Optimal computation

Ronald A. DeVore*

Abstract. A large portion of computation is concerned with approximating a function u. Typically, there are many ways to proceed with such an approximation leading to a variety of algorithms. We address the question of how we should evaluate such algorithms and compare them. In particular, when can we say that a particular algorithm is optimal or near optimal? We shall base our analysis on the approximation error that is achieved with a given (computational or information) budget n. We shall see that the formulation of optimal algorithms depends to a large extent on the context of the problem. For example, numerically approximating the solution to a PDE is different from approximating a signal or image (for the purposes of compression).

Mathematics Subject Classification (2000). Primary 41-02, 46-02; Secondary 62C20, 65N30, 68Q25, 74S05.

Keywords. Optimal computation, encoding and compression, learning theory, entropy and widths.

1. Introduction

A generic scientific problem is to approximate a function u. The problem takes different forms depending on what we know about u. We describe five common settings.

The Data Fitting Problem (DFP). We are given data $\lambda_j(u)$, $j = 1, 2, \ldots, n$, where each λ_j is a linear functional. The problem is to approximate u the best we can from this information. Often the $\lambda_j(u)$'s are point values of u or averages of u over certain sets (called cells).

The Sensing Problem (SP). In this setting we may ask for the values $\lambda_j(u)$, $j = 1, \ldots, n$, of any linear functionals λ_j applied to u . We are given a budget of n such questions and we wish to determine what are the best questions to ask in order to approximate u effectively. This problem differs from DFP because we can choose the functionals to apply.

The Encoding Problem (EP). Here we have complete knowledge of u. We are given a bit budget n and we wish to transmit as much information about u as possible while

*This paper was prepared while the author was visiting Electrical and Computer Engineering Department at Rice University. The author thanks the participants of the Rice Compressed Sensing Seminar, especially Rich Baraniuk and Mike Wakin, for valuable discussions on compresses sensing. He is also grateful to Carl de Boor, Albert Cohen, Emmanuel Candès, Anna Gilbert, and Guergana Petrova for reading versions of this manuscript.

Proceedings of the International Congress
of Mathematicians, Madrid, Spain, 2006

using at most n bits. An encoder maps u into a bitstream and a decoder converts the bitstream into a function which approximates u. Both these maps are typically nonlinear.

The Computation Problem (CP). We are only given the information that u is a solution to some (linear or nonlinear) equation $A(u) = f$. We have complete knowledge of the operator A and any additional information (such as boundary or initial conditions) that are sufficient to uniquely determine u. We are given a computational budget n, say of floating point operations (flops), and we wish to approximate u as efficiently as possible within this budget. This problem is related to numerically inverting the operator A.

The Learning Problem (LP). We are given data $z_i = (x_i, y_i) \in X \times Y, i = 1, \dots, n$, which are drawn independently with respect to some unknown probability measure ρ on $X \times Y$. We wish from this data to fit a function which best represents how the response variable y is related to x. The best representation (in the sense of least squares minimization) is given by the regression function $f_\rho(x) := E(y|x)$ with E the expectation. Since we do not know ρ, we do not know f_ρ The problem is to best approximate f_ρ from the given sample data.

These problems have a long history and still remain active and important research areas. The first three of these problems are related to major areas of Approximation Theory and Information Based Complexity and our presentation is framed by core results in these disciplines. CP is the dominant area of Numerical Analysis and LP is central to Nonparametric Statistics. The complexity of algorithms is also a major topic in Theoretical Computer Science. The purpose of this lecture is not to give a comprehensive accounting of the research in these areas. In fact, space will only allow us to enter two of these topics (SP and LP) to any depth. Rather, we want to address the question of how to evaluate the myriad of algorithms for numerically resolving these problems and decide which of these is best. Namely, we ask "what are the ways in which we can evaluate algorithms?"

2. Some common elements

There are some common features to these problems which we want to underscore. The obvious starting point is that in each problem we want to approximate a function u.

2.1. Measuring performance. To measure the success of the approximation, we need a way to measure error between the target function u and any candidate approximation. For this, we use a norm $\|\cdot\|$. If u_n is our approximation to u, then the error in this approximation is measured by

$$\|u - u_n\|. \tag{2.1}$$

Thus, our problem is to make this error as small as possible within the given budget n.

The norm may be of our choosing (in which case we would want to have a theory that applies to a variety of norms) or it may be dictated by the problem at hand. The typical choices are the L_p norms, $1 \le p \le \infty$. Suppose that Ω is a domain in $\mathbb{R}^d$ where $\mathbb{R}^d$ is the d dimensional Euclidean space. We define

$$\|g\|_{L_p(\Omega)} := \begin{cases} (\int_\Omega |g(x)|^p \, dx)^{1/p}, & 1 \le p < \infty, \\ \operatorname*{esssup}_{x\in\Omega} |g(x)|, & p = \infty. \end{cases} \tag{2.2}$$

When studying the solutions to PDEs, norms involving derivatives of u are often more appropriate. We shall delay a discussion of these norms till needed.

In numerical considerations, the norms (2.2) are replaced by discrete versions. If $x \in \mathbb{R}^N$, then

$$\|x\|_{\ell_p} := \begin{cases} \left(\sum_{j=1}^N |x_j|^p\right)^{1/p}, & 0 < p < \infty, \\ \max_{j=1\dots,N} |x_j|, & p = \infty. \end{cases} \tag{2.3}$$

2.2. The form of algorithms: linear versus nonlinear. The numerical algorithms we consider will by necessity be a form of approximation. To understand them, we can use the analytical tools of approximation theory. This is a classical subject which began with the work of Weierstrass, Bernstein, Chebyshev, and Kolmogorov. The quantitative portion of approximation theory seeks to understand how different methods of approximation perform in terms of rates of convergence. If a certain method of approximation is used in the construction of an algorithm then approximation theory can tell us the optimal performance we could expect. Whether we reach that performance or something less will be a rating of the algorithm.

Approximation theory has many chapters. We will partially unfold only one of these with the aim of describing when numerical algorithms are optimal. To keep the discussion as elementary as possible we will primarily focus on approximation in Hilbert spaces where the theory is most transparent. For approximation in other spaces, the reader should consult one of the major books [16], [31].

Let $\mathcal{H}$ be a separable Hilbert space with inner product $\langle \cdot, \cdot \rangle$ and its induced norm $\|f\| := \langle f, f \rangle^{1/2}$. The prototypical examples for $\mathcal{H}$ would be the space $L_2(\Omega)$ defined in (2.2) and ℓ_2 defined in (2.3). We shall consider various types of approximation in $\mathcal{H}$ which will illustrate notions such as linear, nonlinear, and greedy approximation. At the start, we will suppose that $B := \{g_k\}_{k=1}^\infty$ is a complete orthonormal system for $\mathcal{H}$ and use linear combinations of these basis vectors to approximate u. Later, we shall consider more general settings where B is replaced by more general (redundant) systems.

We begin with linear approximation in this setting. We consider the linear spaces $V_n := \operatorname{span}\{g_k\}_{k=1}^n$. These spaces are nested: $V_n \subset V_{n+1}$, $n = 1, \dots$. Each element

$f \in \mathcal{H}$ has a unique expansion

$$f = \sum_{k=1}^{\infty} c_k g_k, \quad c_k := c_k(f) := \langle f, g_k \rangle, \ k = 1, 2, \dots. \tag{2.4}$$

For an important concrete example, the reader can have in mind the space $\mathcal{H} = L_2(\Pi)$ of 2π-periodic functions defined on $\mathbb{R}$ and the Fourier basis. Then (2.4) is just the expansion of f into its Fourier series.

Given $f \in \mathcal{H}$ the function $P_{V_n} f := \sum_{k=1}^{n} c_k(f) g_k$ is the best approximation to f from V_n and the error we incur in such an approximation is given by

$$E_n(f) := \|f - P_{V_n}(f)\| = \Big(\sum_{k=n+1}^{\infty} |c_k(f)|^2 \Big)^{1/2}, \quad n = 1, 2, \dots. \tag{2.5}$$

We are in the wonderful situation of having an explicit formula for the error of approximation in terms of the coefficients $c_k(f)$. We know that for any $f \in \mathcal{H}$ the right side of (2.5) tends to zero as n tends to infinity. The faster the rate of decay, the better we can approximate f and the nicer f is with respect to this basis.

To understand the performance of an approximation process, such as the one described above, it is useful to introduce approximation classes which gather together all functions which have a common approximation rate. For us, it will be sufficient to consider the classes $\mathcal{A}^r$, $r > 0$, consisting of all functions f that are approximated with a rate $O(n^{-r})$. For example, in the case we are discussing $\mathcal{A}^r := \mathcal{A}^r((V_n)) := \mathcal{A}^r((V_n), \mathcal{H})$ consists of all functions $f \in \mathcal{H}$ such that

$$E_n(f) \le M n^{-r}, \quad n = 1, 2, \dots. \tag{2.6}$$

The smallest M such that (2.6) holds is defined to be the norm on this space:

$$|f|_{\mathcal{A}^r} := \sup_{n \ge 1} n^r E_n(f). \tag{2.7}$$

Notice that these approximation spaces are also nested: $\mathcal{A}^r \subset \mathcal{A}^{r'}$ if $r \ge r'$. Given an $f \in \mathcal{H}$ there will be a largest value of $r = r(f, B)$ for which $f \in \mathcal{A}^{r'}$ for all $r' < r$. We can think of this value of r as measuring the smoothness of f with respect to this approximation process or what is the same thing, the smoothness of f with respect to the basis $\{g_k\}$.

For standard orthonormal systems, the approximation spaces $\mathcal{A}^r$ often have an equivalent characterization as classical smoothness spaces. For example, in the case of the Fourier basis, $\mathcal{A}^r$ is identical with the Besov space $B^r_\infty(L_2(\Pi))$. This space is slightly larger than the corresponding Sobolev space $W^r(L_2(\Pi))$. In the case that r is an integer, $W^r(L_2(\Pi))$ is the set of all $f \in L_2(\Pi)$ whose r-th derivative $f^{(r)}$ is also in $L_2(\Pi)$. We do not have the space here to go into the precise definitions of smoothness spaces, but if the reader thinks of the smoothness order as simply corresponding to the number of derivatives that will give the correct intuition.

Observe that two elements are coming into play: the basis we select and the ordering of that basis. A function f may have a faster convergent expansion with respect to one basis B than another B'. That is $r(f, B) > r(f, B')$. If we knew this in advance the better basis would be preferable. Such knowledge is only present through some additional analysis of the problem at hand, e.g. in the case of numerically solving PDEs such information could be provided by a regularity theorem for the PDE. The ordering of the basis functions also plays an important role. Reordering these basis functions results in a different rate of decay for the approximation error and therefore a different $r(f, B)$. Such reordering is done in practice through nonlinear approximation which we now discuss.

