

Course Notes (Paris 2009)

Ronald DeVore

June 22, 2009

Abstract

1 Lecture 3: Capturing Functions in High Dimensions

1.1 Classifying High Dimensional Functions:

Our last two lectures will study the problem of approximating (or capturing through queries) a function f defined on $\Omega \subset \mathbb{R}^N$ with N very large. The usual way of classifying functions is by smoothness. The more derivatives a function has the nicer it is and the more efficiently it can be numerically approximated. However, as we move into high space dimension, this type of classification will suffer from the so-called *curse of dimensionality* which we shall now quantify.

2 Widths and Entropy of Classes

We can quantify the curse of dimensionality through concepts like entropy and widths.

Kolmogorov Entropy: Let us first consider entropy. Suppose that K is a compact set in the Banach space X with norm $\|\cdot\|_X$. If $\epsilon > 0$, we consider all possible coverings of $K \subset \cup_{i=1}^m B(x_i, \epsilon)$ using balls $B(x_i, \epsilon)$ of radius ϵ with centers $x_i \in X$. The smallest number $m = N_\epsilon(K)_X$ is called the covering number of K . The Kolmogorov entropy of K is then defined as

$$H_\epsilon(K)_X := \log_2(N_\epsilon(K)_X). \quad (2.1)$$

The Kolmogorov entropy measures the size or massivity of K . It has another important property of determining optimal encoding of the elements of K . Namely, if $x \in K$ then we can assign to x the binary bits of an index i for which $x \in B(x_i, \epsilon)$. Each x is then encoded to accuracy ϵ with $\leq \lceil H_\epsilon(K)_X \rceil$ bits and no other encoder can do better.

It is frequently more convenient to consider the entropy numbers

$$\epsilon_n(K)_X := \inf\{\epsilon : H_\epsilon(K)_X \leq n\}. \quad (2.2)$$

Typically, $\epsilon_n(K)_X$ decay like n^{-r} for standard compact sets. Not only does $\epsilon_n(K)_X$ tell us the minimal distortion we can achieve with n bit encoding, it also indicates that any numerical

algorithm which computes an approximation to each of the elements of K to accuracy $\epsilon_n(K)_X$ will require at least n operations.

Widths: There are several types of widths. The most prominent of these is the Kolmogorov width which measures how well K can be approximated through linear spaces of fixed dimension n . However, for us, the following definition of manifold width [1] will be more useful since it also governs nonlinear processes. We consider two continuous functions. The first function a maps each element $x \in K$ into \mathbb{R}^n and the second function M maps \mathbb{R}^n into the set \mathcal{M} (which we view as an n -dimensional manifold although we make no assumptions about the smoothness of the image \mathcal{M}). The manifold width of the compact set K is then defined by

$$\delta_n(K)_X := \inf_{M,a} \sup_{x \in K} \|x - M(a(x))\|_X. \tag{2.3}$$

For typical compact sets K of functions, the manifold widths behave like the entropy numbers. For example, if K is the unit ball of any Besov or Sobolev space of smoothness s which compactly embeds into $L_p(\Omega)$ with $\Omega \subset \mathbb{R}^N$, then (see [2])

$$C_0 n^{-s/N} \leq \delta_n(K)_{L_p(\Omega)} \leq C_1 n^{-s/N}. \tag{2.4}$$

We see in (2.4) the curse of dimensionality. In order to obtain just moderate rates of convergence with $n \rightarrow \infty$ we need s to be comparable with N .

3 Model classes for functions in high dimension

We are interested in what are reasonable models for function classes \mathcal{F} in \mathbb{R}^N when N is large. The ultimate test is whether a proposed model class is commensurate with applications. However, we also need this class to be amenable to computation which in light of the previous section means that its manifold width or Kolmogorov entropy should tend to zero like n^{-r} for not too small of values of r . Of course this can be achieved by assuming the elements in \mathcal{F} are very smooth but this may not be reasonable from the viewpoint of the specified application. Another approach is to suppose that although the elements in \mathcal{F} depend on N variables there are relatively few variables which dominate. Our goal here is to introduce and consider model classes that quantify this idea.

We shall begin by considering the following model classes. We assume that Φ is a compact set of continuous functions $\phi : \Omega \rightarrow \Omega_0$ with $\Omega \subset \mathbb{R}^N$ compact and $\Omega_0 \subset \mathbb{R}^k$ also compact. Here k is fixed and should be viewed as much smaller than N . Let K be a compact set of functions in $C(\Omega_0)$ (which is typically the unit ball of some smoothness norm $\|\cdot\|_{W^s}$). We consider the class $\mathcal{F} := \mathcal{F}(K, \Phi) := \{f = g(\phi) : \phi \in \Phi, g \in K\}$ which will be compact in $C(\Omega)$. Notice that \mathcal{F} combines the notions of smoothness and variable reduction. In the case that K corresponds to a smoothness class for nonlinear approximation (like certain Besov classes) then \mathcal{F} is also incorporating sparsity.

4 Recovering functions of few variables in high dimension

We begin by considering the special case of the above model classes where $K := U(C^s)$ and Φ is the set of coordinate projections onto a k -dimensional coordinate space. This is part of an ongoing study with Przemek Wojtaczzyk and Guergana Petrova. Each $f \in \mathcal{F}(K, \Phi)$ is of the form

$$f(x_1, \dots, x_N) = g(x_{i_1}, \dots, x_{i_k}), \quad |g|_{C^s} \leq 1 \quad (4.1)$$

where $C^s = C^s([0, 1]^k)$ is the set of functions which have s continuous derivatives, equipped with its usual semi-norm

$$|f|_{C^s(\Omega)} := \max_{|\nu| \leq s} \|D^\nu f\|_{C(\Omega)}.$$

We can allow s to be non-integer by working with Lipschitz spaces. Notice that \mathcal{F} is not a linear space.

The first problem we wish to consider is the following. Suppose we are given a budget of n points for which we can ask the values of f . Where should we choose these points in order to best recover f in the norm of $C(\Omega)$ and what is the best error we can receive. This is a special case of what is called *optimal recovery*. It is also sometimes called *directed learning*. There are two settings to consider. The first *adaptively* asks the questions. The j -th query can depend on the answer to the first $j - 1$ queries. The second one sets once and for all the n questions (non-adaptive). We shall consider both settings in what follows.

