) = \min\{n \in \mathbb{N} : \exists S_n = \phi \circ N \text{ for some } \phi : \mathbb{R}^n \to C([0,1]^d) \text{ and } N \in I_n^{\det}$$
$$\text{such that } \Delta(S_n, f; \text{APP}) \le \varepsilon, \quad \forall f \in \mathcal{F}_{\alpha,d,s}\}.$$

Similarly, for randomized algorithms, we write for a given probability of tolerance $\delta \in (0,1)$,

$$n_\varepsilon^{\text{ran}}(\mathcal{F}_{\alpha,d,s}, \text{OPT}; \delta) = \min\{n \in \mathbb{N} : \exists S_n = \phi \circ N \text{ for some } \phi : \mathbb{R}^n \to \mathbb{R} \text{ and } N \in I_n^{\text{ran}}$$
$$\text{such that } \mathbb{P}\{\Delta(S_n, f; \text{OPT}) \le \varepsilon\} \ge 1 - \delta, \quad \forall f \in \mathcal{F}_{\alpha,d,s}\},$$

and

$$n_\varepsilon^{\text{ran}}(\mathcal{F}_{\alpha,d,s}, \text{APP}; \delta) = \min\{n \in \mathbb{N} : \exists S_n = \phi \circ N \text{ for some } \phi : \mathbb{R}^n \to C([0,1]^d) \text{ and } N \in I_n^{\text{ran}}$$
$$\text{such that } \mathbb{P}\{\Delta(S_n, f; \text{APP}) \le \varepsilon\} \ge 1 - \delta, \quad \forall f \in \mathcal{F}_{\alpha,d,s}\}.$$

The main purpose of the present article is to determine the asymptotic behavior of these quantities as the ambient dimension d increases and accuracy ε approaches zero, and contrast the two types of information.

In what follows, we shall consider an arbitrarily *fixed* $\delta \in (0,1)$, and write in what follows $a_{d,\varepsilon} = O(b_{d,\varepsilon})$ or $a_{d,\varepsilon} \lesssim b_{d,\varepsilon}$ if $a_{d,\varepsilon} \le c \cdot b_{d,\varepsilon}$ for some constant $c > 0$ that may depend on s, α or δ, but is independent of d and ε. Similarly we write $a_{d,\varepsilon} = \Omega(b_{d,\varepsilon})$ or $a_{d,\varepsilon} \gtrsim b_{d,\varepsilon}$ if $b_{d,\varepsilon} = O(a_{d,\varepsilon})$, and $a_{d,\varepsilon} \asymp b_{d,\varepsilon}$ if $a_{d,\varepsilon} = O(b_{d,\varepsilon})$ and $a_{d,\varepsilon} = \Omega(b_{d,\varepsilon})$. Oftentimes, the rate of convergence of these quantities remain the same for any fixed $\delta \in (0,1)$, we shall then omit the argument δ in $n_\varepsilon^{\text{ran}}$ for brevity.

1.2. Summary of results

In a pioneering work, [5] developed deterministic algorithms for approximating functions from $\mathcal{F}_{\alpha,d,s}$. Their results were further improved by [29]. Similar strategy has more recently been adopted for optimization by [4]. Their results, for example Corollary 4.3 of [29] and Theorem 4.2 of [4], immediately imply that

Theorem 1 ([4,29]). *For any $s \in \mathbb{N}$,*

$$n_\varepsilon^{\det}(\mathcal{F}_{\alpha,d,s}, \mathrm{OPT}) = O(\varepsilon^{-s/\alpha} \log d),$$

and

$$n_\varepsilon^{\det}(\mathcal{F}_{\alpha,d,s}, \mathrm{APP}) = O(\varepsilon^{-s/\alpha} \log d).$$

It is of great interest to investigate whether these algorithms are optimal. We show that, while they are generally suboptimal for optimization when $s = 1$, these algorithms are nearly if not optimal among all deterministic algorithms when there is more than one relevant variable ($s > 1$). More specifically, we prove that

Theorem 2. *For any $s \in \mathbb{N}$,*

$$n_\varepsilon^{\det}(\mathcal{F}_{\alpha,d,s}, \mathrm{OPT}) = \Omega(\varepsilon^{-s/\alpha} + \varepsilon^{-(s-1)/\alpha} \log d),$$

and

$$n_\varepsilon^{\det}(\mathcal{F}_{\alpha,d,s}, \mathrm{APP}) = \Omega(\varepsilon^{-s/\alpha} + \varepsilon^{-(s-1)/\alpha} \log d).$$

Moreover, if $s = 1$,

$$n_\varepsilon^{\det}(\mathcal{F}_{\alpha,d,s}, \mathrm{OPT}) \asymp \varepsilon^{-1/\alpha}.$$

We also study randomized algorithms and show that they allow for improved complexity bounds. In particular, we show that

Theorem 3. *For any fixed $\alpha > 0$, $s \in \mathbb{N}$ and $\delta \in (0, 1)$,*

$$n_\varepsilon^{\mathrm{ran}}(\mathcal{F}_{\alpha,d,s}, \mathrm{OPT}; \delta) \asymp \varepsilon^{-s/\alpha},$$

and

$$n_\varepsilon^{\mathrm{ran}}(\mathcal{F}_{\alpha,d,s}, \mathrm{APP}; \delta) = O\left(\varepsilon^{-s/\alpha} + \log d\right).$$

If, in addition, $\varepsilon = O((\log d)^{-\alpha/s})$, then we also have

$$n_\varepsilon^{\mathrm{ran}}(\mathcal{F}_{\alpha,d,s}, \mathrm{APP}; \delta) \asymp \varepsilon^{-s/\alpha}.$$

The fact that the ambient dimension d is absent from the rates for $n_\varepsilon^{\mathrm{ran}}(\mathcal{F}_{\alpha,d,s}, \mathrm{OPT})$ and $n_\varepsilon^{\mathrm{ran}}(\mathcal{F}_{\alpha,d,s}, \mathrm{APP})$ at least for sufficiently small ε suggests that, perhaps surprisingly, when optimizing or approximating a high dimensional yet sparse function, there is no dependence on the ambient dimension d!

Theorems 2 and 3 together indicate that randomization is beneficial when there are more than one relevant variables. This is to be contrasted with classes of smooth functions without sparsity constraints such as $\mathcal{H}^\alpha([0, 1]^d)$ for which it is well known that randomization only provides marginal if any improvement over the more restrictive deterministic schemes. See, e.g., Sections 1.2 and 2.2 of [16].

Furthermore, our findings and the algorithms we developed are generally applicable to classes of high dimensional functions with intrinsic sparsity, beyond $\mathcal{F}_{\alpha,d,s}$. For illustration, we shall also discuss the implications of our results on several commonly encountered examples including additive models, finite order functions, and functions with mixed derivatives. Our work contributes to a fast growing literature in diverse areas on effective treatment for high dimensional problems by pinpointing the fundamental role of sparsity in mitigating the curse of dimensionality.

The rest of the paper is organized as follows. In the next two sections, we shall investigate the optimal sample complexity for optimizing and approximating respectively functions from $\mathcal{F}_{\alpha,d,s}$. To demonstrate the generality of our treatment and how the techniques could be broadly applied to other classes of high dimensional and sparse functions, we discuss how our results could be extended to and algorithms adapted for several popular examples in Section 4. We conclude with a brief summary discussion in Section 5.