Approximation by the elements of V_n is called *linear approximation* because the approximants are taken from the linear space V_n. This is to be contrasted with the following notion of n-term approximation. For each $n \geq 1$, we let Σ_n denote the set of all functions that can be expressed as a linear combination of n terms of the orthonormal basis:

$$S = \sum_{k \in \Lambda} a_k g_k, \quad \#\Lambda \leq n, \tag{2.8}$$

where $\#\Lambda$ is the cardinality of Λ. We consider the approximation of $f \in \mathcal{H}$ by the elements of Σ_n and define the error of such approximation by

$$\sigma_n(f) := \sigma_n(f)_{\mathcal{H}} := \inf_{S \in \Sigma_n} \|f - S\|, \quad n = 1, 2, \ldots. \tag{2.9}$$

Notice that E_n is generally reserved for the error in linear approximation and σ_n for the error in nonlinear approximation.

Another view of n-term approximation is that we approximate the function f by the elements of a linear space W_n spanned by the elements of n basis functions. However, it differs from linear approximation in that we allow the space W_n to also be chosen depending on f, that is, it is not fixed in advance as was the case for linear approximation.

It is very easy to describe the best approximant to f from Σ_n and the resulting error of approximation. Given $f \in \mathcal{H}$ we denote by (c_k^*) the decreasing rearrangement of the sequence $(c_j = c_j(f))$. Thus, $|c_k^*|$ is the k-th largest of the numbers $|c_j(f)|$, $j = 1, 2, \ldots$. Each $c_k^* = c_{j_k}(f)$ for some j_k. The choice of the mapping $k \mapsto j_k$ is not unique because of possible ties in the size of coefficient but the following discussion is immune to such differences. A best approximation to $f \in \mathcal{H}$ from Σ_n is given by

$$S^* = \sum_{k=1}^{n} c_{j_k}(f) g_{j_k} = \sum_{j \in \Lambda_n^*} c_j(f) g_j, \quad \Lambda_n^* := \Lambda_n^*(f) := \{j_1, \ldots, j_n\}, \tag{2.10}$$

and the resulting error of approximation is

$$\sigma_n(f)^2 = \sum_{k > n} (c_k^*)^2 = \sum_{j \notin \Lambda_n^*} |c_j(f)|^2. \tag{2.11}$$

Indeed, if $S = \sum_{j\in\Lambda} a_j g_j$ is any element of Σ_n, then

$$\|f - S\|^2 = \sum_{j\in\Lambda}(c_j - a_j)^2 + \sum_{j\in\Lambda^c} c_j^2, \tag{2.12}$$

where Λ^c is the complement of Λ. The second sum on the right side of (2.12) is at least as large as $\sum_{k>n}(c_k^*)^2$ and so we attain the smallest error by taking a set of indices $\Lambda = \Lambda_n^*(f)$ corresponding to the n largest coefficients and then taking $a_j = c_j(f)$ for $j \in \Lambda$.

Notice that the space Σ_n is not linear. If we add two elements from Σ_n, we will generally need $2n$ terms to represent the sum. For this reason, n-term approximation is a form of *nonlinear approximation*. We can define approximation classes $\mathcal{A}^r((\Sigma_n), \mathcal{H})$ for this form of approximation by replacing $E_n(f)$ by $\sigma_n(f)$ in (2.6) and (2.7). To distinguish between linear and nonlinear approximation we will sometimes write $\mathcal{A}^r(L)$ and $\mathcal{A}^r(NL)$ for the two approximation classes thereby indicating that the one corresponds to linear approximation and the other to nonlinear approximation.

It is easy to characterize when an f belongs to $\mathcal{A}^r((\Sigma_n), \mathcal{H})$ in terms of the coefficients $c_k(f)$. For this, recall that a sequence (a_k) is said to be in the space $w\ell_p$ (weak ℓ_p) if

$$\#\{k : |a_k| \geq \eta\} \leq M^p \eta^{-p} \tag{2.13}$$

and the smallest M for which (2.13) holds is called the weak ℓ_p norm ($\|(a_k)\|_{w\ell_p}$) of this sequence. An equivalent definition is that the decreasing rearrangement of (a_k) satisfies

$$|a_k^*| \leq M^{1/p} k^{-1/p}, \quad k = 1, 2, \ldots. \tag{2.14}$$

A simple exercise proves that $f \in \mathcal{A}^r((\Sigma_n), \mathcal{H})$ if and only if $(c_k(f)) \in w\ell_p$ with $1/p = r + 1/2$, and the norms $|f|_{\mathcal{A}^r}$ and $\|(c_k(f))\|_{w\ell_p}$ are equivalent (see [30] or [16]). Notice that $\mathcal{A}^r(L) \subset \mathcal{A}^r(NL)$ but the latter set is much larger. Indeed, for linear approximation a function $f \in \mathcal{H}$ will be approximated well only if its coefficients decay rapidly with respect to the usual basis ordering but in nonlinear approximation we can reorder the basis in any way we want. As an example, consider again the Fourier basis. The space $\mathcal{A}^{1/2}(NL)$ consists of all functions whose Fourier coefficients are in $w\ell_1$. A slightly stronger condition is that the Fourier coefficients are in ℓ_1 which means the Fourier series of f converges absolutely.

In n-term approximation, the n-dimensional space used in the approximation depends on f. However, this space is restricted to be spanned by n terms of the given orthonormal basis. If we are bold, we can seek even more approximation capability by allowing the competition to come from more general collections of n-dimensional spaces. However, we would soon see that opening the competition to be too large will render the approximation process useless in computation since it would be impossible to implement such a search numerically (this topic will be discussed in more detail in Section 4).

There is a standard way to open up the possibilities of using more general families in the approximation process. We say a collection of function $\mathcal{D} = \{g\}_{g\in\mathcal{D}} \subset \mathcal{H}$ is a

dictionary if each $g \in \mathcal{D}$ has norm one ($\|g\| = 1$). We define $\Sigma_n := \Sigma_n(\mathcal{D})$ as the set of all S that are a linear combination of at most n dictionary elements: $S = \sum_{g\in\Lambda} c_g g$ with $\#(\Lambda) \le n$. The n-term approximation error σ_n and the approximation classes $\mathcal{A}^r(\mathcal{D}, \mathcal{H})$ are then defined accordingly. Notice that the elements in $\mathcal{D}$ need not be linearly independent, i.e., the dictionary allows for redundancy.

It is a bit surprising that meaningful theorems about $\mathcal{A}^r$ can be proved in this very general setting. To find good n-term approximations, we cannot simply select indices with large coefficients because of the possible redundancy in the dictionary. Rather, we proceed by an important technique known as *greedy approximation* (sometimes called *matching pursuit*). This is an iterative procedure which selects at each step a best one-term approximation to the current residual. Given $f \in \mathcal{H}$, we initially define $f_0 := 0$ and the residual $r_0 := f - f_0 = f$. Having defined the current approximation f_{n-1} and its residual $r_{n-1} := f - f_{n-1}$ for some $n \ge 1$, we will choose an element $g_n \in \mathcal{D}$ and update the approximation to f by including g_n in the n-term approximation. The usual way of proceeding is to choose g_n as

$$g_n := \operatorname*{Argmax}_{g\in\mathcal{D}} \langle r_{n-1}, g\rangle, \tag{2.15}$$

although there are important variants of this strategy. Having chosen g_n in this way, there are different algorithms depending on how we proceed. In the Pure Greedy Algorithm (PGA), we define

$$f_n := f_{n-1} + \langle f_{n-1}, g_n\rangle g_n \tag{2.16}$$

which yields the new residual $r_n := f - f_n$. We can do better, but with more computational cost, if we define

$$f_n := P_{V_n} f \tag{2.17}$$

where $V_n := \operatorname{span}\{g_1, \dots, g_n\}$. This procedure is called the Orthogonal Greedy Algorithm (OGA). There is another important variant which is analogous to numerical descent methods called Restricted Greedy Approximation(RGA). It defines

$$f_n := \alpha_n f_{n-1} + \beta_n g_n \tag{2.18}$$

where $0 < \alpha_n < 1$ and $\beta_n > 0$. Typical choices for α_n are $1 - 1/n$ or $1 - 2/n$. There are other variants in the RGA where the choice of g_n is altered. The most general procedure is to allow α, β, g to be arbitrary and choose $\alpha f_{n-1} + \beta g$ that minimizes the norm of the residual $r = f - \alpha f_{n-1} - \beta g$.

Greedy algorithms have been known for decades. Their numerical implementation and approximation properties were first championed in statistical settings ([42], [4], [46]). The approximation properties have some analogy to those for nonlinear approximation from a basis but are not as far reaching. To briefly describe some of these results, let $\mathcal{L}_1$ consist of all functions $f \in \mathcal{H}$ such that

$$f = \sum_{g\in\mathcal{D}} c_g g, \quad \sum_{g\in\mathcal{D}} |c_g| < +\infty. \tag{2.19}$$

We can define a norm on this space by

$$|f|_{\mathcal{L}_1} := \inf\left\{\sum_{g\in\mathcal{D}} |c_g| : \ f = \sum_{g\in\mathcal{D}} c_g g\right\}. \tag{2.20}$$

Thus, the unit ball of $\mathcal{L}_1$ is the convex closure of $\mathcal{D} \cup (-\mathcal{D})$.

If $f \in \mathcal{L}_1$, then both the OGA and properly chosen RGA will satisfy

$$\|f - f_n\| \le C_0 |f|_{\mathcal{L}_1} n^{-1/2}, \quad n = 1, 2, \ldots, \tag{2.21}$$

as was proved in [42] (see also [29]). The convergence rates for the PGA are more subtle (see [29]) and its convergence rate on $\mathcal{L}_1$ are not completely known. These results show that $\mathcal{L}_1 \subset \mathcal{A}^{1/2}$ which is quite similar to our characterization of this approximation class for nonlinear approximation when using a fixed orthonormal basis.

One unsatisfactory point about (2.21) is that it does not give any information about convergence rates when f is not in $\mathcal{L}_1$. Using interpolation, one can introduce function classes that guarantee approximation rates $O(n^{-r})$ when $0 < r < 1/2$ (see [5]). Also, using (2.21), one can prove that for $r > 1/2$ a sufficient condition for f to be in $\mathcal{A}^r(\mathcal{D}, \mathcal{H})$ is that it has an expansion $f = \sum_{g\in\mathcal{D}} c_g g$ with $(c_g) \in \ell_p$, $p := (r + 1/2)^{-1}$. However, this condition is generally not sufficient to ensure the convergence of the greedy algorithm at the corresponding rate n^{-r}.