4.1 One variable of dependence

It will be instructive to consider first the case when f depends only on one coordinate variable where the arguments and proofs are most transparent. That is, we suppose $f(x_1, \dots, x_N) = g(x_j)$ with both g and j unknown to us. We take the domains $\Omega = [0, 1]^N$ and $\Omega_0 := [0, 1]$ for simplicity.

We shall describe first a nonadaptive set of points where we will ask for the values of f . This set will be the union of two point sets. The first of these sets is $\mathcal{P} := \{P_i := m^{-1}(i, i, \dots, i)\}_{i=0}^m$ which we call the set of *base points*. The important property of this set is that when its points are projected onto any of the coordinate axes, we get a set of $m + 1$ equally spaced points with spacing $h := 1/m$.

The second set of points we shall need are *padding points*. These points will be used to find the coordinate j . Padding points are associated to a pair of points $P, P' \in \mathcal{P}$ and are constructed as follows. Every integer $j \in \{1, \dots, N\}$ has a unique binary representation $j = 1 + \sum_{k=0}^L b_k(j)2^k$ where $L := \lceil \log_2(N) \rceil - 1$ and each bit $b_k(j) \in \{0, 1\}$. Given a pair of points $P, P' \in \mathcal{P}$ and a value of $k \in \{0, 1, \dots, L\}$, we define the point $[P, P']_k$ whose j -th coordinate is $P(j)$ (i.e. the same as the j -th coordinate of P) if $b_k(j) = 0$ and otherwise this coordinate of $[P, P']_k$ is defined to be the same as the j -th coordinate of P' . Thus any of these padding points corresponds to incrementing about half of the coordinate values from P to P' .

We ask for the values of f at the base points in \mathcal{P} given above. We also ask for the values of f at the following set \mathcal{Q} of padding points. To each pair $P_{i-1}, P_i, i = 1, \dots, m$, of consecutive

points, we associate the padding points $[P_{i-1}, P_i]_k$, $k = 0, \dots, L$. Thus the collection of padding points is $\mathcal{Q} := \{[P_{i-1}, P_i]_k, i = 1, \dots, m, k = 0, \dots, L\}$. Clearly there are $m(L + 1)$ points in \mathcal{Q} .

After receiving the values of f at the base points \mathcal{P} (which are the values $g(i/m)$, $i = 0, \dots, m$, of the univariate function g), we can construct a function $\hat{g} = A_m(g)$ which approximates g to accuracy h^s , $h := 1/m$ (for example using piecewise polynomials). Namely,

$$\|g - \hat{g}\|_{C[0,1]} \leq C_s |g|_{C^s} h^s. \quad (4.2)$$

We can also assume that when $g = c$ is constant on \mathcal{P} then $A_m(g) = c$. The function $\hat{g}(x_j)$ would provide a good approximation to f if we knew j .

Notice that if f is constant on \mathcal{P} then we do not need to know j . If f is not constant on \mathcal{P} then we shall use the padding points to find j . There is an i such that $f(P_{i-1}) \neq f(P_i)$ with $1 \leq i \leq m$. The value $f([P_{i-1}, P_i]_k)$ is either $f(P_{i-1})$ or $f(P_i)$. If it is $f(P_{i-1})$, then we know that $b_k(j) = 0$; if it is $f(P_i)$ then we know $b_k(j) = 1$. Thus from the answer to these questions, we know all the bits of j and hence we know j . We define our approximation to f to be $\hat{f}(x_1, \dots, x_N) := \hat{g}(x_j)$.

Theorem 4.1 *If $f(x_1, \dots, x_N) = g(x_j)$ with $g \in C^s$, then the function \hat{f} defined above satisfies*

$$\|f - \hat{f}\|_{C(\Omega)} \leq C_s |g|_{C^s} h^s. \quad (4.3)$$

Let us note that Algorithm 1 asks for the values of f at $m + 1 + m(L + 1)$ points. The logarithm in L is the price we pay for not knowing the change coordinate j .

Gain of Adaptivity: *If we are willing to use an adaptive algorithm we can save on the number of queries as follows. From the base points we determine an i for which $f(P_i) \neq f(P_{i+1})$. Then we only need the padding points for the pair P_i, P_{i+1} . Hence, we need a total of $m + 1 + L + 1$ points in all. This matches the entropy and manifold width for \mathcal{F} and hence the above result is optimal.*

We can also construct an algorithm which will handle the case where f is not necessarily a function of just one variable but is approximated by such a function. Given the univariate function g , we define the multivariate functions $G_\nu(x_1, \dots, x_N) := g(x_\nu)$ for any $\nu = 1, \dots, N$. Our new model for f is that there is a $j \in \{1, \dots, N\}$ and an $\epsilon > 0$ such that

$$\|f - G_j\|_{C(\Omega)} \leq \epsilon. \quad (4.4)$$

We do not know g , j or ϵ .

Let us describe the adaptive version of an algorithm for this model class. We use the values of f at the points $P_i \in \mathcal{P}$, $i = 0, \dots, m$ to determine \hat{g} as before. Now to find a change coordinate j from this information, we choose a pair $(P_i, P_{i'})$, $i < i'$, for which $|f(P_i) - f(P_{i'})|$ is the largest among all such pairs. To identify the changing coordinate, we proceed as follows. We consider the value $f(Q_k)$ at each of the points $Q_k := [P_i, P_{i'}]_k$, $k = 0, \dots, L$. If this value is closest to $f(P_i)$ we assign the bit $b_k = 0$. If this value is closest to $f(P_{i'})$ or if there is a tie, we assign the bit $b_k = 1$. These bits determine an integer $j \in \{1, \dots, N\}$. We define $\hat{f}(x_1, \dots, x_N) = \hat{g}(x_j)$.