2. Optimal algorithms for optimization

We first consider optimizing a high dimensional sparse function. As noted before, we shall focus on finding the minimum without loss of generality.

Because of the smoothness of functions from $\mathcal{H}^\alpha([0,1]^d)$, we can restrict our attention to their values on a lattice $\mathscr{L}^d$ where $\mathscr{L} = \{0, 1/L, \ldots, 1-1/L, 1\}$. If $f|_{\mathscr{L}^d}$ is known, a piecewise polynomial interpolation $\tilde{f}$ can be constructed based on its restriction $f|_{\mathscr{L}^d}$ with error of the order $L^{-\alpha}$:

$$\sup_{f\in\mathcal{H}^\alpha([0,1]^d)} \|\tilde{f} - f\|_{C([0,1]^d)} = O(L^{-\alpha}).$$

See, e.g., [24]. The difference between $\min_{x\in[0,1]^d}\tilde{f}(x)$ and $\min_{x\in[0,1]^d} f(x)$ is therefore also of the order $O(L^{-\alpha})$. Taking $L \asymp \varepsilon^{-1/\alpha}$ yields an estimate of the optimum $\min_{x\in[0,1]^d} f(x)$ with accuracy ε. The sample complexity of such an algorithm is $(L+1)^d = O(\varepsilon^{-d/\alpha})$, which is also known to be optimal in that no other algorithms can achieve the same level of accuracy with a sample complexity $o(\varepsilon^{-d/\alpha})$. In other words,

$$n_\varepsilon(\mathcal{H}^\alpha([0,1]^d), \mathrm{OPT}) \asymp \varepsilon^{-d/\alpha}.$$

See, e.g., [11,16]. These algorithms, however, are suboptimal for functions from $\mathcal{F}_{\alpha,d,s}$ because they do not exploit the sparsity of these functions. In fact, because f is sparse, it is plausible that we could compute $\min_{x\in[0,1]^d} f(x)$ with the same accuracy without the need to make queries at every point on the lattice.

2.1. Deterministic algorithms

To gain insights into the complexity of optimization for functions from $\mathcal{F}_{\alpha,d,s}$, it is instructive to begin with the simple case when $s = 1$. In other words, there is only one relevant variable. In this case, it is not hard to see, for any $f \in \mathcal{F}_{\alpha,d,1}$,

$$\min_{x\in[0,1]} g(x) = \min_{x\in[0,1]^d} f(x)$$

where $g(x) = f(x\cdot\mathbf{1})$ and $\mathbf{1}$ is the vector of ones of conformable dimension. Therefore, it suffices to make point queries at $(i/L)\cdot\mathbf{1}$ for $0 \le i \le L$ and compute the minimum of a piecewise polynomial approximation to g. This immediately implies

$$n_\varepsilon^{\mathrm{det}}(\mathcal{F}_{\alpha,d,1}, \mathrm{OPT}) \le L + 1 \asymp \varepsilon^{-1/\alpha}.$$

On the other hand, if the relevant variable is known a priori, the problem reduces to minimizing a univariate function with Hölder smoothness so that

$$n_\varepsilon^{\mathrm{det}}(\mathcal{F}_{\alpha,d,1}, \mathrm{OPT}) \ge n_\varepsilon^{\mathrm{det}}(\mathcal{H}^\alpha([0,1]), \mathrm{OPT}).$$

It is well known that

$$n_\varepsilon^{\mathrm{det}}(\mathcal{H}^\alpha([0,1]), \mathrm{OPT}) \asymp \varepsilon^{-1/\alpha}.$$

See, e.g., Section 1.2 of [16]. In summary, we get

Proposition 1. *The information based complexity of optimization for $\mathcal{F}_{\alpha,d,1}$ satisfies*

$$n_\varepsilon^{\mathrm{det}}(\mathcal{F}_{\alpha,d,1}, \mathrm{OPT}) \asymp \varepsilon^{-1/\alpha}.$$

The most interesting observation from this simple exercise is the fact that the complexity of optimizing a d-variable and 1-sparse function is independent of the ambient dimension d, meaning that optimizing a high dimensional 1-sparse function is as difficult as optimizing a univariate function!

The general case, however, turns out to be more complicated. Recently, building upon earlier developments by [5], [29] and [4] developed a deterministic algorithm that computes the optimum

of any function $f \in \mathcal{F}_{\alpha,d,s}$ with accuracy ε from $O(\varepsilon^{-s/\alpha}\log d)$ point queries. Their result indicates that

$$n_\varepsilon^{\det}(\mathcal{F}_{\alpha,d,s}, \mathrm{OPT}) = O(\varepsilon^{-s/\alpha}\log d). \tag{1}$$

Naturally we are interested in whether or not we can devise algorithms with complexity independent of d, as in the case when $s = 1$. It turns out not to be the case for deterministic algorithms when $s > 1$.

Theorem 4. *For any fixed $s \geq 2$,*

$$n_\varepsilon^{\det}(\mathcal{F}_{\alpha,d,s}, \mathrm{OPT}) = \Omega(\varepsilon^{-s/\alpha} + \varepsilon^{-(s-1)/\alpha}\log d). \tag{2}$$

Proof. Note that

$$n_\varepsilon^{\det}(\mathcal{F}_{\alpha,d,s}, \mathrm{OPT}) \geq n_\varepsilon^{\det}(\mathcal{H}^\alpha([0,1]^s), \mathrm{OPT}) \asymp \varepsilon^{-s/\alpha}.$$

See, e.g., [16]. It suffices to show that

$$n_\varepsilon^{\det}(\mathcal{F}_{\alpha,d,s}, \mathrm{OPT}) = \Omega(\varepsilon^{-(s-1)/\alpha}\log d).$$

To this end, assume that there exists a deterministic algorithm, possibly adaptive, that can compute $\min_{x\in[0,1]^d} f(x)$ over all $f \in \mathcal{F}_{\alpha,d,s}$ with accuracy ε and n point queries. When $f = 0$, the algorithm queries at a sequence of points (possibly adaptively), denoted by $\mathscr{X} := \{z_1, z_2(0), \ldots, z_n(0,\ldots,0)\}$ or $\{z_1,\ldots,z_n\}$ for short, and the oracle returns values 0 for all queries. Therefore the algorithm produces an estimate of $\min_{x\in[0,1]^d} f(x)$ within $[-\varepsilon, +\varepsilon]$ whenever $f(z_i) = 0$ for $i = 1,\ldots,n$. This means that, for any function $f \in \mathcal{F}_{\alpha,d,s}$ such that $f(z_i) = 0$ for $i = 1,\ldots,n$, we must have $\min_{x\in[0,1]^d} f(x) \geq -2\varepsilon$.