Although greedy approximation is formulated in a very general context, any numerical algorithm based on this notion will have to deal with finite dictionaries. The size of the dictionary will play an important role in the number of computations needed to execute the algorithm. Greedy approximation remains an active and important area for numerical computation. A fairly up to date survey of greedy approximation is found in [56]. We will touch on greedy approximation again in our discussion of the Sensing Problem and the Learning Problem.

3. Optimality of algorithms

Our goal is to understand what is the optimal performance that we can ask from an algorithm. Recall that in each of our problems, our task is to approximate a function u. What differs in these problems is what we know in advance about u and how we can access additional information about u.

Because of the diversity of problems we are discussing, we shall not give a precise definition of an algorithm until a topic is discussed in more detail. Generically an algorithm is a sequence $\boldsymbol{A} = (A_n)$ of mappings. Here n is the parameter associated to each of our problems, e.g., it is the number of computations allotted in the computation problem. The input for the mapping A_n is different in each of our problems. For example, in the data fitting problem it is values $\lambda_j(u)$, $j = 1, \ldots, n$, that are given to us. The output of A_n is an approximation u_n to the target function u. To study the performance of the algorithm, we typically consider what happens in this

approximation as $n \to \infty$. Such a theory will miss out on important but usually very subtle questions about performance for small n.

We fix the space X and the norm $\|\cdot\| = \|\cdot\|_X$ in which to measure error. Then, for any u,

$$E(u, A_n) := \|u - u_n\|, \quad n = 1, 2, \ldots, \tag{3.1}$$

measures how well the algorithm approximates u. It is tempting to define an optimal algorithm to be a sequence (A_n^*) such that

$$E(u, A_n^*) \leq \inf_{A_n} E(u, A_n), \quad u \in X,\ n = 1, 2, \ldots, \tag{3.2}$$

where the infimum is taken over all algorithms A_n. However, such a definition is meaningless, since to achieve such a performance the algorithm would typically involve a search which is prohibitive from both a theoretical and numerical perspective.

Here is another important point. An algorithm only sees the given data. In the Recovery Problem and Sensing Problem, A_n will only act on the data $\lambda_j(u)$, $j = 1, \ldots, n$. This means that many functions have the same approximation u_n. If $\mathcal{N}$ is the null space consisting of all functions v such that $\lambda_j(v) = 0$, $j = 1, \ldots, n$, then all functions $u + \eta$, $\eta \in \mathcal{N}$, have the same data and hence u_n is a common approximation to all of these functions. Since $\|\eta\|$ can be arbitrarily large, we cannot say anything about $\|u - u_n\|$ being small without additional information about u.

There are two ways to come to a meaningful notion of optimality which we shall describe: optimality on classes and instance-optimal. We begin with the first of these which is often used in statistics, approximation theory and information based complexity. Consider any compact set $K \subset X$. We define

$$E(K, A_n) := \sup_{u \in K} E(u, A_n), \quad n = 1, 2, \ldots, \tag{3.3}$$

which measures the worst performance of A_n on K. We shall say that (A_n^*) is *near optimal* on K with constant $C = C(K)$ if

$$E(K, A_n^*) \leq C \inf_{A_n} E(K, A_n), \quad n = 1, 2, \ldots. \tag{3.4}$$

If $C = 1$ we say (A_n^*) is *optimal* on K. Usually, it is not possible to construct optimal algorithms, so we shall mainly be concerned with near optimal algorithms, although the size of the constant C is a relevant issue.

The deficiency of the above notion of optimality is that it depends on the choice of the class K. An appropriate class for u may not be known to us. Thus, an algorithm is to be preferred if it is near optimal for a large collection of classes K. We say that an algorithm $\boldsymbol{A}$ is *universal* for the collection $\mathcal{K}$, if the bound (3.4) holds for each $K \in \mathcal{K}$ where the constant $C = C(K)$ may depend on K .

There are two common ways to describe compact classes in a function space X. The first is through some uniform smoothness of the elements. For example, the unit ball $K = U(Y)$ of a smoothness space Y of X is a typical way of obtaining a compact

subset of X. We say that Y is compactly embedded in X if each finite ball in Y is a compact subset of X.

The classical smoothness spaces are the Sobolev and Besov spaces. The Sobolev space $W^s(L_q) = W^s(L_q(\Omega))$, $\Omega \subset \mathbb{R}^d$, consists of all functions u which have smoothness of order s in L_q. In the case $s = k$ is an integer and $1 \le q \le \infty$ this simply means that u and all its (weak) derivatives are in L_q. There are generalizations of this definition to arbitrary $s, q > 0$. The Besov spaces $B^s_\lambda(L_q)$ are also smoothness spaces of order s in L_q but they involve another parameter λ which makes subtle distinctions among these spaces. They are similar to but generally different from the Sobolev spaces. The Sobolev embedding theorem describes the smoothness spaces which are embedded in a space $X = L_p(\Omega)$ provided the domain Ω has sufficient smoothness (a C^1 smooth boundary is more than enough). This embedding theorem has a simple geometrical interpretation. We identify each space $W^s(L_q)$ (likewise $B^s_\lambda(L_q)$) with the point $(1/q, s)$ in the upper right quadrant of $\mathbb{R}^2$. Given a value $p \in (0, \infty]$, the line $s = d/q - d/p$ is called the critical line for embedding into $L_p(\Omega)$. Any Sobolev or Besov space corresponding to a point above this line is compactly embedded into $L_p(\Omega)$. Any point on or below the line is not compactly embedded into $L_p(\Omega)$. For example, if $s > d/q$, then the Sobolev space $W^s(L_q(\Omega))$ is compactly embedded into the space $C(\Omega)$ of continuous functions on Ω.

A second way to describe compact spaces is through approximation. For example, suppose that $X_n \subset X$, $n = 1, 2, \ldots$, is a sequence of linear spaces of dimension n. Then each of the approximation classes $\mathcal{A}^r$, $r > 0$, describes compact subsets of X: any finite ball in $\mathcal{A}^r$ (with the norm defined by (2.7)) gives a compact subset of X. In the same way, the approximation classes $\mathcal{A}^r$, $r > 0$, for standard methods of nonlinear approximation also give compact subsets of X.

Given the wide range of compact sets in X, it would be too much to ask that an algorithm be universal for the collection of all compact subsets of X. However, universality for large families would be reasonable. For example, if our approximation takes place in $X = L_p(\Omega)$, we could ask that the algorithm be universal for the collection $\mathcal{K}$ of all finite balls in all the Sobolev and Besov spaces with smoothness index $0 < s \le S$ that compactly embed into X. Here S is arbitrary but fixed. This would be a reasonable goal for an algorithm. There are approximation procedures that have this property. Namely, the two nonlinear methods (i) n-term wavelet approximation restricted to trees, and (ii) adaptive piecewise polynomial approximation described in the following section have this property.

In many settings, it is more comfortable to consider optimality over classes described by approximation. Suppose that we have some approximation procedure (linear or nonlinear) in hand. Then, the set $\mathcal{K} := \{B(\mathcal{A}^r) : 0 < r \le R\}$ of all finite balls of the $\mathcal{A}^r$, is a collection of compact sets and we might ask the algorithm to be universal on this collection. Notice that this collection depends very much on the given approximation procedure, and in a sense the choice of this approximation procedure is steering the form of the algorithms we are considering. Since most often, algorithms are designed on the basis of some approximation procedure, finite balls in

approximation classes are natural compact sets to consider.

In the setting of a specific approximation process, we can carry the notion of optimality even further. If $E_n(u)$ denotes the error in approximating u by the approximation procedure, then we say that the algorithm $\mathcal{A} = (A_n)$ is *instance-optimal* with constant $C > 0$ if

$$E(u, A_n) \le C E_n(u) \quad \text{for all } u \in X. \tag{3.5}$$

In other words, the algorithm, in spite of having only partial knowledge about u, approximates, up to a constant, as well as the best approximation. Instance-optimal is a stronger notion than universality on approximation classes. Consider for example n term approximation from a dictionary $\mathcal{D}$. Then $E_n(u)$ on the right side of (3.5) is the error $\sigma_n(u)$ of n term approximation. Knowing that (3.5) is valid, we conclude immediately that the algorithm is universal on the class of approximation spaces $\mathcal{A}^r$, $r > 0$. The choice of the dictionary plays a critical role in defining these notions of optimality.

In summary, we have two possible ways to evaluate the performance of an algorithm. The first is to test its behavior over classes which leads to the notion of optimal, near optimal, and universal. If we consider algorithms based on a specific approximation process, then we can ask for the finer description of optimality described as instance-optimal.

4. Two important examples

Before returning to our main subject of optimal computation, it will be useful to have some concrete approximation processes in hand. We consider two examples of approximation systems that are used frequently in computation and serve to illustrate some basic principles.

4.1. Wavelet bases. A univariate *wavelet* is a function $\psi \in L_2(\mathbb{R})$ whose shifted dilates

$$\psi_{j,k}(x) := 2^{j/2}\psi(2^j x - k), \quad j, k \in \mathbb{Z}, \tag{4.1}$$

are a basis for $L_2(\mathbb{R})$. In the case that the functions (4.1) form a complete orthonormal system we say that ψ is an *orthogonal wavelet*. The simplest example is the Haar orthogonal wavelet

$$H(x) := \chi_{[0,1/2)} - \chi_{[1/2,1)} \tag{4.2}$$

where χ_A the indicator function of a set A. Although the Haar function was prominent in harmonic analysis and probability, it was not heavily used in computation because the function H is not very smooth and also the Haar system has limited approximation capacity. It was not until the late 1980s that it was discovered (Meyer [50] and Daubechies [26]) that there are many wavelet functions ψ and they can be constructed to meet most numerical needs. The most popular wavelets are the compactly supported orthogonal wavelets of Daubechies [26] and the biorthogonal wavelet of Cohen

and Daubechies [22]. They can be constructed from the framework of multiresolution analysis as described by Mallat [48]. Wavelet bases are easily constructed for $\mathbb{R}^d$ and more generally for domains $\Omega \subset \mathbb{R}^d$. There are several books which give far reaching discussions of wavelet decompositions (see e.g. [51] for wavelets and harmonic analysis, [27] for wavelet constructions, [49] for wavelets in signal processing, and [16] for wavelets in numerical PDEs.).