Theorem 4.2 *Suppose that f is a function of N variables for which there is a function $g \in C^s$ and a $\nu \in \{1, \dots, N\}$ such that*

$$\|f - G_\nu\|_{C(\Omega)} \leq \epsilon. \quad (4.5)$$

Then the function \hat{f} defined above satisfies

$$\|f - \hat{f}\|_{C(\Omega)} \leq C_s[\epsilon + |g|_{C^s} h^s]. \quad (4.6)$$

Proof: We consider two cases.

Case 1: We assume in this case that the maximum deviation in the values $f(P_i)$, $i = 0, \dots, m$, is $> 4\epsilon$. By the definition of i, i' we have that $|f(P_i) - f(P_{i'})| > 4\epsilon$. At each of the padding points $Q_k := [P_i, P_{i'}]_k$, $k = 0, \dots, L$, we have $|f(Q_k) - G_\nu(Q_k)| \leq \epsilon$. Now if $b_k(\nu) = 0$ then $G_\nu(Q_k) = G_\nu(P_i)$ (since G_ν is a function only of the ν -th variable) and therefore $f(Q_k)$ is within 2ϵ of $f(P_i)$ but further than 2ϵ from $f(P_{i'})$. This means that the bit b_k assigned by the algorithm will be zero and therefore $b_k = b_k(\nu)$. The same conclusion holds if $b_k(\nu) = 1$. Hence the value j determined by the algorithm is equal to ν . We therefore obtain

$$\|f - \hat{f}\|_{C(\Omega)} \leq \|f - G_\nu\|_{C(\Omega)} + \|g - \hat{g}\|_{C([0,1])} \leq \epsilon + C_s|g|_{C^s} h^s. \quad (4.7)$$

which is the desired inequality.

Case 2: In this case, the maximum deviation of f over the points P_i , $i = 0, \dots, m$, is $\leq 4\epsilon$. Hence the maximum deviation of g over the points $h\mathcal{L}_1 = \{0, 1/m, \dots, 1\}$ is at most 6ϵ . We consider the function $\tilde{g} = g - c$ where c is the median value of g on $h\mathcal{L}_1$. Then $|\tilde{g}| \leq 3\epsilon$ on $h\mathcal{L}_1$. Using the fact that \tilde{g} is in C^s , it follows that $\|\tilde{g}\|_{C([0,1])} \leq C_s(\epsilon + |g|_{C^s} h^s)$ and further that $\|G_\nu - G_j\|_{C(\Omega)} \leq C_s(\epsilon + |g|_{C^s} h^s)$. Hence,

$$\|f - G_j\|_{C(\Omega)} \leq \|f - G_\nu\|_{C(\Omega)} + \|G_\nu - G_j\|_{C(\Omega)} \leq \epsilon + C_s(\epsilon + |g|_{C^s} h^s), \quad (4.8)$$

where j is the coordinate assigned by our algorithm. Finally, using that $\|\hat{f} - G_j\|_{C(\Omega)} \leq C_s|g|_{C^s} h^s$ we obtain the theorem. Notice that in this case we may have selected a wrong change coordinate j . However, since the maximum deviation of f over P_i , $i = 1, \dots, m$, is small, estimate (4.6) still holds. \square .

5 The general case of k variables

The results we have just obtained for the case of one change variable extend to the general case of k change variables but the constructions and proofs are more substantial. We shall point out some of the essential new ingredients.

Our starting point is to assume that we have a set \mathcal{P} of base points in \mathbb{R}^N with certain properties. Let \mathcal{A} be a collection of partitions \mathbf{A} of $\{1, 2, \dots, N\}$. Each \mathbf{A} consists of k disjoint sets A_1, \dots, A_k . We require:

Partition Assumption *The collection \mathcal{A} is rich enough that it has the following two properties:*

(i) *given any k distinct integers $i_1, \dots, i_k \in \{1, \dots, N\}$, there is a partition \mathbf{A} in \mathcal{A} such that each set in \mathbf{A} contains precisely one of the integers i_1, \dots, i_k .*

(ii) For any $k + 1$ distinct integers i, i_1, \dots, i_k , there is a partition \mathbf{A} such that one of the sets A_ν contains i but none of the other i_1, \dots, i_k .

Property (i) is called *perfect hashing* in combinatorics/ computer science. It is an interesting question to understand the fewest number of sets \mathcal{A} that satisfy the **Partition Assumption**. We shall show later that $C(k) \log N$ partitions suffice. But for now, we assume that we have such a collection \mathcal{A} in hand and proceed to construct an algorithm for approximating f .

Corresponding to any $\mathbf{A} \in \mathcal{A}$ we construct the set of *base points*

$$P = \sum_{i=1}^k \alpha_i \chi_{A_i}, \quad \alpha_i \in \{0, 1/m, \dots, 1\}. \quad (5.1)$$

In other words, P has coordinate value α_i at each of the coordinate indices in A_i . We denote by \mathcal{P} the set of all such base points. Note that there are $(m + 1)^k \#(\mathcal{A})$ such base points.

The important property of the base points is:

Projection Property: *Given any $\mathbf{j} = (j_1, \dots, j_k)$ and any integers $0 \leq i_1, \dots, i_k \leq m$, there is a point $P \in \mathcal{P}$ such that the coordinate j_ν of P is i_{j_ν}/m . Said in another way, given any k dimensional coordinate plane, the projection of \mathcal{P} onto this plane gives a uniform grid with spacing $h := 1/m$.*

Indeed, it is enough to take a partition \mathbf{A} from \mathcal{A} such that each j_ν is in a different set A_i of \mathbf{A} . Then the point (5.1) with the appropriate value of $\alpha_i = i_{j_\nu}/m$ when $j_\nu \in A_i$ will have the value i_{j_ν}/m at coordinate j_ν .

Next we show how the partitions \mathcal{A} give a bit representation for any integer $j \in \{1, \dots, N\}$. We denote by $b_{\mathbf{A}}(j)$ the integer $i \in \{1, \dots, k\}$ for which $j \in A_i$. We view the vector $(b_{\mathbf{A}}(j))_{\mathbf{A} \in \mathcal{A}}$ as a bitstream associated to j .¹ Let us observe

Uniqueness Property: *If two integers j, j' have identical bitstreams, then $j = j'$.*

Indeed, if $j \neq j'$ then we can find \mathbf{A} such that j and j' lie in different partition sets of \mathbf{A} because of the **Partition Assumption**. Therefore, $b_{\mathbf{A}}(j) \neq b_{\mathbf{A}}(j')$.