Now write

$$\Phi(x) = \begin{cases} -a\cdot\prod_{i=1}^s (1-x_i^2)^{k+1} & x \in [-1,1]^s \\ 0 & \text{otherwise} \end{cases}$$

where the constant $a > 0$ is chosen so that $\Phi \in \mathcal{H}^\alpha([0,1]^s)$. Define

$$\Phi_\omega(x) = (2L)^{-\alpha}\cdot\Phi(2L(x-\omega)).$$

It is not hard to see that $\Phi_\omega \in \mathcal{H}^\alpha([0,1]^s)$, $\min_{x\in[0,1]^s}\Phi_\omega(x) = -a(2L)^{-\alpha}$, and Φ_ω vanishes outside

$$\mathcal{D}_\omega := \prod_{1\leq i\leq s} (\omega_i - 1/(2L), \omega_i + 1/(2L)).$$

Now for any $S \subset [d]$ such that $|S| = s$ and $\omega \in \{1/(2L), 3/(2L), \ldots, 1-1/(2L)\}^{|S|} =: \tilde{\mathscr{L}}^{|S|}$, denote by $f_{S,\omega}$ a function mapping from $[0,1]^d$ to $\mathbb{R}$ such that

$$f_{S,\omega}(x) = \Phi_\omega(x_S), \qquad \forall x \in [0,1]^d.$$

Note that by taking $L = c\varepsilon^{-1/\alpha}$ for a small enough constant $c > 0$, we can ensure that

$$\min_x f_{S,\omega}(x) = -a(2L)^{-\alpha} < -2\varepsilon.$$

The validity of the algorithm dictates that

$$\{z_{1,S},\ldots,z_{n,S}\} \cap \mathcal{D}_\omega \neq \emptyset, \tag{3}$$

for all $|S| \leq s$ and $\omega \in \tilde{\mathscr{L}}^{|S|}$. We now show this implies the desired claim.

Consider first the case when $s = 2$. Let $\mathcal{Q} : [0,1]^d \to \tilde{\mathscr{L}}^d$ be a map such that $x_k \in \mathcal{D}_{\mathcal{Q}(x)_k}$, where x_k and $\mathcal{Q}(x)_k$ are the kth coordinate of $x \in [0,1]^d$ and $\mathcal{Q}(x) \in \tilde{\mathscr{L}}^d$ respectively. Condition (3) shows that the set $\{\mathcal{Q}(z_1),\ldots,\mathcal{Q}(z_n)\}$ forms a covering array CAN$(n,2,d,L)$ of size n and strength 2. As shown by [9], we have $n = \Omega(L\log d)$. The claim then follows with the particular choice of L.

Now consider the case when $s > 2$. For any $\tilde{\omega} \in \tilde{\mathscr{L}}^{s-2}$, write

$$\tilde{\mathscr{X}}_{\tilde{\omega}} = \{x_{-[s-2]} : x \in \mathscr{X}, (x_1,\ldots,x_{s-2})^\top \in \mathcal{D}_{\tilde{\omega}}\}.$$

In light of (3), we know

$$\{x_S : x \in \tilde{\mathscr{Z}}_{\tilde{\omega}}\} \cap \mathcal{D}_\omega \neq \emptyset$$

for any $S \subset [d-s+2]$ such that $|S| = 2$ and $\omega \in \tilde{\mathscr{L}}^2$. By the argument for the case when $s = 2$, we get

$$|\tilde{\mathscr{Z}}_{\tilde{\omega}}| = \Omega(L \log d).$$

Hence

$$|\mathscr{Z}| \geq \sum_{\tilde{\omega} \in \tilde{\mathscr{L}}^{s-2}} |\tilde{\mathscr{Z}}_{\tilde{\omega}}| = \Omega(L^{s-1} \log d) = \Omega(\varepsilon^{-(s-1)/\alpha} \log d),$$

which concludes the proof. □

Note that the lower bound (2) does not match the upper bound (1). It remains unclear whether or not there exist deterministic algorithms with lower complexity than those developed by [4]. On the other hand, it is entirely plausible that the lower bound given by Theorem 4 can be further improved but it nonetheless shows that at least when d is large, or ε is large, the information based complexity of deterministic algorithms must depend on d. Interestingly, though, this is not the case when we consider more general randomized algorithms where we can show that the complexity of optimizing a function from $\mathcal{F}_{\alpha,d,s}$ is always $O(\varepsilon^{-s/\alpha})$.

2.2. Randomized algorithms

We note first that the lower bound for randomized algorithms follows easily: it is not hard to see that

$$n_\varepsilon^{\text{ran}}(\mathcal{F}_{\alpha,d,s}, \text{OPT}) \geq n_\varepsilon^{\text{ran}}(\mathcal{H}^\alpha([0,1]^s), \text{OPT}) \asymp \varepsilon^{-s/\alpha}, \tag{4}$$

See, e.g., Section 2.2 of [16] for further discussion on the complexity of optimization over classical Hölder class $\mathcal{H}^\alpha$. To attain the complexity bound on the rightmost hand side for $s \geq 2$, however, requires a more sophisticated algorithm similar in spirit to that of [4].

Consider a hashing function $h : [d] \to [s]$ that maps the d coordinates into s groups. Denote by $Q_h : [0,1]^s \to [0,1]^d$ a map such that the jth coordinate of $Q_h(z)$ is $z_{h(j)}$. It is not hard to see that $f \circ Q_h \in \mathcal{H}^\alpha([0,1]^s)$ for any hashing function h. Therefore, we can compute the minimum of $f \circ Q_h$ by minimizing a piecewise polynomial interpolation of queries on the s-dimensional lattice $\mathscr{L}^s$. In addition,

$$\min_{x \in [0,1]^d} f(x) = \min_{x \in [0,1]^s} f \circ Q_h(x)$$

whenever h *partitions* $\text{supp}(f)$:

$$|h(\text{supp}(f))| = |\text{supp}(f)|, \tag{5}$$

that is, each relevant variable of f is mapped to a different value. This means that we can achieve the sample complexity of $O(\varepsilon^{-s/\alpha})$ as long as h partitions $\text{supp}(f)$.

The challenge is that $\text{supp}(f)$ is not known a priori so that finding a hashing function that partitions $\text{supp}(f)$ is not trivial. A general strategy is to entertain a collection of hashing functions $\mathscr{H}$ so that there exists an element $h \in \mathscr{H}$ that obeys (5). If this is the case, then it is clear that

$$\begin{aligned} \left| \min_{h \in \mathscr{H}} m_h - \min_{x \in [0,1]^d} f(x) \right| &= \left| \min_{h \in \mathscr{H}} m_h - \min_{h \in \mathscr{H}} \min_{x \in [0,1]^s} f \circ Q_h(x) \right| \\ &\leq \max_{h \in \mathscr{H}} \left| m_h - \min_{x \in [0,1]^s} f \circ Q_h(x) \right| = O(L^{-\alpha}), \end{aligned}$$

where m_h is the minimum of $f \circ Q_h$ computed by minimizing a piecewise polynomial interpolation of queries on the s-dimensional lattice $\mathscr{L}^s$ as discussed before. The sample complexity for this approach is $(L+1)^s \cdot |\mathscr{H}| = O(\varepsilon^{-s/\alpha} \cdot |\mathscr{H}|)$.