We shall denote a wavelet basis by $\{\psi_\lambda\}_{\lambda\in\Gamma}$. In the case of $\mathbb{R}^d$ or domains $\Omega \subset \mathbb{R}^d$, the index λ depends on three parameters (j, k, e). The integer j gives the dyadic level of the wavelet as in (4.1). The multi-integer $k = (k_1, \dots, k_d)$ locates the wavelet in space (it is associated to the point $2^{-j}k$). The index e corresponds to a vertex of the unit cube $[0, 1]^d$ and describes the gender of the wavelet. We can also think of the wavelets as indexed on the pairs I, $\mathbf{e}$ where $I = 2^{-j}(k + [0, 1]^d)$ is a dyadic cube. In this way, the wavelets are associated to a tree of dyadic cubes. This tree structure is important in computation.

We will always suppose that the wavelets ψ_λ are compactly supported. Each locally integrable function has a wavelet decomposition

$$f = \sum_{\lambda\in\Gamma} f_\lambda \psi_\lambda = \sum_{(I,e)} f_{I.e}\psi_{I,e}. \tag{4.3}$$

The wavelet system is an unconditional basis for L_p and many function spaces such as the Sobolev and Besov spaces.

Let Ω be a domain in $\mathbb{R}^d$ and $\{\psi_\lambda\}_{\lambda\in\Gamma}$ be an orthogonal (or more generally a biorthogonal) wavelet system for $L_2(\Omega)$. The nonlinear spaces Σ_n and their corresponding approximation spaces $\mathcal{A}^r((\psi_\lambda), L_2(\Omega))$ were already introduced in §2.2. These spaces are closely related to classical smoothness spaces. Consider for example the space $\mathcal{A}^r$ which as we know is identical with the space of functions whose wavelet coefficients are weak ℓ_p with $p = (r + 1/2)^{-1}$. While this is not a Besov space, it is closely related to the space $B_p^{rd}(L_p)$ which is contained in $\mathcal{A}^r$. This latter Besov space is characterized by saying that the wavelet coefficients are in ℓ_p.

General n-term wavelet approximation cannot be used directly in computational algorithms because the largest wavelet coefficients could appear at any scale and it is impossible to implement a search over all dyadic scales. There are two natural ways to modify n-term approximation to make it numerically realizable. The first is to simply restrict n-term approximation to dyadic levels $\le a \log n$ where $a > 0$ is a fixed parameter to be chosen depending on the problem at hand. Notice that the number of wavelets living at these dyadic levels does not exceed $C2^{ad\log n} = Cn^{ad}$. The larger the choice of a then the more intensive will be the computation.

The second way to numerically implement the ideas of n-term approximation in the wavelet setting is to take advantage of the tree structure of wavelet decompositions. Given a dyadic cube I, its *children* are the 2^d dyadic subcubes $J \subset I$ of measure $2^{-d}|I|$ and I is called the *parent* of these children. We say that T is a tree of dyadic cubes if whenever $J \in T$ with $|J| < 1$, then its parent is also in T. We define $\mathcal{T}_n$ to be

the collection of all trees of cardinality $\leq n$. We define Σ_n^t as the set of all functions

$$S = \sum_{I \in T} \sum_{e \in E_I} c_I^e \psi_I^e, \quad \#T \leq n, \tag{4.4}$$

where $E_I = E'$ if $|I| = 1$ and $E_I = E$ otherwise. Replacing Σ_n by Σ_n^t in (2.9) leads to σ_n^t and the approximation classes $\mathcal{A}^r((\Sigma_n^t))$. Tree approximation is only slightly more restrictive than n-term approximation. For example, if $B_\lambda^s(L_p)$ is a Besov space that compactly embeds into L_q then any function in this space is in $\mathcal{A}^{s/d}((\Sigma_n^t), L_q)$. This is the same approximation rate as is guaranteed by general n-term approximation for this class.

4.2. Adaptive partitioning. Much of numerical PDEs is built on approximation by piecewise polynomials on partitions of the domain Ω. A partition Π of Ω is a finite collection of sets C_i (called *cells*), $i = 1, \ldots, N$, whose interiors are pairwise disjoint and union to Ω. We have already met the partitions $\mathcal{D}_j$ of the domain $\Omega = [0, 1]^d$ into dyadic cubes.

The typical way of generating partitions is through a *refinement rule* which tells how a cell is to be subdivided. In the dyadic subdivision case, the cells are dyadic cubes and if a cell I in the partition is subdivided then it is replaced by the set $\mathcal{C}(I)$ of its children.

There are many possible refinement strategies. For simplicity, we discuss only the additional case when Ω is a polygonal domain in $\mathbb{R}^2$ and the cells are triangles. We begin with an initial triangulation Π_0. If any triangle Δ is to be refined then its children consist of $a \geq 2$ triangles which form a partition of Δ. We shall assume that the refinement rule is the same for each triangle and thus a is a fixed constant. The refinement rule induces an infinite tree T^* (called the *master tree*) whose nodes are the triangles that can arise through the refinement process.

The refinement level j of a node of T^* is the smallest number of refinements (starting from Π_0) to create this node. We denote by T_j the proper subtree consisting of all nodes with level $\leq j$ and we denote by Π_j the partition corresponding to T_j which is simply all $\Delta \in T_j$ of refinement level j, i.e., the leaves of T_j. The partitions Π_j, $j = 0, 1, \ldots$, we obtain in this way are called *uniform partitions*. The cardinality $\#(\Pi_j)$ of Π_j is $a^j \#(\Pi_0)$.

Another way of generating partitions is by refining some but not all cells. One begins with the cells in Π_0 and decides whether to refine $\Delta \in \Pi_0$ (i.e. subdivide Δ). If Δ is subdivided then it is removed from the partition and replaced by its children. Continuing in this way, we obtain finite partitions which are not necessarily uniform. They may have finer level of refinements in some regions than in others. Each such partition Π can be identified with a finite subtree $T = T(\Pi)$ of the master tree T^*. The cardinalities of Π and $T(\Pi)$ are comparable.

Given a partition Π, let us denote by $\mathcal{S}_k(\Pi)$ the space of piecewise polynomials

of degree k that are subordinate to Π. Each $S \in \mathcal{S}_k\Pi$) can be written

$$S = \sum_{I \in \Pi} P_I \chi_I, \tag{4.5}$$

where P_I is a polynomial of degree $\leq k$. The functions in $\mathcal{S}_k(\Pi)$ are not continuous. In many applications, one is interested in subspaces of $\mathcal{S}_k(\Pi)$ obtained by imposing some global smoothness conditions.

We fix $k \geq 0$ and some norm $\|\cdot\| = \|\cdot\|_X$ where X is an L_p or Sobolev space. We consider two types of approximation. The first corresponds to uniform refinement and gives the error

$$E_n(u) := E_{n,k}(u) := \inf_{S \in \mathcal{S}_k(\Pi_n)} \|u - S\|, \quad n = 0, 1, \ldots. \tag{4.6}$$

This is a form of linear approximation.

To describe an alternative to linear approximation, we let $\mathcal{P}_n$ denote the set of all partitions of size $\leq n$ obtained by using successive refinements. Each partition in $\mathcal{P}_n$ corresponds to a finite tree T contained in the master tree. We define $\Sigma_{n,k}$ as the union of all the spaces $\mathcal{S}_k(\Pi)$, $\Pi \in \mathcal{P}_n$, and the approximation error $\sigma_n(u) := \sigma_{n,k}(u)$ as usual (see (2.9)). This is a form of nonlinear approximation. Its advantage over fixed partitions is that given u, we have the possibility to refine the partition only where u is not smooth and keep it coarse where u is smooth. This means we should be able to meet a given approximation tolerance with a fewer number of cells in the partition. This is reflected in the following results. For either of the two refinement settings we are describing, approximation from $\Sigma_{n,k}$ is very similar to wavelet approximation on trees. For example, if the approximation takes place in an L_q space, then any Besov or Sobolev classes of smoothness order s that compactly embeds into L_q will be contained in $\mathcal{A}^{s/d}((\Sigma_{n,k}), L_q)$ (see [8] and [32]).

In numerical implementations of nonlinear partitioning, we need a way to decide when to refine a cell or not. An *adaptive algorithm* provides such a strategy typically by using local error estimators that monitor the error $e(I)$ between u and the current approximation on a given cell I. Constructing good error estimators in the given numerical setting is usually the main challenge in adaptive approximation.

5. The Sensing Problem

The Sensing Problem is a good illustration of the concepts of optimality that we have introduced. We are given a budget of n questions we can ask about u. These questions are required to take the form of asking for the values $\lambda_1(u), \ldots, \lambda_n(u)$ of linear functionals λ_j, $j = 1, \ldots, n$. We want to choose these functionals to capture the most information about u in the sense that from this information we can approximate u well. The sensing problem is most prominent in signal processing. Analog signals are time dependent functions. A sensor might sample the function through the linear

functionals λ_j, $j = 1, \ldots, n$, and record quantizations of these samples for later processing.

We begin the discussion by assuming that the functions u we wish to sense come from a space $X = L_p(\Omega)$ with $\Omega \subset \mathbb{R}^d$ and $1 \leq p \leq \infty$ and that we will measure the error of approximation in the norm $\|\cdot\| := \|\cdot\|_X$ for this space. An algorithm $\boldsymbol{A} = (A_n)$ takes the following form. For each $n = 1, 2, \ldots$, we have an *encoder* Φ_n which assigns to $u \in X$ the vector

$$\Phi_n(u) := (\lambda_1^n(u), \ldots, \lambda_n^n(u)) \in \mathbb{R}^n, \tag{5.1}$$

where the λ_j^n, $j = 1, \ldots, n$, are fixed linear functionals on X. The mapping $\Phi_n \colon X \to \mathbb{R}^n$ is linear and the vector $\Phi_n(u)$ is the information the sensor will extract about u. Note that we allow these questions to change with n. Another possibility is to require that these questions are progressive which means that for $n + 1$ we simply add one additional question to our current set. We will also need a *decoder* Δ_n which says how to construct an approximation to u from this given data. Thus, Δ_n will be a (possibly nonlinear) mapping from $\mathbb{R}^n$ back into X. Our approximation to u then takes the form

$$A_n(u) := \Delta_n(\Phi_n(u)), \quad n = 1, 2, \ldots. \tag{5.2}$$

Given a vector $y \in \mathbb{R}^n$, the set of functions $\mathcal{F}(y) = \{u \in X : \Phi_n(u) = y\}$ all have the same sensing information and hence will all share the same approximation $A_n(u) = \Delta_n(y)$. If u_0 is any element of $\mathcal{F}(y)$, then

$$\mathcal{F}(y) = u_0 + \mathcal{N}, \tag{5.3}$$

where $\mathcal{N} := \mathcal{N}_n := \mathcal{F}(0)$ is the null space of Φ_n. The structure of this null space is key to obtaining meaningful results.

