Chapter 4

Regression

Contents

4.1 The model

Consider observing i.i.d. pairs $Z_1 = (X_1, Y_1), \ldots, Z_n = (X_n, Y_n)$ with

$$Y_i = f_0(X_i) + \varepsilon_i, \qquad 1 \le i \le n, \tag{4.1}$$

where X_i are $[0,1]^d$–valued random variables (also called *design points*) and ε_i are independent standard normal $\mathcal{N}(0,1)$ variables, and independent of the X_i's, and $f_0 : [0,1]^d \to \mathbb{R}$ an unknown function.

Typical statistical goals in this setting are

- estimating the unknown regression function f_0 from the observations
- finding estimates that behave (near–)"optimally" with respect to some criterion (e.g. minimax) over natural classes of parameters.

Let $\hat{f}(\cdot) = \hat{f}_n(Z_1, \ldots, Z_n)(\cdot)$ be an estimator of f.

The *prediction* risk in the setting of model (4.1) is defined as follows. Let T be a 'synthetic' data point, that is a variable independent of the X_i's and generated from the distribution of X_1. Let

$$R(\hat{f}, f_0) = E\left[\left(\hat{f}(T) - f_0(T)\right)^2\right] = E\left[\left(\hat{f}(Z_1, \ldots, Z_n)(T) - f_0(T)\right)^2\right]. \tag{4.2}$$

The interpretation is as follows: from the observ ations $Z_1, \dots, Z_n$, one builds an estimator $\hat{f}(Z_1, \dots, Z_n)(\cdot)$, and then evaluate its performance 'on average' for the prediction of the value of f_0 at a new point T. More explicitly, since $Z_1, \dots, Z_n$ are iid with distribution P_{Z_1} the law of Z_1, and T has same law as X_1,

$$R(\hat{f}, f_0) = \int \cdots \int \left(\hat{f}(z_1, \dots, z_n)(t) - f_0(t)\right)^2 dP_{Z_1}^{\otimes n}(z_1, \dots, z_n) dP_{X_1}(t).$$

The results of the chapter also hold for the following 'empirical' risk, where the L^2–norm is replaced by the empirical L^2–norm:

$$\hat{R}(\hat{f}, f_0) = E[\|\hat{f} - f_0\|_n^2] = E\left[\frac{1}{n}\sum_{i=1}^{n}(\hat{f}(X_i) - f_0(X_i))^2\right], \tag{4.3}$$

with $\hat{f}(X_i) = \hat{f}(X_i)$ the value of $\hat{f}$ at point X_i, that is, if one uses a fully explicit notation, $\hat{f}(Z_1, \dots, Z_n)(X_i)$ (note that unlike the notation could perhaps suggest, the quantity $\hat{R}(\hat{f}, f_0)$ is non–random). Results for this risk are obtained as byproduct of the proofs below.

4.2 Near-optimal estimation rates with ReLU–DNNs

Class of smooth functions. Let us first recall that a function g is β–Hölder on $[0,1]$, with $\beta \in (0,1]$, if

$$\sup_{x,y\in[0,1],\, x\neq y} \frac{|g(x) - g(y)|}{|x - y|^\beta} < \infty.$$

Next to get appropriate 'balls' of Hölder functions, one bounds the ratio in the previous display by a finite constant, say K, as well as bounds the supremum norm of g, again by K, or the sum of the two quantities by K (without this second constraint one could add an arbitrary constant to g, while in practice it seems reasonable to assume that g is bounded). We summarise these constraints by now giving the general definition of a Hölder ball in dimension d that we consider in the sequel.

A Hölder ball of functions on $[0,1]^d$ is defined as, for $\beta > 0$ and $K > 0$,

$$\mathscr{C}^\beta([0,1]^d, K) = \left\{ f : [0,1]^d \to \mathbb{R} : \sum_{\boldsymbol{\alpha}:\, \sum_{i=1}^r \alpha_i < \beta} \|\partial^{\boldsymbol{\alpha}} f\|_\infty + \sum_{\boldsymbol{\alpha}:\, \|\boldsymbol{\alpha}\|_1 = \lfloor\beta\rfloor} \sup_{x,y\in[0,1]^d,\, x\neq y} \frac{|\partial^{\boldsymbol{\alpha}} f(x) - \partial^{\boldsymbol{\alpha}} f(y)|}{\|x - y\|_\infty^{\beta - \lfloor\beta\rfloor}} \leq K \right\},$$

where $\lfloor\beta\rfloor$ *here* denotes the largest integer strictly smaller than β (so that $\lfloor 1\rfloor = 0$) and where a multi-index notation is used for the partial derivative $\partial^{\boldsymbol{\alpha}} f = \partial^{\alpha_1} \cdots \partial^{\alpha_d}$, with $(\alpha_1, \dots, \alpha_d) \in \mathbb{N}^d$.

Target rate. The notion of *optimal estimation rate* is defined with respect to the minimax criterion and the prediction risk. The minimax risk over a class $\mathscr{C}$ of functions – for instance $\mathscr{C} = \mathscr{C}_d^\beta([0,1]^d, K)$ – for estimation of f in model (4.1) is defined as

$$R_M = \inf_{\hat{f}} \sup_{f_0 \in \mathscr{C}} E[(\hat{f}(T) - f_0)^2],$$

where the infimum is over all possible estimators of f_0 in model (4.1).

It is a standard result that when $\mathscr{C} = \mathscr{C}_d^\beta([0,1]^d, K)$, then as $n \to \infty$,

$$R_M \asymp n^{-\frac{2\beta}{2\beta + d}}.$$

We refer to a course on nonparametric estimation for more details, where this rate is typically established for regression and related models such as density estimation. There are several popular classes

of estimators that achieve this global rate (sometimes up to logarithmic factors), such as wavelet estimators, kernel estimators, Bayesian posterior distributions etc.

One may note that the rate becomes very slow if d is large (unless perhaps for very high smoothness levels; for example if one imagines that d is allowed to grow as $(\log n)$, then the rate becomes constant). This is sometimes referred to as the *curse of dimensionality*. This is not a problem that comes from the use of a particular estimation method: *any* method will have a slow rate in dimension d large, when applied with certain 'un-favourable' true regression functions. This problem rather comes from the fact that the Hölder class becomes very massive as the dimension d increases. We will come back to the *curse of dimensionality* later in the chapter, where it will be seen that if the true function f_0, although defined on the whole of $\mathbb{R}^d$, has an actual small 'effective' dimension, the deep network estimator can naturally adapt to this smaller dimension.

4.2.1 Deep ReLU estimator

A natural idea to find an estimator of the regression function f in model (4.1) is to do 'maximum likelihood'. Since the errors ε_i are independent normal, this is the same as 'doing least squares'. So, given a class $\mathscr{F}$ of functions to be chosen below and observations $(X_i, Y_i)_i$ from model (4.1), let us set

$$\hat{f}^{ERM} = \hat{f}^{ERM}(\mathscr{F}) = \operatorname*{argmin}_{f\in\mathscr{F}} \frac{1}{n}\sum_{i=1}^{n}(Y_i - f(X_i))^2. \tag{4.4}$$

That is, one wants to find the 'best fit' to the data over the class $\mathscr{F}$ in terms of least squares. This estimator is called the Empirical Risk Minimiser over the class $\mathscr{F}$ (in short ERM).