A sufficient condition for (5) to hold for at least one element of $\mathscr{H}$ is that $\mathscr{H}$ forms a so-called *perfect hashing family*, meaning that for not only $\mathrm{supp}(f)$ but *any* subset S of $[d]$ of size s there exists an $h \in \mathscr{H}$ such that $|h(S)| = s$. It is known (see, e.g., [7]) that for $\mathscr{H}$ to be a perfect hashing family, it is necessary that $|\mathscr{H}| = \Omega(\log d)$ so that the best sample complexity that can be attained using perfect hashing family is necessarily $\Omega(\varepsilon^{-s/\alpha} \log d)$.

The key observation is that we only need one element of $\mathscr{H}$ to partition $\mathrm{supp}(f)$, a *fixed* albeit unknown subset of $[d]$. Therefore an imperfect hashing family may suffice. Indeed, if we assign each coordinate into s groups uniformly, then the chance that it partitions $\mathrm{supp}(f)$ is $s!/s^s$. If we do so independently for $\lceil \log \delta / \log(1 - s!/s^s) \rceil$ times, we can ensure that with probability at least $1 - \delta$, $\mathrm{supp}(f)$ is partitioned at least once. We take $\mathscr{H}$ to be a family of such uniform hashing functions. Once the hashing family $\mathscr{H}$ is constructed, we can then proceed to compute $\min_{x \in [0,1]^d} f(x)$ via Algorithm 1.

Algorithm 1 Optimizing High Dimensional Sparse Functions

Input: $d, s \in \mathbb{N}$, $k \in \mathbb{N} \cup \{0\}$, $\alpha \in (0, 1]$, $\varepsilon > 0$, $\delta \in (0, 1)$.
Output: An approximate value of $\min_x f(x)$ with error bounded by ε.
Construct $\lceil \log \delta / \log(1 - s!/s^s) \rceil$ independent and uniform hashing functions from $[d]$ to $[s]$. Denote by $\mathscr{H}$ the collection of random hashing functions.
Set $L = \lceil c\varepsilon^{-1/\alpha} \rceil$ for a small constant $c > 0$.
for $h \in \mathscr{H}$ **do**
  Query the oracle to evaluate $f \circ Q_h$ over $\mathscr{L}^s$.
  Construct a polynomial interpolation $\tilde{f}_h$ for the s-variate function $f \circ Q_h$.
  Compute the minimum of $\tilde{f}_h$, denoted by m_h.
end for
Compute the minimum of m_h over all $h \in \mathscr{H}$, denoted by m.
return m.

Together with (4), we get

Theorem 5. *For any fixed $\alpha > 0$, $s \in \mathbb{N}$ and $\delta \in (0, 1)$,*

$$n_\varepsilon^{\mathrm{ran}}(\mathcal{F}_{\alpha,d,s}, \mathrm{OPT}; \delta) \asymp \varepsilon^{-s/\alpha}.$$

Moreover the optimal rate of sample complexity given above can be achieved by Algorithm 1.

It is worth emphasizing again that the complexity of optimizing high dimensional sparse functions, as shown above, is completely free of the ambient dimension, and hence entirely immune of the usual curse-of-dimensionality.

3. Optimal algorithms for approximation

Now we turn our attention to the problem of approximation.

3.1. Deterministic algorithms

As noted before, it suffices to restrict our attention to recovery of function values on the full grid $\mathscr{L}^d$. If $f|_{\mathscr{L}^d}$ is known, a piecewise polynomial interpolation $\tilde{f}$ can be constructed based on its restriction $f|_{\mathscr{L}^d}$ with error of the order $L^{-\alpha}$:

$$\|\tilde{f} - f\|_{C([0,1]^d)} \lesssim L^{-\alpha}, \qquad \forall f \in \mathcal{H}^\alpha([0, 1]^d).$$

Taking $L \asymp \varepsilon^{-1/\alpha}$ yields an ε-approximation of f. Therefore, in the following discussion, we shall focus on the sample complexity of recovering $f|_{\mathscr{L}^d}$. It is not hard to see that the piecewise polynomial approximation $\tilde{f}$ constructed from interpolating $f|_{\mathscr{L}^d}$ satisfies

$$\mathrm{supp}(f|_{\mathscr{L}^d}) = \mathrm{supp}(\tilde{f}).$$

Obviously, $\mathrm{supp}(f|_{\mathscr{L}^d}) \subseteq \mathrm{supp}(f)$. In the case when $\mathrm{supp}(f|_{\mathscr{L}^d}) \subsetneq \mathrm{supp}(f)$, treating coordinates in $\mathrm{supp}(f) \setminus \mathrm{supp}(f|_{\mathscr{L}^d})$ as if they are irrelevant does not affect our ability to construct a good approximation to f at the desired accuracy. Therefore, without loss of generality, we shall assume in the rest of the section that $\mathrm{supp}(f|_{\mathscr{L}^d}) = \mathrm{supp}(f)$.

Similar to the case of optimization, [29] proposed deterministic algorithms that compute approximation of any function $f \in \mathcal{F}_{\alpha,d,s}$ with accuracy ε from $O(\varepsilon^{-s/\alpha} \log d)$ point queries by sampling on the lattice $\mathscr{L}^d$. Their result indicates that

$$n_\varepsilon^{\mathrm{det}}(\mathcal{F}_{\alpha,d,s}, \mathrm{APP}) = O(\varepsilon^{-s/\alpha} \log d). \tag{6}$$

This again is very close to, if not, optimal. Note that

$$n_\varepsilon^{\mathrm{det}}(\mathcal{F}_{\alpha,d,s}, \mathrm{APP}) \geq n_\varepsilon^{\mathrm{det}}(\mathcal{F}_{\alpha,d,s}, \mathrm{OPT}).$$

Hence, immediately from Theorem 4, we get

Proposition 2. *For any $s \geq 2$,*

$$n_\varepsilon^{\mathrm{det}}(\mathcal{F}_{\alpha,d,s}, \mathrm{APP}) = \Omega(\varepsilon^{-s/\alpha} + \varepsilon^{-(s-1)/\alpha} \log d).$$

The question is now, naturally, what happens with randomized algorithms.

3.2. Randomized algorithms

Our algorithm for approximation follows from an idea similar to that for optimization. The key difference between optimization and approximation is that for the former, there is no need to identify which variables are relevant; whereas for the latter, it is essential to do so. We start with the construction of a hashing family $\mathscr{H}$ that contains at least one hashing function that partitions $\mathrm{supp}(f)$, and querying at $\cup_{h\in\mathscr{H}} Q_h(\mathscr{L}^s)$. An extra step is needed to identify $\mathrm{supp}(f)$ before we finally construct an approximation to f.