We have emphasized that in comparing algorithms, we have two possible avenues to take. The one was optimality over classes of functions, the other was instance-optimal. If K is a compact subset of X and Φ_n is an encoder then

$$\bar{\Delta}_n(y) := \operatorname*{Argmin}_{\bar{u} \in X} \sup_{u \in \mathcal{F}(y) \cap K} \|u - \bar{u}\| \tag{5.4}$$

is an optimal decoder for Φ_n on K. Notice that $\bar{\Delta}_n$ is not a practical decoder, its only purpose is to give a benchmark on how well Φ_n is performing. This also leads to the optimal algorithm on the class K which uses the encoder

$$\Phi_n^* := \operatorname*{Argmin}_{\Phi_n} \sup_{u \in K} \|u - \bar{\Delta}_n(\Phi_n(u))\| \tag{5.5}$$

together with the optimal decoder $\bar{\Delta}_n$ associated to Φ_n^* and gives the optimal error

$$E_n^*(K) = \sup_{u \in K} \|u - \bar{\Delta}_N(\Phi_n^*(u))\|. \tag{5.6}$$

We have used the asterisk to distinguish E_n^* from the linear approximation error E_n.

$E_n^*(K)$ is essentially related to the Gelfand n-width $d^n(K)$ of K (see [47], [54] for the definition and properties of Gelfand widths). For example, if $K = -K$ and $K + K \subset C_0 K$, then $d^n(K) \leq E_n^*(K) \leq C_0 d^n(K)$. Gelfand widths of classical classes of functions such as unit balls of Besov and Sobolev spaces are known. The deepest results in this field are due to Kashin [44] who used probabilistic methods to find optimal sensing functionals. In Kashin's constructions, the problems are discretized and the discrete problems are solved using certain random matrices. We shall return to these ideas in a moment when we turn to discrete compressed sensing.

The usual models for signals are band-limited functions. A typical function class consists of all functions $u \in L_2$ whose Fourier transform vanishes outside of some interval $[-A\pi, A\pi]$ with $A > 0$ fixed. The famous Shannon sampling theorem says that sampling the signal u at the points n/A contains enough information to exactly recover u. The Shannon theory is the starting point for many Analog to Digital encoders. However, these encoders are severely taxed when A is large. In this case, one would like to make much fewer measurements. Since the Shannon sampling is optimal for band-limited signals, improved performance will require the introduction of new (realistic) model classes for signals. One model in this direction, that we will utilize, is to assume that the signal can be approximated well using n terms of some specified dictionary $\mathcal{D}$ of waveforms. For example, we can assume that u is in one of the approximation classes $\mathcal{A}^r((\Sigma_n), X)$ for n-term approximation by the elements of $\mathcal{D}$. We will consider the simplest model for this problem where $\mathcal{D} = B$ is an orthogonal basis B. When this basis is the wavelet basis then we have seen that $\mathcal{A}^r$ is related to Besov smoothness, so the case of classical smoothness spaces is included in these models.

The obvious way of approximating functions in these classes is to retain only the largest terms in the basis expansion of u. However, this ostensibly requires examining all of these coefficients or in other words, using as samples all of the expansion coefficients. Later we would discard most or many of these coefficients to obtain an efficient decomposition of u using only a few terms. A central question in the Sensing Problem is whether one could avoid taking such a large number of sensing samples and still retain the ability to approximate u well. Of course, we have to deal with the fact that we do not know which of these coefficients will be large. So the information will have to, in some sense, tell us both the position of the large coefficients of u and their numerical value as well.

5.1. Discrete compressed sensing. To numerically treat the sensing problem, we have to make it finite – we cannot deal with infinite expansions or computations with infinite length vectors. Discretization to finite dimensional problems is also used to prove results for function spaces. We will therefore restrict our attention to the following discrete sensing problem. We assume our signal is a vector x in $\mathbb{R}^N$ where N is large. We are given a budget of n linear measurements of x (the application of n linear functionals to x) and we ask how well we can recover x from this information.

The previously mentioned results of Kashin [44] show that using n randomly generated functionals will carry almost as much information as knowing the positions and values of the n largest coordinates of x. However, Kashin's results were never implemented into practical algorithms. It was not until the exciting results of Candès, Romberg, and Tao [12] applied to tomography that the door was opened to view the power of random sampling. This was followed by the fundamental papers [13, 15, 34] which addressed how to build practical encoder/decoders and proved the first results on their provable performance. There was a related development in the computer science community which used random measurements to sketch large data sets which we shall say a little more about later.

The discrete sensing problem can be described by an $n \times N$ matrix Φ. Row i of Φ corresponds to a vector that represents the linear functional λ_i. Thus, sensing with these linear functionals is the same as evaluating $\Phi(x) = y$. The vector y which lives in the lower dimensional space $\mathbb{R}^n$ represents the information we have about x. We are interested in how well we can recover x from this information. As we have noted, the exact way we recover x (or an approximation to x) from the information y is a significant component of the problem. A decoder for Φ is a (possibly nonlinear) mapping Δ from $\mathbb{R}^n$ to $\mathbb{R}^N$. Given $y = \Phi(x)$, then $\Delta(y)$ is our approximation to x.

In analogy with the continuous case, we can use the ℓ_p norms, defined in (2.3), to measure error. Then, the error associated to a particular algorithm A_n built on a matrix Φ and a particular decoder Δ is given by

$$E(x, A_n)_p := E(x, \Phi, \Delta)_p := \|x - \Delta(\Phi(x))\|_{\ell_p}. \tag{5.7}$$

The approximation on a class K of signals is defined as in (3.3). Let us denote by $E_n^*(K, \ell_p)$ the optimal error on K achievable by any sensing algorithm of order n (we are suppressing the dependence on N). In keeping with the main theme of this paper, let us mention the types of results we could ask of such a sensing/decoding strategy. We denote by Σ_k the space of all k-sparse vectors in $\mathbb{R}^N$, i.e., vectors with support $\leq k$. For any sequence space X, we denote by $\sigma_k(x)_X$ the error in approximating x by the elements of Σ_k in the norm $\|\cdot\|_X$.

I. Exact reconstruction of sparse signals. Given k, we could ask that

$$\Delta(\Phi(x)) = x, \quad x \in \Sigma_k. \tag{5.8}$$

We would measure the effectiveness of the algorithm by the largest value of k (depending on n and N) for which (5.8) is true.

II. Performance on classes K. We have introduced optimality and near optimality of an algorithm on a class K in §3. We will restrict our attention to the following sets K: the unit ball of the approximation space $\mathcal{A}^r((\Sigma_k), \ell_p)$; the unit ball of spaces ℓ_τ and weak ℓ_τ. We recall that the norm on $\mathcal{A}^r((\Sigma_k), \ell_p)$ is equivalent to the weak ℓ_τ norm, $1/\tau = r + 1/p$. As we have noted earlier, the optimal performance of sensing algorithms is determined by the Gelfand width of K. These widths are known for all

of the above classes (see Chapter 14 of [47], especially (5.1) and Notes 10.1 of that chapter). Since, the results are extensive, we mention only one result, for the case $p = 2$, which will orient the reader. For the unit ball $U(\ell_1)$ of ℓ_1 in $\mathbb{R}^N$, we have

$$C_1 \min(\sqrt{\frac{\log(eN/n)}{n}}, 1) \le E_n^*(U(\ell_1), \ell_2) \le C_2\sqrt{\frac{\log(eN/n)}{n}}, \qquad (5.9)$$

with absolute constants $C_1, C_2 > 0$. This is the result of Kashin improved (in the logarithm) by Gluskin. The appearance of the logarithm in (5.9) is the (small) price we pay in doing compressed sensing instead of sampling all coefficients and taking the n largest. We can use theoretical results like (5.9) to gauge the performance of any proposed algorithm.

III. Instance-optimal. Instead of asking for optimal performance on classes, we could ask that the sensing/decoding performs as well as k-term approximation on each $x \in \mathbb{R}^N$. If $\|\cdot\|_X$ is a norm on $\mathbb{R}^N$, we say the sensing encoder/decoder pair Φ/Δ is *instance-optimal* of order k if

$$E(x, \Phi, \Delta)_X \le C_0\sigma_k(x)_X, \quad x \in \mathbb{R}^N, \qquad (5.10)$$

holds for a constant independent of N and n. Given N and n, we want the largest value of k for which (5.10) holds. Since we are dealing with finite-dimensional spaces, the role of the constant C_0 is important.

In each case, some relevant issues are: (i) what are the properties of a matrix Φ that guarantee optimal or near optimal performance, (ii) which decoders work with Φ, (iii) what is the computational complexity of the decoding - how many computations are necessary to compute $\Delta(y)$ for a given y?

In [34], Donoho gave an algorithm which is optimal in the sense of II for $p = 2$ and for the unit ball of ℓ_τ, $\tau \le 1$ (hence it is universal for these sets). His approach had three main features. The first was to show that if a matrix Φ has three properties (called CS1-3), then Φ will be an optimal sensor for these classes. Secondly, he showed through probabilistic arguments that such matrices Φ exist. Finally, he showed that ℓ_1 minimization provides a near-optimal decoder for these classes. Independently Candès and Tao ([14], [15]) developed a similar theory. One of the advantages of their approach is that it is sufficient for Φ to satisfy only a version of the CS1 property and yet they obtain the same and in some cases improved results.

To describe the Candès–Tao results, we introduce the following notation. Given an $n \times N$ matrix Φ, and any set $T \subset \{1, \dots, N\}$, we denote by Φ_T the $n \times \#(T)$ matrix formed from these columns of Φ. We also use similar notation for the restriction x_T of a vector from $\mathbb{R}^N$ to T. The matrix Φ is said to have the *restricted isometry property* for k if there is a $0 < \delta_k < 1$ such that

$$(1 - \delta_k)\|x_T\|_{\ell_2} \le \|\Phi_T x_T\|_{\ell_2} \le (1 + \delta_k)\|x_T\|_{\ell_2} \qquad (5.11)$$

holds for all T of cardinality k.