We shall also need padding points. In our analog of the first univariate algorithm, we construct the padding points for certain ordered pairs of base points which we call *admissible pairs*. An ordered pair (P, P') with $P \neq P'$ of base points is admissible if: (a) they are subordinate to the same partition \mathbf{A} , (b) there is an integer i such that P and P' agree on all of the sets $A_\nu \neq A_i$ and on A_i , the coordinate values of P and P' are each constant and differ by $1/m$. Given P , we see that there are $2k$ choices of P' for which P, P' is an admissible pair. Namely, we have k choices for A_i and once this choice is made there are two choices for the values of P' on A_i . Hence, there are $2k \#(\mathcal{P}) = 2k(m + 1)^k \#(\mathcal{A})$ admissible pairs.

For each admissible pair P, P' we define a collection of padding points $Q = [P, P']_{\mathbf{B}, \nu}$ for each $\mathbf{B} \in \mathcal{A}$, $\mathbf{B} \neq \mathbf{A}$, and $\nu \in \{1, \dots, k\}$ as follows. The j -th coordinate of Q for each $j \in A_\mu$, $\mu \neq i$, is the common j -th coordinate of P and P' . In other words, we do not alter the padding points except on A_i . For each $j \in A_i$, the j -th coordinate of Q is the same as that of P' if $j \in A_i \cap B_\nu$. Otherwise it is the same as that of P . In other words, the padding points change some of the coordinates of P to that of P' . The coordinates that are changed are precisely those in $B_\nu \cap A_i$.

¹ In the language of theoretical computer science we are using here a perfect hash function.

We denote the set of all such padding points by \mathcal{Q} . Since there are $\#(\mathcal{A}) - 1$ choices of \mathbf{B} and k choices of ν there are at most $k(\#(\mathcal{A}) - 1)$ padding points associated to each admissible pair. Thus,

$$\#(\mathcal{Q}) \leq 2k^2 \#(\mathcal{P})(\#(\mathcal{A}) - 1) = 2k^2(m + 1)^k (\#(\mathcal{A}))(\#(\mathcal{A}) - 1). \quad (5.2)$$

We now proceed to describe an algorithm which is the non- adaptive version of the algorithm for one change coordinate. Again, adaptive questioning would help reduce some on the number of questions we ask.

We assume that f is a function that depends on at most k variables (unknown to us) whose indexing we denote by $\mathbf{j} = (j_1, \dots, j_k)$ with $1 \leq j_1 < j_2 < \dots < j_k \leq N$. We call the entries in \mathbf{j} the *change coordinates* of f . We introduce the following general notation. Given a vector $\mathbf{l} = (l_1, \dots, l_k)$, with $1 \leq l_1 < \dots < l_k \leq N$, we define the function $G_{\mathbf{l}}$ by $G_{\mathbf{l}}(x_1, \dots, x_N) = g(x_{l_1}, \dots, x_{l_k})$. Thus, we know that $f = G_{\mathbf{j}}$.

The Algorithm starts by asking for the values of f at all points in $\mathcal{P} \cup \mathcal{Q}$. We then proceed to find the change coordinates \mathbf{j} as follows. Given any admissible pair P, P' , let \mathbf{A} be the subordinating partition of P and P' and let A_i be the set in \mathbf{A} where P and P' take differing values. We examine the values of f at all the padding points Q associated to this pair. We say the pair P, P' is *useful* if for each $\mathbf{B} \in \mathcal{A}$, we have exactly one value $\nu = \nu(\mathbf{B})$ where $f([P, P']_{\mathbf{B}, \nu}) = f(P')$ and for all $\mu \neq \nu$, we have $f([P, P']_{\mathbf{B}, \mu}) = f(P)$. For each such admissible and useful pair, we then define

$$J_{P, P'} := \bigcap_{\mathbf{B} \in \mathcal{A}} B_{\nu(\mathbf{B})} \cap A_i \quad (5.3)$$

The following lemma will show that we can identify the change coordinates of f . We say that the change coordinate j_ν is visible at scale m if there exists two points $m^{-1}(i_1, \dots, i_N)$ and $m^{-1}(i'_1, \dots, i'_N)$, $0 \leq i_1, i'_1, \dots, i_N, i'_N \leq m$, which are identical in all coordinates except for the j_ν -th coordinate and $f(m^{-1}(i_1, \dots, i_N)) \neq f(m^{-1}(i'_1, \dots, i'_N))$. We can always take $i'_{j_\nu} = i_{j_\nu} + 1$

Lemma 5.1 *The following properties hold:*

- (i) *For each admissible and useful pair P, P' the set $J_{P, P'}$ is either empty or it contains precisely one integer j ,*
- (ii) *this integer j is one of the change coordinates j_1, \dots, j_k of f .*
- (iii) *For each change coordinate j which is visible at scale m , there is an admissible, useful pair P, P' for which $J_{P, P'} = \{j\}$*

Proof: (i) Given any two distinct integers $j, j' \in A_i$, we want to show that not both of these integers can be in $J_{P, P'}$. To see this, we take a partition \mathbf{B} such that $j \in B_\nu$ and $j' \in B_{\nu'}$ and $\nu \neq \nu'$. The existence of such a partition follows from the **Partition Assumption**. From the definition of useful, we cannot have both $f([P, P']_{\mathbf{B}, \nu}) = f(P')$ and $f([P, P']_{\mathbf{B}, \nu'}) = f(P')$. So only one of these integers j, j' can be in $J_{P, P'}$.

(ii) Suppose $J_{P, P'} = \{j\}$. If j is not a change coordinate then by virtue of the **Partition Assumption** there is a partition \mathbf{B} in which j appears in one set B_μ and all the change coordinates are outside B_μ . But since there are no change coordinates in B_μ , we have

$f([P, P']_{\mathbf{B}, \mu}) = f(P) \neq f(P')$. Hence $\mu \neq \nu(B)$ which is a contradiction to j being in $J_{P, P'}$. Thus, j must be a change coordinate.