To do so, we need to first identify the hashing function, denoted by h_*, from $\mathscr{H}$ that partitions $\mathrm{supp}(f)$. There may be multiple elements from $\mathscr{H}$ that partition $\mathrm{supp}(f)$, in which case we can make an arbitrary choice among them. For a given hashing function $h : [d] \to [s]$ write, with slight abuse of notation, $Q_h : \mathscr{L}^s \to \mathscr{L}^d$ as a map such that the jth coordinate of $Q_h(z)$ is $z_{h(j)}$. First consider the case when $|h(\mathrm{supp}(f|_{\mathscr{L}^d}))| = |\mathrm{supp}(f|_{\mathscr{L}^d})|$, meaning that all coordinates in $\mathrm{supp}(f|_{\mathscr{L}^d})$ are mapped to a different value. In this case,

$$|\mathrm{supp}(f \circ Q_h)| = |\mathrm{supp}(f|_{\mathscr{L}^d})|.$$

On the other hand, if $|h(\mathrm{supp}(f|_{\mathscr{L}^d}))| < |\mathrm{supp}(f|_{\mathscr{L}^d})|$, then it is necessarily true that

$$|\mathrm{supp}(f \circ Q_h)| < |\mathrm{supp}(f|_{\mathscr{L}^d})|.$$

Since our scheme retrieves function $f \circ Q_h$ for all $h \in \mathscr{H}$, we can simply choose h_* to be the hashing function that maximizes $|\mathrm{supp}(f \circ Q_h)|$. The fact that there exists a $h \in \mathscr{H}$ that partitions $\mathrm{supp}(f)$ ensures that such an h_* necessarily satisfies (5).

Next we need to figure out which coordinate in $h_*^{-1}(i)$ is relevant where $h^{-1}(i)$ denotes the preimage of a hashing function h, that is

$$h^{-1}(i) = \{j \in [d] : h(j) = i\}.$$

Because h_* partitions $\mathrm{supp}(f|_{\mathscr{L}^d})$, there is a single relevant coordinate in $h_*^{-1}(i)$ for each $i \in \mathrm{supp}(f \circ Q_{h_*})$. Identifying the relevant coordinate in $h_*^{-1}(i)$ can be done via group testing. More specifically, denote by $\mathrm{range}(z, i, f \circ Q_{h_*})$ the range of $f \circ Q_{h_*}$ with all but the ith coordinates fixed at a value $z \in \mathscr{L}^{|\mathrm{supp}(f\circ Q_{h_*})|-1}$. For brevity, we shall assume that $|\mathrm{supp}(f \circ Q_{h_*})| = s$ without loss of generality. Otherwise, we can simply set all coordinates in $\cup_{i\notin\mathrm{supp}(f\circ Q_{h_*})} h^{-1}(i)$ to zero, and ignore them in what follows. Write

$$z_{*,-i} = \operatorname*{argmax}_{z\in\mathscr{L}^{s-1}} \mathrm{range}(z, i, f \circ Q_{h_*}).$$

For any $z \in \mathscr{L}^{s-1}$, let $I_z : \mathscr{L} \to \mathscr{L}^s$ map $a \in \mathscr{L}$ to a vector of length s whose ith coordinate is fixed at value a and the remaining coordinates at value z. Denote by

$$z^+_{*,i} = \operatorname*{argmax}_{a \in \mathscr{L}} (f \circ Q_{h_*} \circ I_{z_{*,-i}})(a),$$

and

$$z^-_{*,i} = \operatorname*{argmin}_{a \in \mathscr{L}} (f \circ Q_{h_*} \circ I_{z_{*,-i}})(a).$$

To identify which coordinate in $h_*^{-1}(i)$ is relevant, we shall now query at $x \in \mathscr{L}^d$ where

$$x_j = \begin{cases} (Q_{h_*} \circ I_{z_{*,-i}}(0))_j & \text{if } j \notin h_*^{-1}(i) \\ \{z^+_{*,i}, z^-_{*,i}\} & \text{otherwise.} \end{cases}$$

A simple binary-splitting algorithm as described by Algorithm 2 can be applied with $\mathcal{A} = h_*^{-1}(i)$, $z_{-\mathcal{A}} = z_{*,-i}$, $z_+ = z^+_{*,i}$ and $z_- = z^-_{*,i}$ to identify which one from $h_*^{-1}(i)$ is relevant. The number of queries is bounded by $\lceil \log_2 |h_*^{-1}(i)| \rceil$. Repeating this to all $i \in \operatorname{supp}(f \circ Q_{h_*})$ leads to a total of at most

$$\sum_{i=1}^{s} \lceil \log_2 |h_*^{-1}(i)| \rceil \le s \lceil \log_2 d \rceil$$

queries.

Algorithm 2 Binary Splitting Algorithm

Input: Set $\mathcal{A} \subset [d]$ that contains a relevant variable, $z_{-\mathcal{A}} \in \mathbb{R}^{d-|\mathcal{A}|}$ that contains value for coordinates outside $\mathcal{A}$, and $\{z_+, z_-\}$ that specifies possible values for coordinates in $\mathcal{A}$ to take.
Output: The coordinate in $\mathcal{A}$ that changes function value.

Query at $x^0 \in \mathbb{R}^d$ where $x^0_{-\mathcal{A}} = z_{-\mathcal{A}}$ and $x^0_i = z_-$ for all $i \in \mathcal{A}$.
while $|\mathcal{A}| > 1$ **do**
  Arbitrarily partition $\mathcal{A} = \mathcal{A}_1 \cup \mathcal{A}_2$ so that $|\mathcal{A}_1| = \lfloor |\mathcal{A}|/2 \rfloor$.
  Query at $x^1 \in \mathbb{R}^d$ where $x^1_{-\mathcal{A}} = z_{-\mathcal{A}}$ and $x^0_i = z_-$ for all $i \in \mathcal{A}_1$ and $x^0_i = z_+$ for all $i \in \mathcal{A} \setminus \mathcal{A}_1$.
  if $f(x^1) \neq f(x^0)$ **then**
    Set $\mathcal{A}_* = \mathcal{A}_1$;
  else
    Set $\mathcal{A}_* = \mathcal{A}_2$.
  end if
  Run Binary Splitting Algorithm with $\mathcal{A}_*$, $x^0_{-\mathcal{A}_*}$ and $\{z_+, z_-\}$.
end while
return $\mathcal{A}$.

Finally, after all relevant variables are identified, we can reconstruct $f|_{\mathscr{L}^d}$ based on the queries from the first step and then form a piecewise polynomial approximation to f via interpolation. See Algorithm 3 for details. The total number of point queries made by Algorithm 3 is upper bounded by

$$\varepsilon^{-s/\alpha} \cdot \lceil \log \delta / \log(1 - s!/s^s) \rceil + s \lceil \log_2 d \rceil.$$

Therefore,

Theorem 6. *For any fixed $\alpha > 0$, $s \in \mathbb{N}$ and $\delta \in (0, 1)$,*

$$n^{\mathrm{ran}}_\varepsilon(\mathcal{F}_{\alpha,d,s}, \mathrm{APP}; \delta) = O\left(\varepsilon^{-s/\alpha} + \log d\right).$$

Moreover the optimal rate of sample complexity given above can be achieved by Algorithm 3.