Given the matrix Φ, and any $x \in \mathbb{R}^N$, the vector $y = \Phi(x)$ represents our information about x. As we have noted before, we need a decoding strategy for y. Both Candès–Romberg–Tao and Donoho suggest taking the element $\bar{x}$ that minimizes the ℓ_1 norm over all vectors which share the data y:

$$\bar{x} := \Delta(y) := \operatorname*{Arg\,min}_{z \in \mathcal{F}(y)} \|z\|_{\ell_1}. \tag{5.12}$$

Numerically, the decoding can be performed through linear programming. From our perspective, the main question is how well this encoding/decoding approximates x. The main result of [13] is that if

$$\delta_{3k} + 3\delta_{4k} < 2, \tag{5.13}$$

then

$$\|x - \bar{x}\|_{\ell_2} \le C \frac{\sigma_k(x)_{\ell_1}}{\sqrt{k}}. \tag{5.14}$$

Under the same conditions on Φ and k, the following variant of (5.14) was shown in [18]

$$\|x - \bar{x}\|_{\ell_1} \le C \sigma_k(x)_{\ell_1}. \tag{5.15}$$

By interpolation inequalities, we obtain for $1 \le p \le 2$

$$\|x - \bar{x}\|_{\ell_p} \le C \frac{\sigma_k(x)_{\ell_1}}{k^{1-1/p}}. \tag{5.16}$$

In all these inequalities, we can take C as an absolute constant once we have strict inequality in (5.13).

Before interpreting these results, let us first address the question of whether we have matrices Φ that satisfy (5.13). This is where randomness enters the picture. Consider an $n \times N$ matrix Φ whose entries are independent realizations of the Gaussian distribution with mean 0 and variance 1. Then, with high probability, the matrix Φ will satisfy (5.13) for any $k \le C_0 n / \log(N/n)$. Similar results hold if the Gaussian distribution is replaced by a Bernoulli distribution taking the values ± 1 with equal probability [2]. Here we have returned to Kashin who used such matrices in his solution of the n-width problems. Thus, there are many matrices that satisfy (5.13) and any of these can be used as sensing matrices. Unfortunately, this probabilistic formulation does not give us a vehicle for putting our hands on one with absolute certainty. This leads to the very intriguing question of concrete constructions of good sensing matrices.

Let us now return to our three formulations of optimality.

Exact reproduction of k sparse signals (Case I). If we have a matrix Φ that satisfies (5.11) and (5.13) in hand then the above encoding/decoding strategy gives an optimal solution to the Sensing Problem for the class of k-sparse signals: any signal with support k will be exactly captured by $\Delta(\Phi(x))$. Indeed, in this case $\sigma_k(x)_{\ell_1} = 0$

and therefore this follows from (5.14). Thus, for any $k \le Cn\log(N/n)$, there is a matrix Φ and a decoder Δ given by (5.12) that is optimal for the class of k sparse signals under this restriction on k. However, in this case the range of k is not optimal (the logarithm can be eliminated) as can easily be seen and was pointed out in [3]. For any k, we can create a matrix Φ of size $2k \times N$ that has the exact reproduction property of k sparse signals. For example, for $k = 1$, any two linear functionals $\sum_{i=1}^N q^i x_i$ with two distinct numbers q will have enough information to recover a one-sparse signal exactly.

The key to proving optimality of k-sparse signals is to prove that the null space $\mathcal{N}$ of Φ has no nonzero vector with support $\le 2k$. This is an algebraic property of Φ. Any matrix with this property will solve the optimality for k-sparse vectors. Such matrices are readily seen to exist. For any k and $N \ge 2k$, we can find a set Λ_N of N vectors in $\mathbb{R}^{2k}$ such that any $2k$ of them are linearly independent. The matrix Φ whose columns are the vectors in Λ_N will have the exact reproduction property. There are many examples of such sets Λ_N. For example, if $x_1 < x_2 < \cdots < x_N$ are arbitrary real numbers, then we can take the vectors $v_j \in \mathbb{R}^n$ whose entries are x_j^{i-1}, $i = 1, \dots, n$, which gives a van der Monde matrix. Such matrices are unfortunately very unstable in computation and cannot be used for the other sensing problems.

Optimality for classes K (Case II). To go further and discuss what happens in the case that x is not k-sparse, we assume first that $p = 2$, i.e., we measure error in ℓ_2. From our discussion of the existence of matrices Φ that satisfy (5.11),(5.13), we see that we can take $k = Cn/\log(N/n)$ in (5.14). Since $\sigma_k(x)_{\ell_1} \le \|x\|_{\ell_1}$, we obtain optimality on the class $U(\ell_1)$ (see (5.9)). The inequality (5.16) can be used to obtain similar results for approximation in ℓ_p.

Instance-optimal (Case III). If $\|\cdot\|_X$ is any norm on $\mathbb{R}^N$ and Φ is an $n \times N$ matrix, we say Φ has the *null space property* (NSP) for X of order k if for each $\eta \in \mathcal{N} = \mathcal{N}(\Phi)$,

$$\|\eta\|_X \le C_0 \|\eta_{T^c}\|_X, \quad \#(T) = k, \tag{5.17}$$

where T^c is the complement of T in $\{1, \dots, N\}$ and where $C_0 \ge 1$ is a fixed constant. A sufficient condition for Φ to have a decoder Δ such that the pair Φ/Δ is instance-optimal of order k is that $\mathcal{N}$ have the NSP for X of order $2k$ and a necessary condition is that $\mathcal{N}$ have the NSP for X of order k (see [18]).

In view of (5.15), the probabilistic constructions give instance-optimal sensing for $X = \ell_1$ and $k \le Cn/\log(N/n)$. On the other hand, it can be shown (see [18]) that any matrix which has the null space property for $k = 1$ in ℓ_2 will necessarily have $n \ge N/C_0^2$ rows. This means that in order to have instance-optimal for one sparse vectors in ℓ_2 requires the matrix Φ to have $O(N)$ rows. Therefore, instance-optimal is not a viable possibility for ℓ_2. For ℓ_p, $1 < p < 2$, there are intermediate results where the conditions on k relative to N, n are less severe (see [18]).

One final note about the discrete compressed sensing problem. We have taken as our signal classes the approximation spaces $\mathcal{A}^r$ which are defined by k-term approximation using the canonical basis for $\mathbb{R}^n$. In actuality the probabilistic construction

of sensing matrices works for sparsity measured by approximation using much more general bases. All we need is that the matrix representation of Φ with respect to the new basis also have properties (5.11), (5.13). Thus, the choice of a random sensing matrix Φ will encode sparsity simultaneously in many (most) bases. However, the decoding has to be done relative to the chosen basis. This is sometimes referred to as a universal property of the randomly constructed matrices Φ (see [2] for a more precise discussion of universality).

5.2. Computational issues. Notice that in compressed sensing, we have reduced greatly the number of samples n we need from the signal when compared with thresholding techniques. However, this was at the cost of severely complicating the decoding operator. The decoding by ℓ_1 minimization generally requires polynomial in N machine operations which may be prohibitive in some settings when N is large. Issues of this type are a major concern in Theoretical Computer Science (TCE) and some of the work in TCE relates to the sensing problem. For example, there has been a fairly long standing program in TCE , going back to the seminal work of Alon, Matias and Szegedy [1], to efficiently sketch large data sets (see also Henzinger, Raghavan and Rajaopalan [41]). The emphasis has been to treat streaming data with efficient computation measured not only by the number of computations but also the space required in algorithms. Streaming algorithms call for different encoders. The sensing matrix Φ needs to be constructed in a small amount of time. So their constructions typically use less randomness and sometimes are possible using coding techniques such Kerdoch codes or, more generally, Reed–Muller codes.

In some settings, the flavor of the results is also different. Rather than construct one sensing matrix Φ, one deals with a stochastic family $\Phi(\omega)$ of $n \times N$ matrices with ω taking values in some probability space Ω. An algorithm proceeds as follows. Given x, one takes a draw of an $\omega \in \Omega$ according to the probability distribution on Ω (this draw is made independent of any knowledge of x). The information recorded about x is then $\Phi(\omega)x$. There is a decoder $\Delta(\omega)$ which when applied to the information $\Phi(\omega)x$ produces the approximation $x(\omega) := \Delta(\omega)\Phi(\omega)$.

Changing the problem by demanding only good approximation in probability rather than with certainty allows for much improvement in numerical performance of decoding in the algorithm (see [36], [37], [38]). For example, in some constructions the decoding can be done using greedy algorithms with $P(n \log N)$ operations where P is a polynomial. This is a distinct advantage over decoding by ℓ_1 minimization when n is small and N is large. Also, now the spectrum of positive results also improves. For example, when calling for deterministic bounds on the error, we saw that instance-optimal in ℓ_2 is not possible (even for one-term sparsity) without requiring $n \geq c_0 N$. In this new setting, we can obtain instance-optimal performance for k with high probability even in ℓ_2 (see [18] and [24]).

6. The learning problem

This problem differs from the Data Fitting Problem in that our measurements are noisy. We shall assume that $X = [0, 1]^d$ (for simplicity) and that $Y \subset [-M, M]$. This assumption implies that f_ρ also takes values in $[-M, M]$. The measure ρ factors as the product

$$d\rho(x, y) = d\rho_X(x, y)d\rho(y|x) \tag{6.1}$$

of the marginal ρ_X and the conditional $\rho(y|x)$ measures.

Let $Z := X \times Y$. Given the data $z \in Z^n$, the problem is to find a good approximation f_z to f_ρ. We shall call a mapping $\boldsymbol{E}_n$ that associates to each $z \in Z^n$ a function f_z defined on X to be an *estimator*. By an *algorithm*, we shall mean a family of estimators $\{\boldsymbol{E}_n\}_{n=1}^\infty$. To evaluate the performance of estimators or algorithms, we must first decide how to measure the error in the approximation of f_ρ by f_z. The typical candidates to measure error are the $L_p(X, \rho_X)$ norms:

$$\|g\|_{L_p(X,\rho_X)} := \begin{cases} \left(\int_X |g(x)|^p d\rho_X\right)^{1/p}, & 1 \le p < \infty, \\ \operatorname*{esssup}_{x\in X} |g(x)|, & p = \infty. \end{cases} \tag{6.2}$$

Other standard choices in the statistical literature correspond to taking measures other than ρ_X in the L_p norm, for instance the Lebesgue measure. We shall limit our discussion to the $L_2(X, \rho_X)$ norm which we shall simply denote by $\|\cdot\|$. This is the most common and natural measurement for the error. Note that since we do not know ρ, we do not know this norm precisely. However, this will not prevent us from obtaining estimates relative to this norm.

The error $\|f_z - f_\rho\|$ depends on z and therefore has a stochastic nature. As a result, it is generally not possible to say anything about this error for a fixed z. Instead, we can look at behavior in probability as measured by

$$\rho^n\{z : \|f_\rho - f_z\| > \eta\}, \quad \eta > 0, \tag{6.3}$$

or in expectation

$$E_{\rho^n}(\|f_\rho - f_z\|) = \int_{Z^n} \|f_\rho - f_z\| \, d\rho^n, \tag{6.4}$$

where the expectation is taken over all realizations z obtained for a fixed n and ρ^n is the n-fold tensor product of ρ.