(iii) Given a change coordinate j which is visible at scale m , we know there is a point $R := m^{-1}(i_1, \dots, i_N)$ such that incrementing i_j by one gives a point $R' := m^{-1}(i'_1, \dots, i'_N)$ at which the value of f changes, i.e. $f(R') \neq f(R)$. By the **Partition Assumption**, we can choose a partition \mathbf{A} such that each cell A_μ , $\mu = 1, \dots, k$, contains exactly one change coordinate. Let j be in cell A_i . Consider the pairs P, P' subordinate to \mathbf{A} , for which P and P' differ only on A_i . We can take such a pair so that P is identical to R at each change coordinate and P' is identical to R' at each change coordinate. Then, for any \mathbf{B} , we have $f([P, P']_{\mathbf{B}, \nu}) = f(P')$ if and only if $j \in B_\nu$. Thus, P, P' is useful and this also shows that $j \in J_{P, P'}$, as desired. \square

The lemma proves that the change coordinates of f that are visible at scale m have been identified. There may be $\ell \leq k$ of these coordinates, so we add arbitrarily $k - \ell$ coordinates to obtain \mathbf{j}' such that $j'_\nu = j_\nu$ for any ν such that j_ν is visible at scale m . On the lattice $h(i_1, \dots, i_N)$, $0 \leq i_1, \dots, i_N \leq m$, with $h := m^{-1}$, we have $f = G_{\mathbf{j}} = G_{\mathbf{j}'}$.

We can now identify the values of g at each of the lattice points $h\mathcal{L}_k := \{h(i_1, \dots, i_k)\}$, where $0 \leq i_1, \dots, i_k \leq m$ as follows. We take a partition $\mathbf{A} \in \mathcal{A}$ for which each coordinate j'_ν lies in a different set of the partition. If we take the coordinates of P to be $i_{j'_\nu}/m$ on the set of \mathbf{A} which contains j'_ν , then we obtain a base point P such that $f(P) = G_{\mathbf{j}}(P) = G_{\mathbf{j}'}(P) = g(h(i_1, \dots, i_k))$.

Now that we have the values of g on the lattice $h\mathcal{L}_k$, we can find the approximation $A_m(g)$ which satisfies $\|g - A_m(g)\|_{C(\Omega_0)} \leq C_s |g|_{C^s} h^s$. We define $\hat{f}(x_1, \dots, x_N) := A_m(g)(x_{j'_1}, \dots, x_{j'_k})$.

Theorem 5.2 *The number of point values used in Algorithm 1 is $\leq 2k^2(m+1)^k(\#\mathcal{A})^2$. If $f = G_{\mathbf{j}}$ with $g \in C^s$, then the function \hat{f} defined by Algorithm 1 satisfies*

$$\|f - \hat{f}\|_{C(\Omega)} \leq C_s |g|_{C^s} h^s. \quad (5.4)$$

Proof: The number of point values used is

$$\#(\mathcal{P}) + \#(\mathcal{Q}) \leq (m+1)^k(\#\mathcal{A}) + 2k^2(m+1)^k(\#\mathcal{A})(\#\mathcal{A} - 1) \leq 2k^2(m+1)^k(\#\mathcal{A})^2.$$

To prove the bound on the approximation error, we note that

$$f - \hat{f} = G_{\mathbf{j}} - \hat{f} = G_{\mathbf{j}} - G_{\mathbf{j}'} + G_{\mathbf{j}'} - \hat{f} = G_{\mathbf{j}} - G_{\mathbf{j}'} + [g - A_m(g)](x_{j'_1}, \dots, x_{j'_k}). \quad (5.5)$$

The first term on the right satisfies

$$[G_{\mathbf{j}} - G_{\mathbf{j}'}](x_1, \dots, x_N) = g(x_{j_1}, \dots, x_{j_k}) - g(x_{j'_1}, \dots, x_{j'_k}) = \bar{g}(x_{j_1}, \dots, x_{j_k}, x_{j'_1}, \dots, x_{j'_k}).$$

Since \bar{g} is a C^s function of $\ell < 2k$ variables which vanishes on the lattice $h\mathcal{L}_\ell$, one finds $\|G_{\mathbf{j}} - G_{\mathbf{j}'}\| \leq C_s |g|_{C^s} h^s$. The second term on the right of (5.5) can also be bounded by $C_s |g|_{C^s} h^s$ and we therefore obtain (5.4). \square

Remarks :

1. We show next that there are sets \mathcal{A} which satisfy the **Partition Assumption** which contain $\approx 2ke^k[\ln N]$ elements. This gives a $C(k)[\log N]^2$ bound for the number of queries in the algorithm. One logarithm can be dropped by working adaptively.

2. There is also an adaptive version of the approximation result Theorem 4.2 which we shall not formulate because of time.

6 Constructing separating partitions

The last algorithm we have given begins with a set \mathcal{A} of partitions that satisfy the **Partition Assumption**. It is important to know how large \mathcal{A} needs to be for this assumption to hold. Indeed $\#(\mathcal{A})$ controls the size of the sets \mathcal{P} and \mathcal{Q} where we sample f . We shall begin by giving a constructive way to find sets \mathcal{A} that satisfy the **Partition Assumption** that works in the case k is small. Later we shall give a probabilistic proof that for any k there are sets \mathcal{A} which satisfy the **Partition Assumption** and have reasonable cardinality.