Algorithm 3 Approximation of High Dimensional Sparse Functions

Input: $d, s \in \mathbb{N}$, $k \in \mathbb{N} \cup \{0\}$, $\alpha \in (0, 1]$, $\varepsilon > 0$, $\delta \in (0, 1)$.
Output: An approximation of f with error bounded by ε.
Construct $\lceil \log \delta / \log(1 - s!/s^s) \rceil$ independent and uniform hashing functions from $[d]$ to $[s]$. Denote by $\mathscr{H}$ the collection of random hashing functions.
Set $L = \lceil \varepsilon^{-1/\alpha} \rceil$.
for $h \in \mathscr{H}$ **do**
  Query the oracle to evaluate f over $Q_h(\mathscr{L}^s)$.
  Compute the support of $f \circ Q_h$.
  Compute $s_h := |\mathrm{supp}(f \circ Q_h)|$.
end for
Identify an arbitrary $h_* \in \mathrm{argmax}_{h \in \mathscr{H}} s_h$.
Initialize the set of relevant variables $\mathcal{A} = \emptyset$.
for $i \in \mathrm{supp}(f \circ Q_{h_*})$ **do**
  Run Binary Splitting Algorithm with $h_*^{-1}(i)$, $z_{*,-i}$ and $\{z_{*,i}^{\pm}\}$ to identify relevant variables $j \in h_*^{-1}(i)$.
  Update $\mathcal{A} = \mathcal{A} \cup \{j\}$.
end for
Construct a piecewise polynomial approximation of $f \circ Q_{h_*}$, denoted by $\tilde{g}$.
return Approximation $\tilde{f}$ such that

$$\tilde{f}(x) = \tilde{g}(x_{\mathcal{A}}), \qquad \forall x \in [0, 1]^d.$$

It is of interest to note that, similar to optimization, when it comes to high precision approximation ($\varepsilon \lesssim (\log d)^{-\alpha/s}$), the complexity given in Theorem 6 becomes

$$n_\varepsilon^{\mathrm{ran}}(\mathcal{F}_{\alpha,d,s}, \mathrm{APP}) = O(\varepsilon^{-s/\alpha}),$$

and is free of the ambient dimension. Because

$$n_\varepsilon^{\mathrm{ran}}(\mathcal{F}_{\alpha,d,s}, \mathrm{APP}) \geq n_\varepsilon^{\mathrm{ran}}(\mathcal{H}^\alpha([0, 1]^s), \mathrm{APP}) = \Omega(\varepsilon^{-s/\alpha}),$$

we conclude that, in this case,

$$n_\varepsilon^{\mathrm{ran}}(\mathcal{F}_{\alpha,d,s}, \mathrm{APP}) \asymp \varepsilon^{-s/\alpha}.$$

On the other hand, for low precision approximation ($\varepsilon \gtrsim (\log d)^{-\alpha/s}$), Theorem 6 indicates that

$$n_\varepsilon^{\mathrm{ran}}(\mathcal{F}_{\alpha,d,s}, \mathrm{APP}) = O(\log d),$$

which is driven entirely by the ambient dimension, and independent of the accuracy ε or smoothness index α. Whether or not this is optimal, however, remains unclear.

4. General sparse functions

We want to emphasize that although we have focused on the space $\mathcal{F}_{\alpha,d,s}$ thus far, the phenomena we observed for $\mathcal{F}_{\alpha,d,s}$ occur broadly for other classes of high dimensional sparse functions as well. In general, one can expect the sample complexity of optimizing a d-variate s-sparse function to be of the same order as optimizing an s-variate function from a compatible class of functions, and approximating it to be of the order of the complexity of approximating an s-variate function, up to an additive factor of $\log d$. Furthermore, these sample complexities can be achieved using ideas similar to those behind Algorithms 1 and 3. In general, we can turn an optimal algorithm for optimizing or approximating an s-variate function into an optimal algorithm for optimizing or approximating a d-variate and s-sparse function using the following meta algorithms.

Algorithm 4 Generic Algorithm for Optimizing High Dimensional Sparse Functions

Input: A subroutine to optimize a s-variate function called OPT_s.
Output: An approximate value of the optimum of f.

Construct $\lceil \log \delta / \log(1 - s!/s^s) \rceil$ independent and uniform hashing functions from $[d]$ to $[s]$, denote the random hashing family $\mathscr{H}$.
for $h \in \mathscr{H}$ **do**
  Call OPT_s to compute the optimum of $f \circ Q_h$, denoted by m_h.
end for
Compute the optimum of m_h over all $h \in \mathscr{H}$, denoted by m.
return m.

Algorithm 5 Generic Algorithm for Approximation of High Dimensional Sparse Functions

Input: A subroutine to approximate a s-variate function called APP_s.
Output: An approximation of a d-variate and s-sparse function f.

Construct $\lceil \log \delta / \log(1 - s!/s^s) \rceil$ independent and uniform hashing functions from $[d]$ to $[s]$, denote the random hashing family $\mathscr{H}$.
for $h \in \mathscr{H}$ **do**
  Call APP_s to approximate $f \circ Q_h$.
  Compute $s_h := |\mathrm{supp}(f \circ Q_h)|$.
end for
Identify the hashing function h_* that maximizes s_h.
Identify relevant variables for f as in Algorithm 3.
Call APP_s to construct an approximation of $f \circ Q_{h_*}$, denoted by $\tilde{g}$.
return Approximation $\tilde{f}$ such that

$$\tilde{f}(x) = \tilde{g}(x_{\mathcal{A}}), \qquad \forall x \in [0,1]^d.$$

It is not hard to see that the sample complexity of Algorithm 4 is of the same order as that of OPT_s, and the sample complexity of Algorithm 5 is of the same order as that of APP_s, up to an additive factor of $O(\log d)$. For illustration, we now consider several specific and commonly encountered examples.

4.1. Additive models

Additive models are widely used in statistics and other related fields. See, e.g., [10]. There has been a lot of interest in recent years in developing effective schemes to approximate a high dimensional and sparse additive function from both statistical and computational perspectives. See, e.g., [12,13,15,20,27,33]. Our treatment could be extended straightforwardly to determine the optimal sample complexity in recovering a high dimensional sparse additive function which remains unknown prior to our work.

Under the additive model, a d-variate function f can be represented by

$$f(x) = f_1(x_1) + \cdots + f_d(x_d), \qquad \forall x \in [0,1]^d.$$

To avoid ambiguity of the above decomposition, write

$$f(x) = f_0 + f_1(x_1) + \cdots + f_d(x_d)$$

where $f_0 \in \mathbb{R}$ and

$$f_j \in \bar{\mathcal{H}}^\alpha([0,1]) := \{g \in \mathcal{H}^\alpha([0,1]) : g(0) = 0\}.$$

Obviously we can query at $(0, \ldots, 0)$ to retrieve f_0. On the other hand, one can query at

$$x \in \mathscr{L}_j := \{0\}^{j-1} \times \mathscr{L} \times \{0\}^{d-j},$$

to optimize or approximate f_j for $j = 1, \ldots, d$. The optimum and approximation of f are simply the sum of those for f_js. Note that to ensure an overall accuracy of ε, it suffices to approximate each component function f_j with error at most $d^{-1}\varepsilon$. As a result, the sample complexity of such a strategy is $O(d \cdot n^{\mathrm{ran}}_{\varepsilon/d}(\mathcal{H}^\alpha([0,1]), \mathrm{APP}; \delta)) = O(d^{1+1/\alpha}\varepsilon^{-1/\alpha})$ for both optimization and approximation.