Let us immediately note that in practice one often does not know an exact minimiser as in (4.4) but one can only compute some approximation of it $\tilde{f}$. How 'far' this practical estimator is from the ERM can be measured through

$$\Delta_n(\tilde{f}, f_0) = E_{f_0}\left[\frac{1}{n}\sum_{i=1}^{n}(Y_i - \tilde{f}(X_i))^2 - \inf_{f\in\mathscr{F}}\frac{1}{n}\sum_{i=1}^{n}(Y_i - f(X_i))^2\right].$$

Within this Chapter we consider the properties of the ERM itself, i.e. we assume that we have access to the ERM (or of a good enough approximation thereof, so that the term Δ_n is negligible compared to the main convergence rate term). More discussion about the term Δ_n will be given within the Interpolation Chapter of this course. Below we set $a\wedge b = \min(a,b)$ and $a\vee b = \max(a,b)$.

Definition 4.1 (Classes of DNN networks)**.** *Consider a NN Φ with input dimension d, output dimension 1, depth L, width vector $N = (N_l)_{1\le l\le L}$ and activation ρ*

$$\Phi = ((A_1, b_1), \dots, (A_L, b_L))$$

Define a 'ball' of network realisations, with parameters bounded by 1, as

$$\mathscr{F}(L, N) = \big\{f = R(\Phi),\ \text{for some } (A_j)_{1\le j\le L}, (b_j)_{1\le j\le L},\quad \max_{1\le j\le L}\big(\|A_j\|_\infty \vee |b_j|_\infty\big) \le 1\big\}.$$

We also set, for $s > 0$ a sparsity parameter,

$$\mathscr{F}(L, N, s, F) = \Big\{f \in \mathscr{F}(L, N),\quad \sum_{j=1}^{L}(\|A_j\|_0 + |b_j|_0) \le s,\quad \|f\|_\infty \le F\Big\}$$

Definition 4.2 (Deep ReLU estimator)**.** *For $\mathscr{F} = \mathscr{F}(L, N, s, F)$ as in Definition 4.1 for some $L, N, s, F > 0$, let us set*

$$\hat{f} = \hat{f}^{ReLU} = \operatorname*{argmin}_{f\in\mathscr{F}} \frac{1}{n}\sum_{i=1}^{n}(Y_i - f(X_i))^2. \tag{4.5}$$

That is, $\hat{f}$ is the ERM with optimisation over the set $\mathscr{F}(L, N, s, F)$ of s–sparse ReLU neural networks.

We assume here that the network parameters are bounded in absolute value by 1. Another positive constant could be used. The rationale behind this choice is that in practice most of the time network parameters are initialized using bounded parameters. It is also typically observed that even after training the parameters of the network remain quite close in range to initial parameters. From the theoretical point of view, some approximation results are quite easily reachable if network parameters are allowed to be very large. But in order to be closer to practical applications where parameters typically remain bounded, we restrict ourselves to that case, and we will see below that this does not prevent us to obtain good inference properties.

4.2.2 Global convergence rate for Hölder functions

The following result shows that deep ReLU neural network can achieve the optimal rate $\varepsilon_n^2 = n^{-2\beta/(2\beta+d)}$ in terms of prediction loss in regression, up to logarithmic factors, provided the parameters of the class of networks $\mathscr{F}$ over which the ERM is computed (4.5) are well–chosen.

Theorem 4.3. *Suppose the true unknown $f_0 \in \mathscr{C}^\beta([0,1]^d, K)$ for an arbitrary $\beta > 0$ and $K > 0$. Let $\hat{f} = \hat{f}^{ReLU}$ be the estimator in (4.5) with $\mathscr{F} = \mathscr{F}(L, N, s, F)$ the class of realisations of neural networks with depth L, width vector $N = (N_l)_{1\le l\le L}$, sparsity s and uniform bound F. Suppose $F \ge K \vee 1$ and a choice of parameters as follows*

$$\log_2(4\beta)\log_2(n) \le L \le n^{\frac{d}{2\beta+d}}, \qquad n^{\frac{d}{2\beta+d}} \le \min_{1\le l\le L} N_l \le \max_{1\le l\le L} N_l \le n^2, \qquad s \asymp (\log n) n^{\frac{d}{2\beta+d}}.$$

Then there exists $C = C(q, d, \beta, F)$ such that

$$\sup_{f_0 \in \mathscr{C}^\beta([0,1]^d, K)} R(\hat{f}, f_0) \le CL(\log n)^2 n^{-\frac{2\beta}{2\beta+d}}.$$

Let us now discuss possible choices of L, N, s in more details. From Theorem 4.3 it appears that the network should not be too deep, as the depth is in factor of the target rate. On the other hand, it will be clear from the proof that a depth of at least logarithmic order allows for approximation of smooth functions by the ReLU network realisation at a polynomial rate in n^{-1} (see e.g. Lemmas 4.8 and 4.9). So it seems natural to choose $L \asymp \log n$. The widths of the network, on the other hand, can be chosen to be polynomial in n, for instance one can take $N_l \asymp n$ (the upper–bound n^2 can be replaced by any power of n). Finally, the perhaps most important parameter in terms of sensibility of the rate is the sparsity s: it should be finely tuned in terms of the true regularity β of f_0 as $n^{1/(2\beta+d)}$ (up to a log factor). This parameter plays a similar role as e.g. the 'bandwidth' parameter of a kernel estimator: note that in fact the choice is similar to the one of an optimal bandwidth for kernel regression.

Corollary 4.4. *Under the same setting as Theorem 4.3, if one optimises over networks with depth $L \asymp \log n$, all widths $N_l \asymp n$ and sparsity $s \asymp (\log n) n^{\frac{1}{2\beta+1}}$, then the ERM $\hat{f}$ in (4.5) verifies*

$$\sup_{f_0 \in \mathscr{C}^\beta([0,1]^d, K)} R(\hat{f}, f_0) \lesssim (\log n)^3 n^{-\frac{2\beta}{2\beta+d}}.$$

Note that statistically, the previous results are still 'non-adaptive', that is, in order to use the previous parameters one needs to know the regularity of f_0, which is rarely the case in practice. One may envision coupling the ERM with one of the adaptation methods from the standard nonparametrics toolbox, for instance a penalisation method. Alternatively, one could use a Bayesian method (instead of the ERM), for which adaptation (in L^2 losses) is often relatively straightforward to obtain.

Let us now underline remarkable consequences of these results. We have seen in Chapter 1 that deep ReLU networks realisations are Lipschitz functions, and it has been seen in Chapter 2 that deep ReLU networks have good approximation properties over the class of Lispchitz functions.

- *Smoothness.* We see that deep ReLU empirical risk minimisers can attain (up to a poly-log factor) the optimal minimax rate for Hölder classes for *any* smoothness parameter $\beta > 0$, not only regularities between 0 and 1, as would be the case for regular histogram estimators, or between 0 and 2 as would be the case for classical piecewise affine 'local polynomial' estimators. The flexibility of deep ReLU networks comes from the compositional structure: with a small number of compositions, it enables to estimate quickly smooth functions, although the overall resulting realisation is still piecewise constant.
- *adaptation to hidden structures.* The convergence rate obtained in Theorem 4.3 is only an upper-bound, which can be quite pessimistic; it turns out that in many cases, the actual rate attained by the deep ReLU estimator is much faster, at least if the 'intrinsic' dimensionality of the regression function is smaller than the ambient space dimensionality d. This will be studied in Section 4.4.
- *weights all between* 0 *and* 1. The optimisation set $\mathscr{F}$ for the network parameters assumes bounded weights, between 0 and 1: it is interesting to see that a near-optimal estimation rate can be achieved without taking weights possibly growing with the sample size.