The problem of finding sets \mathcal{A} satisfying (i) of our **Partition Assumption** is known in theoretical computer science as perfect hashing. Let us consider the case $k = 2$ which will be illustrative of how to do low dimensional constructions. Consider first the collection \mathcal{B} of binary partitions \mathbf{B} of $\{1, \dots, N\}$. Thus each \mathbf{B} is determined by an integer $\nu \in \Lambda := \{1, \dots, \lceil \log_2 N \rceil\}$ and gives the two sets $B_0(\nu), B_1(\nu)$ where B_0 (respectively B_1) consists of all the integers in $\{1, \dots, N\}$ whose ν -th binary bit is 0 (respectively 1). The collection \mathcal{B} will clearly satisfy (i) of the **Partition Assumption** but it will not satisfy (ii). To obtain (ii), we add some additional partitions to \mathcal{B} . Namely, for each $\nu, \nu' \in \{1, \dots, \lceil \log_2 N \rceil\}$, $\mu, \mu' = 0, 1$, we consider the new partition given by the two sets

$$[B_0(\nu) \cap B_\mu(\nu')] \cup [B_1(\nu) \cap B_{\mu'}(\nu')], \quad [B_0(\nu) \cap B_{\bar{\mu}}(\nu')] \cup [B_1(\nu) \cap B_{\bar{\mu}'}(\nu')], \quad (6.1)$$

where $\bar{\mu}$ denotes the complimentary index to μ (if $\mu = 0$ then $\bar{\mu} = 1$ and vice versa). When these partitions are added to the collection \mathcal{B} , we obtain a collection \mathcal{A} which satisfies both (i) and (ii) of the **Partition Assumption**. To verify (ii), let j, j_1, j_2 be three distinct integers from $\{1, \dots, N\}$ and choose ν such that j is in one of the $B_i(\nu)$ (say $B_0(\nu)$) and j_1 is in the other (say $B_1(\nu)$). If j_2 is in $B_1(\nu)$ then we have the desired partition. In the other case, where both j, j_2 are in $B_0(\nu)$, we choose a second partition $B_0(\nu'), B_1(\nu')$ which separates j and j_2 . In this case, we can without loss of generality assume that $j \in B_0(\nu) \cap B_0(\nu')$ and $j_2 \in B_0(\nu) \cap B_1(\nu')$. The integer j_1 will be in either $B_1(\nu) \cap B_0(\nu')$ or $B_1(\nu) \cap B_1(\nu')$. Hence one of the partitions in (6.1) will separate j from j_1, j_2 . Notice that the collection \mathcal{A} will have cardinality $\lceil \log_2 N \rceil + 4\lceil \log_2 N \rceil^2$.

Constructions of the above form become more unwieldy as k increases. Also they increase the power of $\log_2 N$. However, there are probabilistic arguments which show for any given k , the existence of a family \mathcal{A} which satisfies the Partition Assumption and has favorable cardinality.

Suppose we are given N and $k < N$. We are interested in partitions \mathbf{A} of $\Lambda := \{1, \dots, N\}$ consisting of k disjoint sets A_1, \dots, A_k . We view each A_i as a bucket which will have in it a collection of integers from Λ . We want to have sufficiently many partitions \mathbf{A} to satisfy the Partition Assumption but we also want a control on the number of such partitions we shall need. We shall see that $2ke^k \lceil \ln N \rceil$ partitions will suffice.

To see this, we let ϕ denote the random variable which takes the values $\{1, \dots, k\}$ with the equal probability $1/k$. Alternatively, one can think of a draw from a box of balls labeled $1, \dots, k$. We randomly draw a ball, record its label, and then replace the ball into the box to repeat the experiment. We shall take qN independent draws ϕ_j of ϕ . We shall decide q later. Those variables define q random partitions $\mathbf{A} = \mathbf{A}(j) = \{A_1(j), \dots, A_k(j)\}$, $j = 1, \dots, q$, as

follows. For each $\mu \in \Lambda$, we place $\mu \in A_s(j)$ if and only if $\phi_{(j-1)N+\mu}(\omega) = s$. In other words, for each partition $\mathbf{A}(j)$ we run through the integers $1, \dots, N$ in order and place the integer μ in the bucket $A_s = A_s(j)$ if the μ -th draw is s . Notice that for a given j , some of the sets $A_1(j), \dots, A_k(j)$ may be empty.

If we are given $\mathbf{l} = \{l_1, \dots, l_k\}$ then it is easy to see that the probability that for a given j , the partition $\mathbf{A}(j)$ separates the entries of \mathbf{l} into distinct sets is $k!/k^k$. Indeed, the probability that l_i is in $A_i(j)$, for each $i = 1, \dots, k$, is k^{-k} . But any permutation of the l_i will do as well and we have $k!$ of these. So the probability that a random partition does not separate a given \mathbf{l} is $a := (1 - \frac{k!}{k^k})$. Therefore, if we have q independent partitions the probability that none of them separates \mathbf{l} is a^q . There are $\binom{N}{k}$ k -tuples \mathbf{l} when arranged in increasing order. Thus, if $\binom{N}{k}a^q < 1$ then a set of q random partitions will separate the coordinates of every \mathbf{l} with positive probability.

To see how large we need to take q we use Stirling's formula to find

$$\binom{N}{k} \left(1 - \frac{k!}{k^k}\right)^q \leq \binom{N}{k} \left(1 - \frac{\sqrt{2\pi k}}{e^k}\right)^q \leq N^k (1 - e^{-k})^q \quad (6.2)$$

If we take $q \approx 2ke^k \ln N$ and use the fact that $(1 - 1/x)^x \leq e^{-1}$ for $x > 1$, we see that the right side of (6.2) is

$$< N^k e^{-2k \ln N} \leq N^{-k}.$$

Thus, we see that if we take $q \geq 2ke^k \ln N$ partitions generated randomly, then with probability greater than $1 - N^{-k}$ the resulting set \mathcal{A}_0 will satisfy (i) of the **Partition Assumption**.

Now we look at random partitions that satisfy condition (ii). Observe that once we assigned i to a set, the probability that j_1 is assigned to a different set is $(k-1)/k$, so the probability that i is assigned to one set and all j_i 's to other sets is $((k-1)/k)^k$. This means that the probability that for given numbers i, j_1, \dots, j_k the random partition fails (ii) is $1 - ((k-1)/k)^k$. So the probability that (ii) fails for q independent partitions and for all choices of j_1, \dots, j_k does not exceed (note we have $k \geq 2$)

$$\binom{N}{k} \left(1 - \left(\frac{k-1}{k}\right)^k\right)^q \leq N^k (3/4)^{2ke^k \ln N} = N^{k(1+2e^k \ln(3/4))}. \quad (6.3)$$

One checks that this is at most $N^{-\beta k}$ with $\beta \geq .2$. Comparing this with the estimate for the probability of failure of (i) we see that with great probability $q \approx 2ke^k \ln N$ random partitions satisfy **Partition Assumption**.