We can define optimal, near optimal, and universal algorithms as in §3. The starting point of course are the compact classes $K \subset L_2(X, \rho_X)$. For each such compact set K, we have the set $\mathcal{M}(K)$ of all Borel measures ρ on Z such that $f_\rho \in K$. There are two notions depending on whether we measure performance in expectation or probability. We enter into a competition over all estimators $\boldsymbol{E}_n : z \to f_z$ and define

$$e_n(K) := \inf_{\boldsymbol{E}_n} \sup_{\rho\in\mathcal{M}(K)} E_{\rho^n}(\|f_\rho - f_z\|_{L_2(X,\rho_X)}), \tag{6.5}$$

and

$$\boldsymbol{AC}_n(K,\eta) := \inf_{\boldsymbol{E}_n} \sup_{\rho\in\mathcal{M}(K)} \rho^n\{z : \|f_\rho - f_z\| > \eta\}. \tag{6.6}$$

As emphasized by Cucker and Smale [25] estimates in probability are to be preferred since they automatically imply estimates in expectation (by integrating with respect to $d\rho^n$). However, for the sake of simplicity of this presentation, most of our remarks will center around estimates in expectation.

Since we do not know the measure ρ_X, the compact subsets K of $L_2(X, \rho_X)$ are also not completely known to us. One way around this is to consider only compact subsets of $C(X)$ since these will automatically be compact in $L_2(X, \rho_X)$. Thus, classical spaces such as Sobolev and Besov classes which embed compactly into $C(X)$ are candidates for our analysis. A second, more robust, approach, is to consider the compact sets defined by an approximation process as described in §3.

The problem of understanding optimal performance on a compact set $K \subset L_2(X, \rho_X)$ takes a different turn from the analysis in our other estimation problems because the stochastic nature of the problem will prevent us from approximating f_ρ to accuracy comparable to best approximation on classes. This means that understanding optimality requires the establishment of both lower and upper bounds on convergence rates. There are standard techniques in statistics based on Kuhlbach–Leibler information and Fano inequalities for establishing such lower bounds. In [28] a lower bound for the performance of an algorithm on K was established using a slight modification of Kolmogorov entropy. This result can be used to show that whenever K is a finite ball in a classical Besov or Sobolev class of smoothness order s which compactly embed into $C(X)$, then the optimal performance attainable by any algorithm is

$$e_n(K) \geq c(K)n^{-\frac{s}{2s+d}}, \quad n = 1, 2, \ldots. \tag{6.7}$$

An algorithm $(\boldsymbol{E}_n)$ is near optimal in expectation on the class K if for data z of size n, the functions f_z produced by the estimator $\boldsymbol{E}_n$ satisfy

$$E_{\rho^n}(\|f_\rho - f_z\|) \leq C(K)e_n(K), \tag{6.8}$$

whenever $f_\rho \in K$. It is often the case that estimation algorithms may miss near optimal performance because of a $\log n$ factor. We shall call such algorithms *quasi-optimal*.

There are various techniques for establishing upper bounds comparable to the best lower bounds. On a very theoretical level, there are the results of Birgé and Massart [10] which use ϵ nets for Kolmogorov entropy to establish upper bounds. While these results do not lead to practical algorithms, they do show that optimal performance is possible. Practical algorithms are constructed based on specific methods of linear or nonlinear approximation. The book [40] gives an excellent accounting of the state of the art in this regard (see Theorems 11.3 and 13.2). Let us point out the general approach and indicate some of the nuances that arise.

Suppose that we have chosen a sequence (Σ_m) of spaces Σ_m (linear or nonlinear of dimension m) to be used for the approximation of f_ρ from our given data z. How should we define our approximation? Since all that we have available to us is the data z, the natural choice for an approximation from Σ_m is the minimizer of the empirical risk

$$f_{z,\Sigma_m} := \operatorname*{Argmin}_{f\in\Sigma_m} \mathcal{E}_z(f), \quad \text{with } \mathcal{E}_z(f) := \frac{1}{n}\sum_{j=1}^{n}(y_j - f(x_j))^2. \tag{6.9}$$

In other words, f_{z,Σ_m} is the best approximation to $(y_j)_{j=1}^n$ from Σ_m in the empirical norm

$$\|g\|_z^2 := \frac{1}{n}\sum_{j=1}^{n}|g(x_j)|^2. \tag{6.10}$$

A key issue is how should we choose the dimension m. Choosing m too large results in fitting the noise (to be avoided) while choosing m too small reduces the approximation effectiveness. If we knew that f_ρ was in the approximation class $\mathcal{A}^s((\Sigma_m), L_2(X,\rho_X))$ then a choice $m \approx \left(\frac{n}{\log n}\right)^{\frac{1}{2s+1}}$ would result in an algorithm that is quasi-optimal on the class. However, we do not know s and so we need a method to get around this. The common approach is what is called model selection.

Model selection automatically chooses a good value of m (depending on z) by introducing a penalty term. For each $m = 1, 2, \dots, n$, we have the corresponding function f_{z,Σ_m} defined by (6.9) and the empirical error

$$E_{m,z} := \frac{1}{n}\sum_{j=1}^{n}(y_j - f_{z,\Sigma_m}(x_j))^2. \tag{6.11}$$

Notice that $E_{m,z}$ is a computable quantity. In complexity regularization, one typically chooses a value of m by

$$m^* := m^*(z) := \operatorname*{Argmin}_{1\le m\le n}\left[E_{m,z} + \frac{\kappa m\log n}{n}\right], \tag{6.12}$$

with the parameter κ to be chosen (it will govern the range of s that is allowed). Then, one defines the estimator

$$\hat{f}_z := f_{z,\Sigma_{m^*}}. \tag{6.13}$$

Here is an important remark. One does not use $\hat{f}_z$ directly as the estimator of f_ρ since it is difficult to give good estimates for its performance. The main difficulty is that this function may be large even though we know that $|f_\rho| \le M$. Given this knowledge about f_ρ, it makes sense to post-truncate $\hat{f}_z$ and this turns out to be crucial in practice. For this, we define the truncation operator

$$T_M(x) := \min(|x|, M)\,\operatorname{sign}(x) \tag{6.14}$$

for any real number x and define

$$f_z := T_M(\hat{f}_z) \tag{6.15}$$

as our estimator to f_ρ.

One can show that under quite general conditions, the estimator (6.15) is quasi-optimal on the approximation classes $\mathcal{A}^s((\Sigma_m), L_2(X, \rho_X))$. The question arises as to which approximation process (Σ_m) should be employed. We mention some of the main issues in making this choice. First note that each approximation scheme has its own approximation classes. Without any additional knowledge about f_ρ there is from the viewpoint of approximation rates no obvious preference of one approximation process over another except that nonlinear methods are preferable to linear methods since the resulting approximation classes for nonlinear methods are larger.

A major issue, especially in implementing nonlinear methods, is computational cost. The larger the nonlinear process (e.g. the larger the dictionary in n-term approximation), then the more computation needed for minimization in the model selection (6.12). This factor is one of the major concerns in choosing the approximation scheme. It is especially critical for high space dimension d. We mention a couple of methods that can address these computational issues and relate to the methods of approximation introduced in this lecture.

Suppose we use a dictionary $\mathcal{D}$ of size n^a and employ model selection. This would require examining all m dimensional subspaces formed by elements of the dictionary for each $m = 1, 2, \ldots, n$. While the number of computations can possibly be reduced by using the fact that the data size is n, it will still involve solving $O(2^n)$ such least squares problems. This computation can be reduced considerably by using greedy algorithms. Through such an algorithm, we can find, with the evaluation of $O(n^{a+1})$ inner products, the greedy sequence $v_1, \ldots, v_n$ of the dictionary that provides near optimal empirical approximation with respect to the approximation classes described in our discussion of greedy algorithms (see [5]). We can then do model selection over the n subspaces $V_m := \text{span}\{v_1, \ldots, v_m\}$, $m = 1, \ldots, n$, to find the approximation f_z. Thus, the model selection is done over n, rather than $O(2^n)$, subspaces, after the implementation of the greedy algorithm. The price we pay for this increased efficiency is that the approximation classes for greedy algorithms are in general smaller than those for m-term approximation. This means that in general we may be losing approximation efficiency.

Another setting in which a model selection based on n-term approximation can be improved is in the use of adaptive approximation as described in §4. Here, one takes advantage of the tree structure of such adaptive methods. For a given data set z of size n, one associates to each node of the master tree T^* the empirical least squares error on the cell associated to the node. There are fast algorithms known as CART algorithms (see [33]) for assimilating the local errors and pruning the master tree to implement model selection. Another alternative put forward in [6] is to utilize empirical thresholding in a very similar fashion to wavelet tree approximation. Another advantage of adaptive algorithms is that they can be implemented on-line.

Adding one or several new data points does not require re-solving the entire empirical minimization problem but only to do tree updates. Moreover, in some cases, the analysis of adaptive algorithms can be made in probability.

7. Concluding remarks

Because of space limitations, we were only able to discuss the Sensing Problem and the Learning Problem in any detail. We will make a few brief remarks to direct the reader interested in some of the other problems.

The Data Fitting Problem is a special case of the optimal recovery problem. An excellent resource for results on optimal recovery is the survey [52] and the articles referenced in that survey. This problem, as well as aspects of the sensing problem, are also treated in the wealthy literature in Information Based Complexity (see [57] and [58] for a start) where the approach is very similar to our discussion of optimality.

The encoding problem is a main consideration in image processing and information theory. Our approach of deterministic model classes is in contrast to the usual stochastic models used in information theory. While stochastic models still dominate this field, there are a growing number of treatments addressing the image compression problem from the deterministic viewpoint. Optimal algorithms for Besov and Sobolev classes for the encoding problem can be obtained by employing wavelet thresholding and quantization (see [17]). More advanced methods of image compression model images by approximation classes based on other forms of approximation such as curvelets [11] and wedgeprints [55].

The computation problem is the most dominant area of numerical analysis. The most advanced and satisfying theory appears in the solution of elliptic equations with error measured in the energy norm. For model classes described by linear approximation (for example, classical Finite Element Methods based on piecewise polynomial approximations on fixed partitions), the Galerkin solutions relative to these spaces provide optimal algorithms. Much less is known for nonlinear methods. For model classes based on wavelet tree approximation, near optimal algorithms (in terms of total number of computations) have been given in [19], [20], [21]. For adaptive finite element methods, similar results have been given for simple model problems (the Poisson problem) in [7] by building on the results in [35], [53].