4.2.3 Key ideas

The proof of Theorem 4.3 is based on two important sub-results. The first is of deterministic nature and deals with approximation of smooth functions through ReLU DNNs.

Theorem 4.5 (Approximation of smooth functions by DNNs)**.** *Let $f \in \mathscr{C}_d^\beta([0,1]^d, K)$ be a function of regularity $\beta > 0$. Let $m, \mathcal{N} \geq 1$ be two integers. There exists a network, with $\Lambda := 6(d + \lceil\beta\rceil)\mathcal{N}$,*

$$\tilde{f} \in \mathscr{F}(L, (d, \Lambda, \dots, \Lambda, 1), s, \infty)$$

with depth and sparsity verifying, for c_0, C_0 depending on d, β only,

$$L = C_0 m, \qquad s \leq c_0 m \mathcal{N},$$

such that, for $c_1, c_2, \mathcal{N}_0$ depending on d, β, K only, and all $\mathcal{N} \geq \mathcal{N}_0$,

$$\|\tilde{f} - f\|_\infty \leq c_1 \frac{\mathcal{N}}{4^m} + c_2 \mathcal{N}^{-\frac{\beta}{d}}.$$

The second result is a general oracle inequality that is valid for any empirical risk minimiser. We apply it below to the DNN estimator $\hat{f}$ in (4.5).

Theorem 4.6 (Lemma 4 in [SH20b])**.** *Let $\mathscr{F}$ be a class of functions from $[0,1]^d$ to $[-F, F]$ (for some $F \geq 1$) and $\hat{f}$ be the empirical risk minimizer over this class in the regression model (4.1), that is,*

$$\hat{f} \in \operatorname*{argmin}_{f \in \mathscr{F}} n^{-1} \sum_{i=1}^n (Y_i - f(X_i))^2.$$

Then we have for any $f_0 : [0,1]^d \to [-F, F]$, for all $\delta, \varepsilon > 0$,

$$E\left[(\hat{f}(T) - f_0(T))^2\right] \leq (1+\varepsilon)^2 \left[\inf_{f \in \mathscr{F}} E\left[(f(T) - f_0(T))^2\right] + F^2 \frac{18 \log \mathcal{N}_n(\delta) + 72}{n\varepsilon} + 32\delta F\right],$$

for which $\mathcal{N}_n(\delta) := \mathcal{N}(\delta, \mathscr{F}, \|\cdot\|_\infty)$ and assuming $3 \leq \mathcal{N}_n(\delta) \leq e^n$.

The upper-bound for the prediction risk in Theorem 4.6 can be interpreted as displaying a bias–variance trade–off. The first term corresponds to *bias*: it measures the best possible approximation from the given candidate class $\mathscr{F}$. The term displaying the entropy can be interpreted as complexity or 'variance', measuring the complexity of the class $\mathscr{F}$ over which the ERM is taken.

Both results will turn useful again when analysing different assumptions on the function class the true f_0 belongs to in Section 4.4.

4.3 Proof of the global rate theorem

Let us first see how Theorem 4.3 follows by combining Theorems 4.5 and 4.6. The later oracle inequality also needs a quantitative bound on the entropy, which is provided in Lemma 4.13 below.

One applies Theorem 4.6 with $\mathscr{F} = \mathscr{F}(L, N, s, F)$ the class of DNN realisations from Theorem 4.3 with parameters as specified in the statement. One can bound

$$\inf_{f\in\mathscr{F}} E\left[(f(T)-f_0(T))^2\right] \leq \inf_{f\in\mathscr{F}} \|f-f_0\|_\infty^2.$$

Setting $\delta = 1/n, \varepsilon = 1$ and using Theorem 4.6,

$$E\left[(\hat{f}(T)-f_0(T))^2\right] \lesssim \inf_{f\in\mathscr{F}} \|f-f_0\|_\infty^2 + F^2\frac{18\log \mathscr{N}_n(1/n)+72}{n} + 32\frac{F}{n}.$$

Using Lemma 4.13 and recalling $V \leq (2\max N_l)^L \leq (2n^2)^L$ via (4.15) and the conditions on N_ls,

$$\log \mathrm{N}(\delta, \mathscr{F}(L,N,s), \|\cdot\|_\infty) \leq (s+1)\log\left(\frac{2LV^2}{\delta}\right) \leq 2s[\log(2Ln) + 2L\log(2n^2)],$$

which is bounded by $CsL\log n$. Now to control the term with the infimum above, we apply Theorem 4.5 with $\mathscr{N} \asymp n^{\frac{d}{2s+d}}$, $m = \lfloor \log_2 n \rfloor$. There exists a network in $\mathscr{F}(L,(d,\Lambda,\dots,\Lambda,1),s,\infty)$, with $L \asymp m \asymp \log n$ and $s \lesssim m\mathscr{N} \asymp (\log n) n^{\frac{d}{2s+d}}$ such that its realisation $\tilde{f}$ verifies

$$\|\tilde{f}-f_0\|_\infty \lesssim \frac{\mathscr{N}}{4^m} + \mathscr{N}^{-\frac{\beta}{d}} \lesssim n^{-\frac{\beta}{2\beta+d}}.$$

Putting the previous bounds together leads to

$$E\left[(\hat{f}(T)-f_0(T))^2\right] \lesssim n^{-\frac{2\beta}{2\beta+d}} + \frac{sL\log n}{n} + \frac{1}{n} \lesssim L(\log n)^2 n^{-\frac{2\beta}{2\beta+d}},$$

which concludes the proof of Theorem 4.3.

4.3.1 Ingredient 1: smooth approximation with deep ReLU networks

Let us now prove Theorem 4.5. The general idea is as follows: there are two main steps. The first is not specific to DNNs and is that any β–Hölder function can be well–approximated locally, using Taylor expansions, by a polynomial of order $\lfloor\beta\rfloor$: one can approximate f_0 by a piecewise polynomial function, with a quality of approximation that depends on β. The second idea, where the choice of activation function σ comes in, is that it is possible to approximate quickly, in one dimension, the monomial $(x,y) \to xy$ using a ReLU network. From there one then shows that ReLU networks suitably approximate products $x \to x_1\cdots x_d$ as well as more general monomials. From monomials one can easily approximate polynomials by combining networks, and now one can connect to the first part of

the argument, by constructing a network that approximates the piecewise polynomial function mentioned above, that itself approximates f_0.