Now consider additive models with sparsity. Denote by

$$\bar{\mathcal{H}}_j = \{f \in \mathcal{H}^\alpha([0,1]^d) | \mathrm{supp}(f) = \{j\}, f(0, \ldots, 0) = 0\}.$$

Then the space of d-variate additive functions can be written as $\{1\} \oplus \bar{\mathcal{H}}_1 \oplus \cdots \oplus \bar{\mathcal{H}}_d$. Write

$$\mathcal{F}^{\mathrm{add}}_{\alpha,d,s} = \{1\} \bigoplus \left(\bigcup_{S \subset [d], |S| \le s} \bigoplus_{j \in S} \bar{\mathcal{H}}_j \right).$$

Using the aforementioned sampling scheme for additive models with s variables in Algorithm 4 leads to

$$n^{\mathrm{ran}}_\varepsilon(\mathcal{F}^{\mathrm{add}}_{\alpha,d,s}, \mathrm{OPT}; \delta) = O(\varepsilon^{-1/\alpha}).$$

In light of the fact that $\mathcal{F}_{\alpha,d,1} \subset \mathcal{F}^{\mathrm{add}}_{\alpha,d,s}$, we know

$$n^{\mathrm{ran}}_\varepsilon(\mathcal{F}^{\mathrm{add}}_{\alpha,d,s}, \mathrm{OPT}; \delta) \asymp \varepsilon^{-1/\alpha}.$$

Similarly, for approximation, using Algorithm 5 yields

$$n^{\mathrm{ran}}_\varepsilon(\mathcal{F}^{\mathrm{add}}_{\alpha,d,s}, \mathrm{APP}; \delta) = O(\varepsilon^{-1/\alpha} + \log d).$$

This again is optimal when $\varepsilon = O((\log d)^{-\alpha})$.

4.2. Functional ANOVA models

More generally, consider decomposing f into sums of component functions indexed by subsets of $[d]$:

$$f(x) = \sum_{A \subseteq [d]} f_A(x_A), \qquad \forall x \in [0,1]^d.$$

Notable examples of such decompositions include the ANOVA decomposition and the anchored decomposition both of which can be traced back at least to [23]. See also [14] and references therein for more recent developments. We shall focus on the anchored decomposition with respect to $(0, \ldots, 0)$ although, with some modifications, our treatment can also be applied to deal with other types of decompositions.

Write

$$\mathcal{P}_j f(x) = f(x_1, \ldots, x_{j-1}, 0, x_{j+1}, \ldots, x_d), \qquad \forall x \in [0,1]^d.$$

Then

$$f = \left(\prod_{j=1}^d [(I - \mathcal{P}_j) + \mathcal{P}_j] \right) f = \sum_{A \subseteq [d]} \left(\prod_{j \in A} (I - \mathcal{P}_j) \right) \left(\prod_{j \notin A} \mathcal{P}_j \right) f,$$

so that

$$f_A(x_A) = \left(\prod_{j \in A} (I - \mathcal{P}_j) \right) \left(\prod_{j \notin A} \mathcal{P}_j \right) f(\tilde{x}),$$

where $\tilde{x} \in [0,1]^d$ such that $\tilde{x}_A = x_A$ and $\tilde{x}_j = 0$ for all $j \notin A$. See [14] for further details. Of particular interest here are the so-called finite order functions obeying

$$f(x) = \sum_{A \subset [d], |A| \le r} f_A(x_A), \qquad \forall x \in [0,1]^d. \tag{7}$$

It is not hard to see that additive models correspond to $r = 1$. As before, we shall assume that the component functions f_As are smooth in that $f_A \in \mathcal{H}^\alpha([0,1]^r)$, and denote by $\mathcal{H}_r^{\text{anova}}$ the collection of functions satisfying (7). Note that $\mathcal{H}_r^{\text{anova}}$ is closely related to reproducing kernel Hilbert spaces with the so-called finite-order weights which have been studied extensively. See, e.g., Section 25.4 of [19]. Our interest here is on how to exploit the additional sparsity for finite order functions. More specifically, write, for a fixed $s \ge r$,

$$\mathcal{F}_{\alpha,d,s,r}^{\text{anova}} = \left\{ f \in \mathcal{H}_r^{\text{anova}} : |\text{supp}(f)| \le s \right\}.$$

An effective sampling scheme for $\mathcal{F}_{\alpha,d,s,r}^{\text{anova}}$ is similar to that for additive models. More specifically, because $f_A \in \mathcal{H}^\alpha([0,1]^{|A|})$, it can be approximated with accuracy ε via piecewise polynomial interpolation of queries on a $|A|$ dimensional lattice. For any set $S \subset [d]$ such that $|S| \le s$, we can repeat this for each $A \subset S$ such that $|A| \le r$ and compute the approximation, denoted by $\tilde{f}_S$, of f_S by summing them up. The optimum of f_S can then be computed by that of $\tilde{f}_S$. For any given S, the accuracy of this procedure is $O(\varepsilon)$, and the sample complexity is $O(\varepsilon^{-r/\alpha})$. Using this sampling scheme in conjunction with Algorithm 4 leads to

$$n_\varepsilon^{\text{ran}}(\mathcal{F}_{\alpha,d,s,r}^{\text{anova}}, \text{OPT}; \delta) = O(\varepsilon^{-r/\alpha}),$$

which is optimal because $\mathcal{F}_{\alpha,d,r} \subset \mathcal{F}_{\alpha,d,s,r}^{\text{anova}}$. Similarly, when using such a sampling scheme in Algorithm 5, we get

$$n_\varepsilon^{\text{ran}}(\mathcal{F}_{\alpha,d,s,r}^{\text{anova}}, \text{APP}; \delta) = O(\varepsilon^{-r/\alpha} + \log d).$$

In the special case when $r = 2$, this improves a recent result by [28] who develops a sophisticated sampling scheme that, under additional technical assumptions, with accuracy ε from $O(\varepsilon^{-2/\alpha}(\log d)^3)$ point queries.

4.3. Sparse grids

As a final example, we consider optimizing or approximating functions of mixed smoothness. More specifically, write

$$\tilde{\mathcal{H}}_d = \{f : [0,1]^d \to \mathbb{R} | f|_{\partial[0,1]^d} = 0, \|D^{(l)} f\|_{C([0,1]^d)} \le 1, \forall l_1, \ldots, l_d \le 2\}.$$

and

$$\mathcal{V}_{d,s} = \{f \in \tilde{\mathcal{H}}_d, |\text{supp}(f)| \le s\}.$$

It is not hard to see that in this case, a direct application of Algorithm 1 incurs a sample complexity $O(\varepsilon^{-s/2})$, and Algorithm 3 has a sample complexity $O(\varepsilon^{-s/2} + \log d)$. Although these rates indicate that the curse-of-dimensionality in terms of the ambient dimension can be alleviated, they still have an exponential dependence on s and therefore may not be practical for moderate values of s. Fortunately, these complexities can be improved by using sparse grids in Algorithms 4 and 5 when f has mixed smoothness.