7 A second model class for high dimensional functions

One of the weaknesses of the model classes studied above is that they are very coordinate biased. It would be desirable to have results that hold for general change of bases. Namely, we would like the set Φ to consist of all $k \times N$ matrices. We will indicate the beginnings of such a theory under development with Albert Cohen, Gerard Kerkycharian, Dominique Picard, and Ingrid Daubechies. We will only discuss this theory for $k = 1$.

Consider a function defined for $x \in \Omega := [0, 1]^N$ by

$$f(x) = g(a \cdot x),$$

where g is an unknown function on $[0, 1]$ and $a = (a_1, \dots, a_N)$ is an unknown vector with nonnegative coordinate values. We shall assume $\sum_{j=1}^N a_j = 1$ (this assumption can be weakened somewhat). We will also fix a value of $q < 1$ and assume that $\|a\|_{\ell_q} \leq M$ and try to see what quality of results we can obtain depending on s and q .

We are interested in recovering f from a small number of measurements. In order to get a convergence estimate we will assume that $g \in C^s$ for some $s > 1$. To simplify this presentation, we shall assume that $s = 2$. The arguments below generalize easily to $s > 1$.

Let $h := 1/m$ as before. For convenience of notation, we assume that $|g|_{C^2} = 1$. Let us first note that we can obtain the value of g at any point t by asking for the values of f at $t(1, \dots, 1)$. We start by asking for the values at the base points of the previous section. Namely, we ask for the values $f(ih, \dots, ih) = g(ih)$ for $i = 0, \dots, m$. From this we reconstruct a piecewise linear interpolation \hat{g} of g associated to the grid points $t_i := ih$. Then $\|g - \hat{g}\|_{C[0,1]} \leq h^2$ and $\|g' - \hat{g}'\|_{L^\infty[0,1]} \leq h$.

Our next goal is to find an approximation \hat{a} to a by asking for further values of f . This is analogous to the padding points in the previous section. Let $A := \max_{0 \leq i, j \leq m} |g(t_i) - g(t_j)|$ and take a pair of points $t_i < t_j$ which assume A . Let $I_0 := [t_i, t_j]$. We ask for the value of g at the midpoint t of I_0 . If $|g(t_i) - g(t)| \geq |g(t_j) - g(t)|$, we define I_1 as $[t_i, t]$, otherwise we define I_1 as $[t, t_j]$. In either case, the values of g at the two endpoints of I_1 differ by at least $A/2$. We continue in this way and define I_2, \dots, I_J where J is an integer which will be chosen shortly. We do impose now that $2^{-J} \leq h^4$. Now at the two endpoints of $I_J = [\alpha_0, \alpha_1]$, g has values that differ by at least $A2^{-J}$. So there is a point $\xi \in I_J$ where $|g'(\xi)| \geq A$. Let $\delta := |I_J| \leq 2^{-J} \leq h^4$. It follows that

$$|g'(\alpha_0)| \geq |g'(\xi)| - h^4 \geq A - h^4. \quad (7.1)$$

CASE 1: $A \leq h^2$. In this case, there is a constant c such that $g - c$ takes values at the points j/m which are in absolute value $\leq h^2/2$. From this and the fact that $|g|_{C^2} = 1$ gives that $\|g - c\|_{C(\Omega_0)} \leq Ch^2$ and $\|\hat{g} - c\|_{C(\Omega_0)} \leq Ch^2$. Hence regardless how we define \hat{a} (as long as $\hat{a} \geq 0$ and $\|\hat{a}\|_{\ell_1} \leq 1$) we will have for $\hat{f} := \hat{g}(\hat{a})$,

$$\|f - \hat{f}\|_{C(\Omega)} \leq \|f - c\|_{C(\Omega)} + \|\hat{f} - c\|_{C(\Omega)} \leq \|g - c\|_{C(\Omega_0)} + \|\hat{g} - c\|_{C(\Omega_0)} \leq Ch^2. \quad (7.2)$$

CASE 2: $A > h^2$. In this case we proceed further to find a good approximation to a by asking more questions. Let Φ be an $n \times N$ Bernoulli compressed sensing matrix and let r be one of its rows. If we ask for the value of f at the point $\alpha_0(1, \dots, 1) + \delta r$, we receive the value of g at $\alpha_0 + \delta a \cdot r$. We want to use these to compute an approximation to $y_r := a \cdot r$. Observe that

$$|y_r| \leq \|a\|_{\ell_1^N} \|r\|_{\ell_\infty^N} \leq n^{-1/2}. \quad (7.3)$$

We have the following approximation to y_r :

$$\hat{y}_r := \frac{g(\alpha_0 + \delta y_r) - g(\alpha_0)}{g(\alpha_0 + \delta) - g(\alpha_0)} = \frac{\delta y_r g'(\alpha_0) + g''(\xi_1)(\delta y_r)^2/2}{\delta g'(\alpha_0) + g''(\xi_2)\delta^2/2} = y_r \frac{1 + \epsilon_1}{1 + \epsilon_2} = y_r + y_r \frac{\epsilon_1 - \epsilon_2}{1 + \epsilon_2}, \quad (7.4)$$

where

$$|\epsilon_1| = \frac{|g''(\xi_1)|\delta|y_r|}{2|g'(\alpha_0)|} \leq \frac{\delta n^{-1/2}}{2(A-h^4)} \quad (7.5)$$

and

$$|\epsilon_2| := \frac{|g''(\xi_2)|\delta}{2|g'(\alpha_0)|} \leq \frac{\delta}{2(A-h^4)}. \quad (7.6)$$

Assuming that $m \geq 2$, we have $A-h^4 \geq 3h^2/4$ and $|\epsilon_2| \leq 2h^2/3 \leq 1/6$. Hence, $(1+\epsilon_2)^{-1} \leq 6/5$. This gives

$$|y_r - \hat{y}_r| \leq \frac{24}{15}\delta h^{-2}|y_r| \leq 2|y_r|\delta h^{-2}. \quad (7.7)$$

If we do the above for each row r of Φ then we will ask for n additional values of f and we will find an approximation \hat{y} to $y = \Phi a$ satisfying

$$\|y - \hat{y}\|_{\ell_1^n} \leq 2\delta h^{-2}\|y\|_{\ell_1^n} \leq 2\sqrt{n}\delta h^{-2}, \quad (7.8)$$

where we have used that $\|y\|_{\ell_1^n} \leq \sqrt{n}$.