Approximation, Step 1. In order to approximate a function $f \in \mathscr{C}_d^\beta([0,1]^d, K)$, we define a grid of $[0,1]^d$ as, for $M \geq 1$ an integer to be chosen below,

$$D(M) = \left\{ x_l = \left(\frac{l_j}{M}\right)_{j=1,\dots,d}, \quad l = (l_1, \dots, l_d) \in \{0,1,\dots,M\}^d \right\}.$$

Around a given point $\boldsymbol{a} \in [0,1]^d$, the function f can be approximated by its Taylor polynomial: in dimension d its expression is, for $\boldsymbol{a} = (a_1, \dots, a_d)$, denoting $\alpha! = \alpha_1 \cdots \alpha_d$, and $u^\alpha = u_1^{\alpha_1} \cdots u_d^{\alpha_d}$ for $u \in \mathbb{R}^d$,

$$P_{\boldsymbol{a}}^\beta f(x) := \sum_{0 \leq |\alpha| < \beta} (\partial^\alpha f)(\boldsymbol{a}) \frac{(x-\boldsymbol{a})^\alpha}{\alpha!}. \tag{4.6}$$

Taylor's expansion with Lagrange remainder gives, for any $f \in \mathscr{C}_d^\beta([0,1]^d, K)$

$$|f(x) - P_{\boldsymbol{a}}^\beta f(x)| \leq K \|x - \boldsymbol{a}\|_\infty^\beta. \tag{4.7}$$

Let us check (4.7). By Taylor's formula there exists $\xi \in [0,1]$ such that

$$f(x) = \sum_{0 \leq |\alpha| < \beta - 1} (\partial^\alpha f)(\boldsymbol{a}) \frac{(x-\boldsymbol{a})^\alpha}{\alpha!} + \sum_{\beta - 1 \leq |\alpha| < \beta} (\partial^\alpha f)\left(\boldsymbol{a} + \xi(x-\boldsymbol{a})\right) \frac{(x-\boldsymbol{a})^\alpha}{\alpha!},$$

so substracting (4.6) and using the triangle inequality gives

$$\begin{aligned} |f(x) - P_{\boldsymbol{a}}^\beta f(x)| &\leq \sum_{\beta - 1 \leq |\alpha| < \beta} \left| (\partial^\alpha f)\left(\boldsymbol{a} + \xi(x-\boldsymbol{a})\right) - (\partial^\alpha f)(\boldsymbol{a}) \right| \frac{\|x-\boldsymbol{a}\|_\infty^{|\alpha|}}{\alpha!} \\ &\leq K|\xi| \|x-\boldsymbol{a}\|_\infty^{\beta - \lfloor\beta\rfloor} \sum_{\beta - 1 \leq |\alpha| < \beta} \frac{1}{\alpha!} \|x-\boldsymbol{a}\|_\infty^{\lfloor\beta\rfloor} \leq K \|x-\boldsymbol{a}\|_\infty^\beta, \end{aligned}$$

using the fact that f is β–Hölder.

Define, again for any $f \in \mathscr{C}_d^\beta([0,1]^d, K)$ and $x = (x_1, \dots, x_d)$,

$$P^\beta f(x) := \sum_{x_l \in D(M)} (P_{x_l}^\beta f)(x) \prod_{j=1}^d (1 - M|x_j - x_{l,j}|)_+. \tag{4.8}$$

Inside the hypercubes defined by consecutive gridpoints, $P^\beta f(x)$ is a polynomial, so the overall function $P^\beta f$ is piecewise–polynomial.

Lemma 4.7 (Approximation of f by a piecewise–polynomial function)**.** *For any $f \in \mathscr{C}_d^\beta([0,1]^d, K)$, define $P^\beta f$ as in (4.8). Then*

$$\|f - P^\beta f\|_\infty \leq K M^{-\beta}.$$

Proof. Observe the following sum-product formula (expand the middle term)

$$\sum_{x_l = (l_1/M, \dots, l_d/M)} \prod_{j=1}^d (1 - M|x_j - x_{l,j}|)_+ = \prod_{j=1}^d \sum_{l=0}^M (1 - M|x_j - l/M|)_+ = 1.$$

Indeed, if $l^* = \lfloor Mx_j \rfloor$, then $(1 - M|x_j - l/M|)_+$ is possibly non-zero only for $j = l^*, l^*+1$ and

$$(1 - M|x_j - l^*/M|)_+ + (1 - M|x_j - (l^*+1)/M|)_+ = 1 - M(x_j - l^*/M) + 1 - M((l^*+1)/M - x_j) = 1.$$

One notes that the terms of the sum in the definition (4.8) are nonzero only at a given x for x_l such that $\|x - x_l\|_\infty \le 1/M$ – the corners of the hypercube of radius $1/M$ the point x belongs to –, otherwise the product in (4.8) is zero. Denoting by $\mathscr{H}_{x_l}(x) = \prod_{j=1}^d (1 - M|x_j - x_{l,j}|)_+$,

$$\left|f(x) - P^\beta f(x)\right| \le \sum_{x_l} |f(x) - (P^\beta_{x_l} f)(x)| \mathscr{H}_{x_l}(x) \le \max_{\|x - x_l\|_\infty \le 1/M} |f(x) - (P^\beta_{x_l} f)(x)| \sum_{\|x - x_l\|_\infty \le 1/M} \mathscr{H}_{x_l}(x) \le K M^{-\beta},$$

using Taylor's approximation (4.7), which concludes the proof. □

Approximation, Step 2 (specific to ReLU activation).

Lemma 4.8 (Approximating $x(1-x)$ with piecewise affine functions)**.** *Let $T^1 : [0,1] \to [0,1/4]$ and more generally $T^k : [0, 2^{-2(k-1)}] \to [0, 2^{-2k}]$, $k \ge 1$, be the maps*

$$T^1(x) = \frac{x}{2} \wedge \left(\frac{1}{2} - \frac{x}{2}\right), \quad T^k(x) = \frac{x}{2} \wedge \left(\frac{1}{2^{2k-1}} - \frac{x}{2}\right).$$

Let us set $R^k := T^k \circ T^{k-1} \circ \cdots \circ T^1$, for $k \ge 1$. Then for any $m \ge 1$,

$$\left| x(1-x) - \sum_{k=1}^m R^k(x) \right| \le 4^{-m-1}.$$

Proof. Let $C(x) = x(1-x)$. The key is to observe the 'fractal'-like property

$$C(x) = T^1(x) + \frac{1}{4} C(4T^1(x)).$$

This can be seen on a picture or just checking algebraically. Next note that by definition $T^2(y) = \frac{1}{4} T^1(4y)$ and more generally $T^{k+1}(y) = \frac{1}{4^k} T^1(4^k y)$. By recursion one immediately obtains

$$C(x) = T^1(x) + T^2 \circ T^1(x) + \cdots + T^k \circ \cdots \circ T^1(x) + \frac{1}{4^k} C(4^k T^k \circ \cdots \circ T^1(x)).$$

The result follows by applying this with $k = m$ and noting that $C(\cdot)$ is bounded by $1/4$ on $[0,1]$. □

Lemma 4.9 (Approximating $(x,y) \to xy$ by a DNN)**.** *Let $m \ge 1$. There exists a DNN, $Mult_m(x,y)$, with*

$$Mult_m \in \mathscr{F}(m+4, (2, 6, \cdots, 6, 2, 2, 2, 1)),$$

such that for any $x, y \in [0,1]$ it holds $Mult_m(x,y) \in [0,1]$, $Mult_m(0,y) = Mult_m(x,0) = 0$ and

$$\left|Mult_m(x,y) - xy\right| \le 4^{-m}.$$

Proof. In order to approximate $(x,y) \to xy$, we use a 'polarisation' formula. The most classical polarisation writes xy in terms of squares as $xy = (x+y)^2/4 - (x-y)^2/4$. Here we rather use, since from Lemma 4.8 we have access to $x(1-x) =: C(x)$, the formula

$$xy = C\left(\frac{x-y+1}{2}\right) - C\left(\frac{x+y}{2}\right) + \frac{x+y}{2} - \frac{1}{4},$$

(verify it!) where we know how to approximate every element on the right hand–side by affine functions. Denoting $C_m := \sum_{k=1}^{m} R^k$, Lemma 4.9 gives $\|C_m - C\|_\infty \le 4^{-m-1}$. Let us further denote

$$Z_m(x,y) := \left(C_m\left(\frac{x-y+1}{2}\right) + \frac{x+y}{2} - C_m\left(\frac{x+y}{2}\right) - \frac{1}{4} \right)_+ \wedge 1. \tag{4.9}$$

Then, since the map $(u,v) \to \Delta(u,v) := (u-v)_+ \wedge 1$ is 1–Lipschitz, we have

$$|Z_m(x,y) - xy| \le 2\|C - C_m\|_\infty \le 4^{-m}/2.$$

Let us set $\mathrm{Mult}_m(x,y) := Z_m(x,y)$ and see how to practically implement this into a ReLU network.