References

[1] Alon, N., Matias, Y., and Szegedy, M., The space complexity of approximating the frequency moments. In *Proceedings of the twenty-eighth annual ACM symposium on the theory of computing* (Philadelphia, PA, 1996), ACM, New York 1996, 20–29.

[2] Baraniuk, R., Davenport, M., DeVore, R., and Wakin, M., The Johnson-Lindenstrauss lemma meets compressed sensing. Preprint

[3] Baron, D., Wakin, M., Duarte, M., Sarvotham, S., and Baraniuk, R., Distributed Compressed Sensing. Preprint.

[4] Baron, D., Universal approximation bounds for superposition of n sigmoidal functions. *IEEE Trans. Inform. Theory* **39** (1993), 930–945

[5] Barron, A., Cohen, A., Dahmen, W., and DeVore, R., Approximation and learning by greedy algorithms. Preprint.

[6] Binev, P., Cohen, A., Dahmen, W., and Temlyakov, V., Universal Algorithms for Learning Theory Part I: piecewise constant functions. *J. Mach. Learn. Res.* **6** (2005), 1297–1321.

[7] Binev, P., Dahmen, W., and DeVore, R., Adaptive Finite Element Methods with Convergence Rates. *Numer. Math.* **97** (2004), 219–268.

[8] Binev, P., Dahmen, W., DeVore, R., and Petrushev, P., Approximation Classes for Adaptive Methods. *Serdica Math. J.* **28** (2002), 391–416.

[9] Binev, P., and DeVore, R., Fast Computation in Adaptive Tree Approximation. *Numer. Math.* **97** (2004), 193–217.

[10] Birgé, L., and Massart, P., Rates of convergence for minimum contrast estimators. *Probab. Theory Related Fields* **97** (1993), 113-150.

[11] Candès, E. J., and Donoho, D. L., Continuous Curvelet Transform: I. Resolution of the Wavefront Set. *Appl. Comput. Harmon. Anal.* **19** (2005), 162–197.

[12] Candès, E. J., Romberg, J., and Tao, T., Robust Uncertainty Principles: Exact Signal Reconstruction from Highly Incomplete Frequency Information. *IEEE Trans. Inform. Theory* **52** (2006), 489–509.

[13] Candès, E., Romberg, J., and Tao, T., Stable signal recovery from incomplete and inaccurate measurements. *Comm. Pure and Appl. Math.* **59** (2006), 1207–1223.

[14] Candès, E., and Tao, T., Decoding by linear programing. *IEEE Trans. Inform. Theory* **51** (2005), 4203–4215.

[15] Candès, E., and Tao, T., Near optimal signal recovery from random projections:universal encoding strategies. Preprint.

[16] Cohen, A., *Numerical Analysis of Wavelet Methods*. Stud. Math. Appl. 32, North Holland, Amsterdam 2003.

[17] Cohen, A., Dahmen, W., Daubechies, I., and DeVore, R., Tree approximation and encoding. *Appl. Comput. Harmon. Anal.* **11** (2001), 192–226.

[18] Cohen, A., Dahmen, W., and DeVore, R., Compressed sensing and k-term approximation. In preparation.

[19] Cohen, A., Dahmen, W., DeVore, R., Adaptive wavelet methods for elliptic operator equations: convergence rates. *Math. Comp.* **70** (2001), 27–75.

[20] Cohen, A., W. Dahmen, DeVore, R., Adaptive wavelet methods II. Beyond the elliptic case. *Found. Comput. Math.* **2**(2002), 203-245.

[21] Cohen, A., Dahmen, W., and DeVore, R., Adaptive Wavelet Schemes for Nonlinear Variational Problems. *SIAM J. Numer. Anal.* **41** (2003), 1785–1823.

[22] Cohen, A., Daubechies, I., and Feauveau, J. C., Biorthogonal bases of compactly supported wavelets. *Comm. Pure Appl. Math.* **45** (1992), 485–560.

[23] Cohen, A., Daubechies, I., and Vial, P., Wavelets and fast transforms on an interval. *Appl. Comput. Harmon. Anal.* **1**(1993), 54–81.

[24] Cormode, G., and Muthukrishnan, S., Towards and algorithmic theory of compressed sensing. Technical Report 2005-25, DIMACS, 2005.

[25] Cucker, F., and Smale, S., On the mathematical foundations of learning theory. *Bull. Amer. Math. Soc.* **39** (2002), 1–49.

[26] Daubechies, I., Orthonormal bases of compactly supported wavelets. *Comm. Pure Appl. Math.* **41** (1988), 909–996.

[27] Daubechies, I., *Ten Lectures on Wavelets*. CBMS-NSF Regional Conf. Ser. in Appl. Math. 61, SIAM, Philadelphia 1992.

[28] DeVore, R., Kerkyacharian, G., Picard, D., and Temlyakov, V., Approximation Methods for Supervised Learning. *Found. Comput. Math.* **6** (2006),3–58.

[29] DeVore, R., and Temlyakov, V., Some remarks on greedy algorithms. *Adv. Comput. Math.* **5** (1996), 173–187.

[30] DeVore, R., Nonlinear approximation. *Acta Numer.* **7** (1998), 51–150.

[31] DeVore, R., and Lorentz, G. G., *Constructive Approximation*. Grundlehren Math. Wiss. 303, Springer-Verlag, Berlin,Heidelberg 1993.

[32] DeVore, R., and Yu, X.-M., Degree of adaptive approximation. *Math. Comp.* **55** (1990), 625–635.

[33] Donoho, D. L, CART and best-ortho-basis: a connection. *Ann. Statist.* **25** (1997), 1870–1911.

[34] Donoho, D., Compressed Sensing. *IEEE Trans. Inform. Theory* **52** (2006), 1289–1306.

[35] Dörfler, W., A convergent adaptive algorithm for Poisson's equation. *SIAM J. Numer. Anal.* **33** (1996), 1106–1124.

[36] Gilbert, A., Kotidis, Y., Muthukrishnan, S., and Strauss, M., How to summarize the universe: Dynamic maintenance of quantiles. *Proc. VLDB*, 2002, 454–465.

[37] Gilbert, A., Guha, S., Kotidis, Y., Indyk, P., Muthukrishnan, S., and Strauss, M., Fast, small space algorithm for approximate histogram maintenance. In *Proceedings of the thiry-fourth annual ACM symposium on the theory of computing*, ACM, New York 2002, 389–398.

[38] Gilbert, A., Guha, S., Indyk, P., Muthukrishnan, S., and Strauss, M., Near-optimal sparse Fourier estimation via sampling. *Proceedings of the thiry-fourth annual ACM symposium on the theory of computing*, ACM, New York 2002, 152–161.

[39] Goldreich, O., Levin, L., A hardcore predicate for one way functions. In *Proceedings of the twenty-first annual ACM Symposium on the Theory of Computing*, ACM, New York 1989, 25–32.

[40] Györfi, L., Kohler, M., Krzyzak, A., and Walk, H., *A Distribution -free Theory of Nonparametric Regression*. Springer Series in Statistics, Springer-verlag, New York 2002.

[41] Henzinger, M., Raghavan, P., and Rajagopalan, S., Computing on data stream. Technical Note 1998-011, Digital systems research center, Palo Alto, May 1998.

[42] Jones, L. K., A simple lemma on greedy approximation in Hilbert spaces and convergence rates for projection pursuit regression and neural network training. *Ann. Statist.* **20** (1992), 608–613.

[43] Johnson, W., and Lindenstrauss, J., Extensions of Lipschitz maps into Hilbert space. *Contemp. Math.* **26** (1984), 189–206.

[44] Kashin, B., The widths of certain finite dimensional sets and classes of smooth functions. *Izv. Akad. Nauk SSSR Ser. Mat.* **41** (1977), 334–351; English transl. *Math. USSR-Izv.* **11** (1978), 317–333.

[45] Kushilevitz, E., Mansour, Y., Learning decision trees using the Fourier spectrum. In *Proceedings of the twenty-third annual ACM symposium on the theory of computing*, ACM, New York 1991, 455–464; *SIAM J. Comput* **22** (1993), 1331–1348.

[46] Lee, W. S., Bartlett, P., and Williamson, R. C., Efficient agnostic learning of neural networks with bounded fan-in. *IEEE Trans. Inform. Theory* **42** (1996), 2118–2132.

[47] Lorentz, G. G., von Golitschek, M., and Makovoz, Yu.. *Constructive Approximation:Advanced Problems*. Grundlehren Math. Wiss. 304, Springer-Verlag, Berlin, Heidelberg 1996.

[48] Mallat, S., Multiresolution approximation and wavelet orthonormal bases of $L^2(\mathbb{R})$. *Trans. Amer. Math. Soc.* **315** (1989), 69–88.

[49] Mallat, S., *A Wavelet Tour of Signal Processing*. Academic Press, New York 1998.

[50] Meyer, Y., Ondelettes sur l'intervalle. *Rev. Mat. Iberoamer.* **7** (1991), 115–134.

[51] Meyer, Y., *Ondelettes et Opérateurs I*. Actualites Math., Hermann, Paris 1990 (English translation by D. H. Salinger, Cambridge Stud. Adv. Math. 37, Cambridge University Press, Cambridge 1992).

[52] Micchelli, C., and Rivlin, T., A survey of optimal recovery. In *Optimal Estimation in Approximation Theory* (C. A. Micchelli and T. J. Rivlin, eds.), Plenum Press, New York 1977, 1–54.

[53] Morin, P., Nochetto, R., and Siebert, K., Data Oscillation and convergence of adaptive FEM. *SIAM J. Numer. Anal.* **38** (2000), 466–488.

[54] Pinkus, A., *n-Widths in Approximation Theory*. Ergeb. Math. Grenzgeb. 7, Springer Verlag, Berlin 1985.

[55] Romberg, J., M. Wakin, R. Baraniuk, Approximation and Compression of Piecewise Smooth Images Using a Wavelet/Wedgelet Geometric Model. IEEE International Conference on Image Processing, Barcelona, Spain, September 2003.

[56] Temlyakov, V., Nonlinear Methods of Approximation. *Found. Comput. Math.* **3** (2003), 33–107.

[57] Traub, J., and Wozniakowski, H., *A General Theory of Optimal Algorithms*. Academic Press, New York, NY, 1980.

[58] Traub, J. F., Wasilkowski, G. W., and Woźniakowski, H., *Information-Based Complexity*. Academic Press, New York, NY, 1988.

Department of Mathematics, University of South Carolina, Columbia, SC 29208, U.S.A.
E-mail: devore@math.sc.edu