More specifically, sparse grids interpolate not on the full grid $\mathscr{L}^s$ but rather a subgrid with degrees of freedom $O(L(\log L)^{s-1})$ that is optimized to take advantage of the mixed smoothness. The idea can be traced back at least to [1,22,34] among others. See also [2,8] for an overview of more recent developments, and Chapter 15 of [18] for a survey from the perspective of information based complexity. It can be shown that the sample complexity of the sparse grid is $O(\varepsilon^{-1/2}|\log \varepsilon|^{3(s-1)/2})$ for approximating functions from $\tilde{\mathcal{H}}_s$. As a result, the sample complexity for optimizing functions from $\mathcal{V}_{d,s}$ via Algorithms 4 is $O(\varepsilon^{-1/2}|\log \varepsilon|^{3(s-1)/2})$, for approximation via Algorithms 5 $O(\varepsilon^{-1/2}|\log \varepsilon|^{3(s-1)/2} + \log d)$. The fact that the dependence on s is only through the logarithmic factor of ε offers tremendous practical appeal.

5. Concluding remarks

In this paper, we studied the information based complexity for optimizing and approximating high dimensional smooth functions with sparsity. Our results reveal several interesting effects of the sparsity in mitigating and oftentimes eliminating the curse-of-dimensionality, and also highlight the role of randomization in approximating high dimensional sparse functions.

In addition to the information based complexity, computational complexity is another recurring challenge that we often need to deal with when faced high dimensional problems. It is worth noting that the computational complexity of the algorithms presented here are at most polynomials of the sample complexity, and generally tractable in practice. This further suggests that information based complexity offers a useful perspective of the fundamental difficulties for high dimensional sparse functions. For the ease of presentation, as well as technical considerations, we have made several simplifying assumptions in our analysis. The promising insights we obtained while doing so suggest that a more general treatment may be a worthwhile direction for future work.

Acknowledgments

The authors wish to thank two anonymous referees for their careful reading and insightful comments that helped greatly to improve the presentation and fix numerous mistakes in earlier drafts.

References

[1] K.I. Babenko, Approximation by trigonometric polynomials in a certain class of periodic functions of several variables, Dokl. Akad. Nauk 132 (5) (1960) 982–985.
[2] Hans-Joachim Bungartz, Michael Griebel, Sparse grids, Acta Numer. 13 (2004) 147–269.
[3] Emmanuel J. Candès, Compressive sampling, in: Proceedings of the International Congress of Mathematicians, vol. 3, 2006, pp. 1433–1452, Madrid, Spain.
[4] Albert Cohen, Ronald Devore, Guergana Petrova, Przemysław Wojtaszczyk, Finding the minimum of a function, Methods Appl. Anal. 20 (4) (2013) 365–382.
[5] Ronald DeVore, Guergana Petrova, Przemyslaw Wojtaszczyk, Approximation of functions of few variables in high dimensions, Constr. Approx. 33 (1) (2011) 125–143.
[6] David L. Donoho, Compressed sensing, IEEE Trans. Inf. Theory 52 (4) (2006) 1289–1306.
[7] Michael L. Fredman, János Komlós, On the size of separating systems and families of perfect hash functions, SIAM J. Algebr. Discrete Methods 5 (1) (1984) 61–68.
[8] Jochen Garcke, Michael Griebel, Sparse Grids and Applications, Vol. 88, Springer Science & Business Media, 2012.
[9] Luisa Gargano, János Körner, Ugo Vaccaro, Sperner capacities, Graphs and Comb. 9 (1) (1993) 31–46.
[10] Trevor Hastie, Robert J. Tibshirani, Generalized Additive Models, CRC press, 1990.
[11] Viktor V. Ivanov, On optimum minimization algorithms in classes of differentiable functions, Dokl. Akad. Nauk 201 (3) (1971) 527–530.
[12] Vladimir Koltchinskii, Ming Yuan, Sparse recovery in large ensembles of kernel machines, in: Proceedings of COLT, Vol. 3, 2008.
[13] Vladimir Koltchinskii, Ming Yuan, Sparsity in multiple kernel learning, Ann. Statist. 38 (6) (2010) 3660–3695.
[14] F. Kuo, I. Sloan, Grzegorz Wasilkowski, Henryk Woźniakowski, On decompositions of multivariate functions, Math. Comput. 79 (270) (2010) 953–966.
[15] Lukas Meier, Sara Van de Geer, Peter Bühlmann, High-dimensional additive modeling, Ann. Statist. 37 (6B) (2009) 3779–3821.
[16] Erich Novak, Deterministic and Stochastic Error Bounds in Numerical Analysis, Vol. 1349, Springer, 1988.
[17] Erich Novak, Henryk Woźniakowski, Tractability of Multivariate Problems: Linear Information, Vol. 6, European Mathematical Society, 2008.
[18] Erich Novak, Henryk Woźniakowski, Tractability of Multivariate Problems: Volume II: Standard Information for Functionals, European Mathematical Society Publishing House Zürich, 2010.
[19] Erich Novak, Henryk Woźniakowski, Tractability of Multivariate Problems: Volume III: Standard Information for Operators, European Mathematical Society Publishing House Zürich, 2012.
[20] Garvesh Raskutti, Martin J. Wainwright, Bin Yu, Minimax-optimal rates for sparse additive models over kernel classes via convex programming, J. Mach. Learn. Res. 13 (Feb) (2012) 389–427.
[21] Ian H. Sloan, Henryk Woźniakowski, When are quasi-Monte Carlo algorithms efficient for high dimensional integrals?, J. Complexity 14 (1) (1998) 1–33.
[22] Sergei Abramovich Smolyak, Quadrature and interpolation formulas for tensor products of certain classes of functions, Dokl. Akad. Nauk 148 (5) (1963) 1042–1045.
[23] I.M. Sobol, Multidimensional Quadrature Formulas and Haar Functions, Nauka, Moscow, 1969.

[24] V.M. Tikhomirov, Widths of sets in functional spaces and approximation theory, Usp. Mat. Nauk 15 (3) (1960) 81–120.
[25] Joseph Frederick Traub, Grzegorz Włodzimierz Wasilkowski, Henryk Woźniakowski, Information, uncertainty, complexity, Addison-Wesley Publishing Company, 1983.
[26] Joseph Frederick Traub, Henryk Woźniakowski, A General Theory of Optimal Algorithms, Academic Press, 1980.
[27] Hemant Tyagi, Bernd Gärtner, Andreas Krause, Efficient sampling for learning sparse additive models in high dimensions, in: Advances in Neural Information Processing Systems, 2014, pp. 514–522.
[28] Hemant Tyagi, Anastasios Kyrillidis, Bernd Gärtner, Andreas Krause, Algorithms for learning sparse additive models with interactions in high dimensions, Inf. Inference J. IMA 7 (2) (2017) 183–249.
[29] Przemysław Wojtaszczyk, Complexity of approximation of functions of few variables in high dimensions, J. Complexity 27 (2) (2011) 141–150.
[30] H. Woźniakowski, A survey of information-based complexity, J. Complexity 1 (1985) 11–44.
[31] Henryk Woźniakowski, Tractability and strong tractability of linear multivariate problems, J. Complexity 10 (1) (1994) 96–128.
[32] Henryk Woźniakowski, ABC on IBC, J. Complexity 52 (2019) 2–23.
[33] Ming Yuan, Ding-Xuan Zhou, Minimax optimal rates of estimation in high dimensional additive models, Ann. Statist. 44 (6) (2016) 2564–2593.
[34] Christoph Zenger, Sparse grids, in: Parallel Algorithms for Partial Differential Equations, 1991.