Basic networks for T_+, T_-^k, T^k, Δ. Recall $T^k(x) = (x/2) \wedge (2^{1-2k} - x/2)$. Since $a \wedge b = a_+ - (a-b)_+$ if $a \ge 0$,

$$T^k(x) = T_+(x) - T_-^k(x), \qquad T_+(x) := (x/2)_+, \quad T_-^k(x) := (x - 2^{1-2k})_+.$$

So, the functions T_+, T_- and T^k for $k \ge 1$ can all be written with a shallow network using only one neuron, as shown in Figure 4.1. Finally, using that $\Delta(u,v) = 1 \wedge (u-v)_+ = 1 - (1-(u-v)_+)_+$, one can easily encode Δ using a ReLU network with two hidden layers.

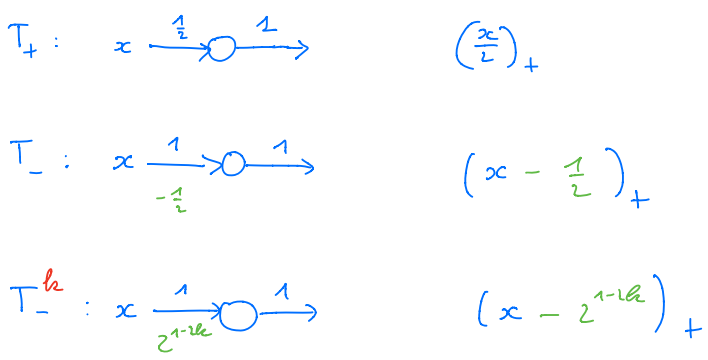

Figure 4.1: DNN representation of basic T functions encoding (on top of arrows: matrix coefficients; below arrows in green: non-zero translations; circle: neuron pass)

Combining basic networks to compute $Z_m(x,y)$. The basic networks can then be combined according to the network depicted in Figure 4.2, which taking as input $(T_+(x), h(x), T_-(x))$, computes $C_m(x) + h(x)$, where $h : [0,1] \to [0,\infty)$ is a given function taking nonnegative values. Note that the neuron pass on the middle row in Figure 4.2 is just the identity, as the input is always nonnegative.

It is then enough to run two sub-networks in parallel: a first computes

$$(x,y) \to \left(T_+\left(\frac{x-y+1}{2}\right), \frac{x+y}{2}, T_-\left(\frac{x-y+1}{2}\right) \right)$$

and applies the network N_m from Figure 4.2, thus computing $C_m\left(\frac{x-y+1}{2}\right) + \frac{x+y}{2}$. A second sub-network does the same after first computing

$$(x,y) \to \left(T_+\left(\frac{x-y}{2}\right), \frac{1}{4}, T_-\left(\frac{x+y}{2}\right) \right),$$

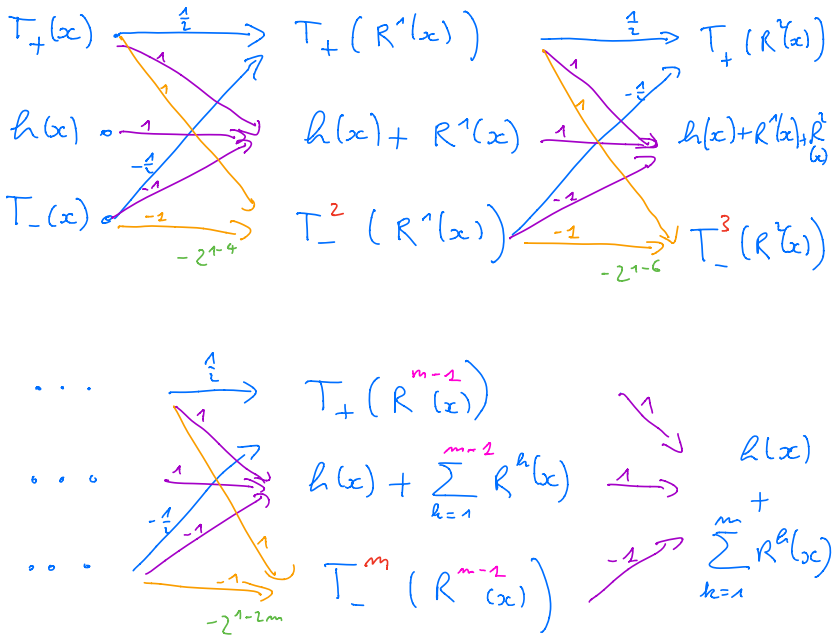

Figure 4.2: Encoding approximation of $h(x)+C(x)$, with $C(x)=x(1-x)$ with ReLU (on top of arrows: matrix coefficients; below arrows in green: non-zero translations; formulas display computed quantity just after neuron pass, except last one for final output, for which we do not apply a neuron)

thus computing $C_m\left(\frac{x+y}{2}\right)+\frac{1}{4}$. Finally applying Δ to the two sub-results provides a computation of $Z_m(x,y)$ as required.

To conclude the proof, one checks that the number of parameters is as required, and that the function $\mathrm{Mult}_m(x,y)$ is indeed 0 for inputs of the form $(x,0)$ and $(0,y)$ (this is checked on successive functions $R^1,R^2,\dots,R^m$ and left as an exercise). □

More generally, the next lemma shows, first, how to construct a DNN approximating the product $x_1\cdots x_r$. Second, one can similarly approximate any monomial, that is a polynomial of the form $x_1^{\alpha_1}\cdots x_r^{\alpha_r}$, and by synchronising the resulting networks, all monomials up to a certain degree γ *simultaneously*. Let us write, for f a $\mathbb{R}^{N_L}$–valued function, $\|f\|_\infty$ as a shorthand for $\||f|_\infty\|_\infty$.