We now use ℓ_1 -minimization to decode \hat{y} resulting in a vector $\hat{a} \in \mathbb{R}^N$. The following lemma tells us the quality of the approximation of \hat{a} as an approximation to a .

Lemma 7.1 *Under the assumptions of Case 2, for any $k = 1, 2, \dots$, by choosing $n \approx k \log(N/k)$ and J large enough so that $2^{-J} \leq h^4 n^{-3/2} [\log(N/n)]^{-1/2}$, we find*

$$\|a - \hat{a}\|_{\ell_1} \leq C\{\sigma_k(a)_{\ell_1} + h^2\}, \quad (7.9)$$

for an absolute constant C .

Proof: To start with, we know that

$$\|y - \hat{y}\|_{\ell_2^n} \leq \sqrt{n}\|y - \hat{y}\|_{\ell_1^n} \leq 2n\delta h^{-2} \leq 2n^{-1/2}h^2, \quad (7.10)$$

and

$$\|y - \hat{y}\|_{\ell_\infty^n} \leq [\sqrt{\log(N/n)}]^{-1}\{2n^{-1/2}\sqrt{\log(N/n)}\delta h^{-2}\} \leq 2[\sqrt{\log(N/n)}]^{-1}n^{-1/2}h^2. \quad (7.11)$$

Hence, from the **CBMP** that there is a vector z with $\Phi(z) = \hat{y} - y$ and

$$\|z\|_{\ell_1^N} \leq C_0 h^2, \quad (7.12)$$

where we have used that $\delta \leq 2^{-J}$ and the choice of J . Since $\Phi(a+z) = \Phi(\hat{a}) = \hat{y}$, we have from the ℓ_1 instance optimality in ℓ_1 that

$$\|a+z-\hat{a}\|_{\ell_1} \leq C\sigma_k(a+z)_{\ell_1^N} \leq C\{\sigma_k(a)_{\ell_1} + \|z\|_{\ell_1}\} \leq C\{\sigma_k(a)_{\ell_1} + h^2\}, \quad (7.13)$$

where we used $\|z\|_{\ell_1^N} \leq C_0 h^2$. \square

Remark: We want the resulting \hat{a} to have nonnegative coordinates and $\sum_{j=1}^N \hat{a}_j = 1$. We can always modify the \hat{a} above to have this property without effecting the approximation error (7.9) because we know that a has these properties. So in going forward we assume \hat{a} has these properties.

The following theorem now summarized the performance of the algorithm. We formulate the theorem for all $1 < s$ even though we are only proving it for $s = 2$.

Theorem 7.2 If $f(x) = g(a \cdot x)$ with $\|a\|_{\ell_q^N} = M$ then the above algorithm uses

$$\leq C(m + m^{\frac{2}{1/q-1}} \log N) \quad (7.14)$$

queries of f and returns an approximation $\hat{f}(x) := \hat{g}(\hat{a} \cdot x)$ which satisfies

$$\|f - \hat{f}\|_{C(\Omega)} \leq C_s \{|g|_{C^s} + \|a\|_{\ell_q^N}\} h^s, \quad h := m^{-1}. \quad (7.15)$$

Proof: As has been our convention, we give the proof for $s = 2$ under our assumption that $|g|_{C^s} = 1$ and $\|a\|_{\ell_q^N} \leq M$. We ask m question which determines \hat{g} and we know that

$$\|g - \hat{g}\|_{C[0,1]} \leq h^2. \quad (7.16)$$

Next, given that $a \in \ell_q$, we can choose k as the smallest integer so that

$$k^{1-1/q} \leq m^{-2}. \quad (7.17)$$

Thus, $k \approx m^{\frac{2}{1/q-1}}$. We now define n to satisfy $n \approx k \log(N/n)$. Finally, we choose J so that

$$2^{-J} \leq m^{-4} n^{-3/2} [\log(N/n)]^{-1/2} \quad (7.18)$$

so that the assumptions of Lemma 7.1 apply. We will ask n questions using the compressed sensing matrix and arrive at the approximation

$$\|a - \hat{a}\|_{\ell_1^N} \leq C(h^2 + \sigma_k(a)_{\ell_1^N}) \leq C(h^2 + Mk^{-1/q+1}), \quad (7.19)$$

where the last inequality uses the fact that $\sigma_k(a)_{\ell_1^N} \leq \|a\|_{\ell_q^N} k^{1-1/q}$ as was proven in our first set of lecture notes.

In total, we have asked $m + J + n$ questions and from this and the definition of J and n one derives (7.14). To verify (7.15) we write $f(x) - \hat{f}(x) = g(a \cdot x) - g(\hat{a} \cdot x) + g(\hat{a} \cdot x) - \hat{g}(\hat{a} \cdot x)$, from which we derive

$$\|f - \hat{f}\|_{C(\Omega)} \leq \|g - \hat{g}\|_{C(\Omega)} + |g|_{Lip1} \|a - \hat{a}\|_{\ell_1^N} \leq C(h^2 + Mk^{-1/q+1}) \leq C(1 + M)h^2, \quad (7.20)$$

where we have used the definition of k . □

Remark: By using manifold widths, one can show that the above result cannot be improved. Although this is an interesting story, we will not have time to go into it in these lectures.

References

- [1] R. DeVore, R. Howard and C. Micchelli, *Optimal non-linear approximation*, Manuscripta Math., **63** (1989), 469–478.
- [2] R. DeVore, G. Kyriazis, D. Leviatan, and V. Tichomirov, *Wavelet compression and nonlinear n -widths*, Advances in Computational Math., **1**(1993), 197–214.