Lemma 4.10 (Approximating products and monomials by a DNN)**.** *Let $m,r\geq 1$ two integers. There exists a DNN, $Mult_m^r:[0,1]^r\to[0,1]$, with depth $L\lesssim m\log r$ and maximum width $6r$ such that for $x=(x_1,\dots,x_r)\in[0,1]^r$,*

$$\left|Mult_m^r(x)-\prod_{i=1}^r x_i\right|\leq r^2 4^{-m},$$

and $Mult_m^r(x)=0$ if one of the x_i's is zero. More generally, let $m,\gamma,d\geq 1$ three integers. Let $C_{d,\gamma}$ denote the number of monomials over d variables with degree $|\boldsymbol{\alpha}|<\gamma$. There exists a DNN, $Mon_{m,\gamma}^d:[0,1]^d\to[0,1]^{C_{d,\gamma}}$ with depth $L\lesssim m\log\gamma$ and maximum width $12\gamma C_{d,\gamma}$, that approximates all monomials of degree less than γ simultaneaously

$$\left\|Mon_{m,\gamma}^d(x)-\left(x^{\boldsymbol{\alpha}}\right)_{|\boldsymbol{\alpha}|<\gamma}\right\|_\infty\leq\gamma^2 4^{-m}.$$

Remark. The number $C_{d,\gamma}$ of monomials of degree less than γ is less than $(\gamma+1)^d$.

Proof. We start by the network Mult_m^r and notice that one can always assume that r is a power of 2. If this is not the case, one just artificially extends the product by multiplying by a number of 1's. So, let us set $r = 2^q$. An approximation to the product of x_i's is computed recursively as follows: first compute

$$(\mathrm{Mult}_m(x_1,x_2), \mathrm{Mult}_m(x_3,x_4), \ldots, \mathrm{Mult}_m(x_{r-1},x_r)),$$

which gives 2^{q-1} terms left, and then repeat the same operation until there is only one term left, with an output that we define as $\mathrm{Mult}_m^r(x)$. By Lemma 4.9 and the triangle inequality, for $a,b,c,d \in [0,1]$,

$$|\mathrm{Mult}_m(a,b) - cd| \le |\mathrm{Mult}_m(a,b) - ab| + |(a-c)b + (b-d)c| \le 4^{-m} + |a-c| + |b-d|.$$

An immediate recursion then gives, as announced,

$$\left|\mathrm{Mult}_m^r(x) - (x_1 \cdots x_r)\right| \le 3^{q-1}4^{-m} \le 4^q 4^{-m} = r^2 4^{-m}.$$

Now turning to the network computing all monomials, one notes that for a degree of at most 1, one can just use a shallow network, with the later computing exactly a constant or a linear function (recall the identity can be obtained as such a network). More generally, one uses the same argument as for Mult_m^r to compute a given monomial $x_1^{\alpha_1}\cdots x_r^{\alpha_r}$, up to an error, recalling $|\boldsymbol{\alpha}| = \sum_{i=1}^r \alpha_i < \gamma$, bounded from above by $\gamma^2 4^{-m}$. In a last step, we stack all obtained networks in parallel (using depth synchronisation to have the same given depth for all networks, meaning we take the largest depth).

□

End of the proof of Theorem 4.5. Now that we have constructed a network computing all monomials, one can go back to the local polynomial (4.8) approximating f, namely

$$P^\beta f(x) := \sum_{x_l \in D(M)} (P_{x_l}^\beta f)(x) \prod_{j=1}^d (1 - M|x_j - x_{l,j}|)_+.$$

Let us define M as the largest integer such that

$$(M+1)^d \le \mathcal{N}. \tag{4.10}$$

To conclude the proof, one constructs the final network in three steps.

Step (i), hat function network. One constructs a network Hat^d approximating the hat functions $\prod_{j=1}^d (M^{-1} - |x_j - x_{l,j}|)_+$ (note the specific normalisation, in order to have an easy construction with weights bounded by 1) *simultaneously* for all x_ℓ on the grid $D(M)$.

Since $|x| = x_+ + (-x)_+$, we have the formula, for a,b,c in $[0,1]$,

$$(a - |b-c|)_+ = (a - (b-c)_+ - (c-b)_+)_+.$$

One can use a first hidden layer with width $2d(M+1)$ to compute all functions $(x_j - \ell/M)_+$ and $(\ell/M - x_j)_+$, and a second hidden layer to compute all functions $(1/M - |x_j - \ell/M|)_+$ using the formula in the last display, using this time a width $d(M+1)$. All these functions take values in $[0,1]$.

If $d = 1$ we are done (the network computes the function exactly). For $d > 1$, one uses the networks Mult_m^d from Lemma 4.10 to compute the desired products. Each one of these products requires (recalling Mult_m^d has width less than Cd, depth at most $Cm\log d$) at most $Cmd^2\log d$ nonzero parameters. We have of $(M+1)^d$ of these products in parallel which gives sparsity $C'm(M+1)^d d^2 \log d \lesssim m\mathcal{N}$ in

total (adding also the non-zero parameters of the first two layers from the previous paragraph, which require only $Cd(M+1)$ non-zero weights). The resulting network verifies

$$\Big|\mathrm{Hat}^d(x) - \Big(\prod_{j=1}^{d}(M^{-1}-|x_j-x_{l,j}|)_+\Big)_{x_l\in D(M)}\Big|_\infty \le d^2 4^{-m}. \tag{4.11}$$

Step (ii), networks Q_1 and Q_2. We now build two networks verifying the following. For $B = 3Ke^d$, we have $Q_1(x) \in [0,1]^{(M+1)^d}$ and

$$\Big|Q_1(x) - \Big(\frac{P^\beta_{x_l} f(x)}{B} + \frac{1}{2}\Big)_{x_l\in D(M)}\Big| \le \beta^2 4^{-m}. \tag{4.12}$$

The role of the constant B is to keep the approximated quantity in the last display between 0 and 1, thanks to Lemma 4.11. Next, the network Q_2 verifies

$$\Big|Q_2(x) - \sum_{x_l\in D(M)}\Big(\frac{P^\beta_{x_l} f(x)}{B} + \frac{1}{2}\Big)\prod_{j=1}^{d}(M^{-1}-|x_j-x_{l,j}|)_+\Big| \le (1+d^2+\beta^2)4^{d-m}. \tag{4.13}$$

The construction of Q_1 is immediate by forming the weighted sum of the joint network of monomials $\mathrm{Mon}^d_{m,\gamma}$ for every point $x_l \in D(M)$ in parallel (thus getting an output dimension $(M+1)^d$), noting with Lemma 4.11 that the weights are smaller than 1 thanks to the division by B. By Lemma 4.10, the depth of Q_1 is bounded by Cm. The sparsity of Q_1 is bounded by that of $\mathrm{Mon}^d_{m,\gamma}$ plus $C_{d,\beta}(M+1)^d$, that is by $Cm + C(M+1)^d \le C(m+\mathcal{N})$, where C depends only on d,β.

To build Q_2, one proceeds on two steps: first, one stacks in parallel the networks Q_1 and the Hat^d network (which simultaneously outputs all hat functions). Next, one notes that both Q_1 and Hat^d have outputs indexed by x_l. One pairs these outputs two-by-two and applies to them the Mult_m network. There are $(M+1)^d \le \mathcal{N}$ pairs, and recall that Mult_m has of order m active (nonzero) parameters (depth m and constant width). So this part of the network has sparsity at most $m\mathcal{N}$. Finally, one adds all results using a final layer, leading to the term approximating Q_2 in (4.13), except that hat functions and local polynomial are replaced by their approximations. Combining (4.11) with (4.12) now gives (4.13). In passing we note that building C_2 uses at most $Cm\mathcal{N}$ nonzero parameters, with depth of order m.

Step (iii), shifting and rescaling the entries in (4.13). Finally, we build a network Q_3 that verifies

$$\Big|Q_3(x) - \sum_{x_l\in D(M)} P^\beta_{x_l} f(x)\prod_{j=1}^{d}(1-M|x_j-x_{l,j}|)_+\Big| \le (2K+1)(1+d^2+\beta^2)(2e)^d\mathcal{N}4^{-m}. \tag{4.14}$$

The construction of Q_3 is based on shifting/rescaling Q_2. One needs to pay attention that we wish to keep weights between 0 and 1. Note that to compute Q_3 by putting Q_2 'on the right scale', it suffices to build a network computing the scaling $x \to BM^r x =: \mathcal{K}x$. To do so, one may use a network with a weight matrix having all entries equal to 1 and zero shift vectors, which uses $C\mathcal{K} \lesssim \mathcal{N}$ active parameters. The overall network Q_3 thus keeps up to a constant the same sparsity (at most $Cm\mathcal{N}$) and depth as Q_2. The approximation (4.14) directly follows from (4.13).

Putting (4.14) together with Lemma 4.7, one obtains that the network Q_3 verifies

$$\|f - Q_3\|_\infty \lesssim \mathcal{N}4^{-m} + M^{-\beta} \lesssim \mathcal{N}4^{-m} + \mathcal{N}^{-\beta/d},$$

with a sparsity $Cm\mathcal{N}$ (we omit the precise dependence of the constants on K,d given in the statement, which can easily be tracked in the above arguments) which concludes the proof of Theorem 4.5.

Lemma 4.11. *Let $P^\beta_{\boldsymbol{\alpha}} f$ the polynomial defined in (4.6). Writing $P^\beta_{\boldsymbol{\alpha}} f(x) = \sum_{0\le|\gamma|\le\beta} c_\gamma x^\gamma$, for any $f \in \mathscr{C}^\beta_d([0,1]^d, K)$,*

$$\sup_{x\in[0,1]^d} \left|(P^\beta_{\boldsymbol{\alpha}} f)(x)\right| \le \sum_{0\le|\gamma|\le\beta} |c_\gamma| \le K e^d.$$

Proof. Left as an exercise, see [SH20a], page 6. □

4.3.2 Ingredient 2 : entropy and error propagation in DNNs

Let ρ be the ReLU activation, or more generally a 1–Lip function with $\rho(0) = 0$. Considering the class of functions $\mathscr{F}(L, N, s)$, let us denote

$$V = V(N) := \prod_{l=0}^{L} (N_l + 1). \tag{4.15}$$

Since ρ is fixed throughout, we write $R(W)$ for $R(\Phi)$. The next lemma quantifies how much small errors in network parameters propagate into a global error for the network realisation.

Lemma 4.12. *Suppose $f = R(W)$ and $f^* = R(W^*)$ belong to $\mathscr{F}(L, N)$ with $W = (A_k, b_k)_{k=1,\dots,L}$ and $W^* = (A^*_k, b^*_k)_{k=1,\dots,L}$. Suppose that individual entries of A_k's and b_k's are at most $\varepsilon > 0$ away from the corresponding entries of A^*_k and b^*_k. Then for V as in (4.15),*

$$\|f - f^*\|_\infty \le \varepsilon L V.$$

Proof. Recall $f = T_L \circ \rho \circ \cdots \circ \rho \circ T_1$ with $T_k(x) = A_k x + b_k$ and define, for $k = 1, \dots, L$,

$$B_k f = \rho \circ T_k \circ \cdots \circ \rho \circ T_1,$$
$$E_k f = T_L \circ \rho \circ \cdots \circ T_{k+1} \circ \rho,$$

and set $E_L f = B_0 f = \mathrm{Id}$. We first prove two basic facts about $B_k f, E_k f$.

Fact 1. If $f \in \mathscr{F}(L, N)$, then $|(B_k f)(x)|_\infty \le \prod_{l=1}^k (N_{l-1} + 1)$ for $x \in [0,1]^d$.

Let us check first that $|(\rho \circ T_i)(y)|_\infty \le N_{i-1}|y|_\infty + 1$ for any integer i. Indeed, $|\rho(y)|_\infty \le |y|_\infty$ and $|T_k y|_\infty \le |A_k y|_\infty + |b_k|_\infty \le N_{k-1}|y|_\infty + 1$, using $\|A_k\|_\infty \le 1, |b_k|_\infty \le 1$. In particular, if $|y|_\infty \ge 1$ we have $|(\rho \circ T_i)(y)|_\infty \le (N_{i-1} + 1)|y|_\infty$ for any i.

The result follows by recursion: for $i = 1$ we get $|(\rho \circ T_1)(x)|_\infty \le N_0|x|_\infty + 1 \le N_0 + 1$. Since $N_0 + 1 \ge 1$ it suffices feeds this bound into the previous inequality in terms of y.

Fact 2. The map $x \to (E_k f)(x)$ is Λ_k–Lipschitz, with $\Lambda_k \le \prod_{l=k+1}^L N_{l-1}$.

The composition of an L_1–Lip by an L_2–Lip function is an $L_1 L_2$–Lip function. By definition ρ is 1–Lip, while T_i is N_{i-1}–Lip for any i, from which the fact follows.

Now let us write the difference $f - f^*$ as the telescopic sum

$$f(x) - f^*(x) = \sum_{k=1}^{L} \Big[(E_k f) \circ T_k \circ (B_{k-1} f^*)(x) - (E_k f) \circ T^*_k \circ (B_{k-1} f^*)(x) \Big].$$

Combining the triangle inequality with Fact 2 above,

$$\begin{aligned}|f(x)-f^*(x)| &\le \sum_{k=1}^{L}\Lambda_k\left|(T_k-T_k^*)\circ(B_{k-1}f^*)(x)\right|_\infty \\ &\le \sum_{k=1}^{L}\Lambda_k\left[\|A_k-A_k^*\|_\infty|(B_{k-1}f^*)(x)|_1+|b_k-b_k^*|_\infty\right] \\ &\le \sum_{k=1}^{L}\Lambda_k\left[\varepsilon N_{k-1}|(B_{k-1}f^*)(x)|_\infty+\varepsilon\right].\end{aligned}$$

The term under brackets in the last display is at most, using Fact 1,

$$\varepsilon N_{k-1}\prod_{l=1}^{k-1}(N_{l-1}+1)+1\le\varepsilon\prod_{l=1}^{k}(N_{l-1}+1).$$

One deduces the announced result

$$|f(x)-f^*(x)|\le\varepsilon\sum_{k=1}^{L}\prod_{l=1}^{L}(N_{l-1}+1)\le\varepsilon LV.$$

□

The previous lemma allows for a "quantisation" of the set of neural network realisations: in the next result we explicitly construct a finite set of functions (themselves NNs) that cover it.

Lemma 4.13. *For V as in (4.15) and any $\delta>0$,*

$$\log \mathrm{N}(\delta,\mathscr{F}(L,N,s),\|\cdot\|_\infty)\le(s+1)\log\left(\frac{2LV^2}{\delta}\right).$$

In particular if $L\lesssim\log n$ and $N_l\le n$ for any integer l, we have

$$\log \mathrm{N}(\delta,\mathscr{F}(L,N,s),\|\cdot\|_\infty)\lesssim s\left[(\log n)^2+\log(1/\delta)\right].$$

The proof has been given in Chapter 3.

4.3.3 A generic oracle inequality for the prediction risk

Let us now prove Theorem 4.6. It is helpful to relate the prediction risk $R(\hat f,f_0)=E\left[(\hat f(X)-f_0(X))^2\right]$ to the empirical risk (4.3). The proof will be complete once we show

$$R(\hat f,f_0)\le(1+\varepsilon)\hat R(\hat f,f_0)+\frac{1+\varepsilon}{\varepsilon^2}3\frac{F^2}{n}\log\mathscr{N}_n+(1+\varepsilon)\delta\frac{F}{n},\tag{4.16}$$

as well as the following direct bound on the empirical risk, for $F\ge1$,

$$\hat R(\hat f,f_0)\le(1+\varepsilon)\left\{\inf_{f\in\mathscr{F}}R(f,f_0)+3\frac{1+\varepsilon}{\varepsilon}\frac{F^2}{n}\log\mathscr{N}_n+F\delta\right\}.\tag{4.17}$$

As follows from the proofs below, the upper-bound (4.16) of the prediction risk by the 'empirical' risk holds for *any* estimator $\hat f$, not necessarily the ERM. The bound (4.17) uses crucially that $\hat f$ is the ERM.

Proof of (4.16). (a) Let us cover $\mathscr{F}$ by $N := \mathscr{N}_n$ balls of radius δ and centers $f_1, \dots, f_N$. One may assume $\|f_i\|_\infty \le F$ (otherwise consider balls centered at $\bar{f}_i = (f_i \wedge F) \vee (-F)$ instead).

Let j^* be a random integer such that $\|\hat{f} - f_{j^*}\|_\infty \le \delta$. For $\Delta = \hat{f} - f_{j^*}$ (so $\|\Delta\|_\infty \le \delta$), let us write

$$\hat{f} - f_0 = (\hat{f} - f_{j^*}) + f_{j^*} - f_0 = \Delta + f_{j^*} - f_0,$$

(b) In order to more easily compare the risks, let us note that the prediction risk may be written

$$R(\hat{f}, f_0) = E\left[\frac{1}{n}\sum_{i=1}^n (\hat{f} - f_0)^2(T_i)\right],$$

where T_i are iid variables with law $\mathscr{L}(X_i) = \mathscr{L}(X_1)$ (recall the X_i are iid). One may now write

$$R(\hat{f}, f_0) - \hat{R}(\hat{f}, f_0) = E\left[\frac{1}{n}\sum_{i=1}^n (f_{j^*} - f_0)^2(T_i) - (f_{j^*} - f_0)^2(X_i)\right] + \mathscr{R}_1,$$

where the remainder term $\mathscr{R}_1$ verifies $|\mathscr{R}_1| \le 2\delta^2 + 2 \times 4\delta F \le 10\delta F$ for small δ, which is obtained by expanding the squares and using Cauchy–Schwarz' inequality. Deduce

$$|R(\hat{f}, f_0) - \hat{R}(\hat{f}, f_0)| \le E\left|\frac{1}{n}\sum_{i=1}^n g_{j^*}(X_i, T_i)\right| + 10\delta F,$$

where we have set, for any integer $j \le \mathscr{N}_n$,

$$g_j(X_i, T_i) := (f_j - f_0)^2(T_i) - (f_j - f_0)^2(X_i).$$

(c) Let us set, for $j = 1, \dots, N$, and $a \vee b = \max(a, b)$,

$$r_j^2 = \frac{\log \mathscr{N}_n}{n} \vee E_T[(f_j - f_0)^2(T)],$$

where E_T means that one takes the expectation with respect to T (only). Let us further set

$$\mathscr{U}^2 := E_T[(\hat{f} - f)^2(T)], \qquad \mathscr{T} := \max_j \left|\sum_{i=1}^n \frac{g_j(X_i, T_i)}{r_j F}\right|.$$

Since T is independent of $(X_i, Y_i)_i$, one has $E_T[(f_j - f_0)^2(T)] = E_T[(f_j - f_0)^2(T) \,|\, (X_i, Y_i)_i]$, which allows to define r_{j^*} as r_j with j replaced by the (random) quantity j^*. Namely,

$$r_{j^*}^2 = \frac{\log \mathscr{N}_n}{n} \vee E_T[(f_{j^*} - f_0)^2(T)],$$

where the last display is random (via j^*). One may bound, using $(a+b)^2 \le 2a^2 + 2b^2$,

$$r_{j^*}^2 \le \frac{\log \mathscr{N}_n}{n} + 2E_T[(f_{j^*} - \hat{f})^2(T)] + 2\mathscr{U}^2 \le \frac{\log \mathscr{N}_n}{n} + 2\delta^2 + 2\mathscr{U}^2,$$

using that f_{j^*} is uniformly at most δ away from f_0. Then

$$\left|\sum_{i=1}^n g_{j^*}(X_i, T_i)\right| = \left|\sum_{i=1}^n \frac{g_{j^*}(X_i, T_i)}{r_{j^*} F}\right| r_{j^*} F \le \max_j \left|\sum_{i=1}^n \frac{g_j(X_i, T_i)}{r_j F}\right| r_{j^*} F = \mathscr{T}\, r_{j^*} F.$$

Combining with the above bound on r_{j^*} and since by definition of the prediction risk $E[\mathscr{U}^2] = R(\hat{f}, f_0)$,

$$\left|\sum_{i=1}^n g_{j^*}(X_i,T_i)\right|/F \le E\left[\mathscr{T}\cdot\sqrt{\frac{\log\mathscr{N}_n}{n}+2\delta^2+2\mathscr{U}^2}\right] \le \sqrt{2}E[\mathscr{T}\cdot\mathscr{U}] + \left\{\sqrt{\frac{\log\mathscr{N}_n}{n}}+\sqrt{2}\delta\right\}E[\mathscr{T}]$$
$$\le \sqrt{2E[\mathscr{T}^2]}\sqrt{R(\hat{f},f_0)} + \left\{\sqrt{\frac{\log\mathscr{N}_n}{n}}+\sqrt{2}\delta\right\}E[\mathscr{T}].$$

(d) Let us now provide bounds for $E[\mathscr{T}], E[\mathscr{T}^2]$. We start by deriving a deviation bound $P[\mathscr{T} > t]$ for $t > 0$ to be chosen later. A union bound gives, setting $Z_{ij} := g_j(X_i,T_i)/(r_jF)$,

$$P[\mathscr{T} > t] \le \sum_{j=1}^{\mathscr{N}_n} P\left[\left|\sum_{i=1}^n Z_{ij}\right| \ge t\right].$$

The variables Z_{ij} are centered and bounded in absolute value by $[(2F)^2+(2F)^2]/(r_jF) \le 8F/r_j =: M_j$. Also, using the definition of r_j,

$$\mathrm{Var}[Z_{ij}] \le \frac{2}{r_j^2F^2}E[(f_j-f_0)(X_1)^4] \le \frac{2(2F)^2}{r_j^2F^2}E[(f_j-f_0)(X_1)^2] \le 8 =: v_{ij}.$$

An application of Bernstein's inequality to the independent variables Z_{ij} gives

$$P[\mathscr{T} > t] \le \sum_{j=1}^{\mathscr{N}_n} 2\exp\left(-\frac{t^2}{2M_jt/3+2\sum_{i=1}^n v_{ij}}\right) \le 2\mathscr{N}_n\exp\left(-\frac{t^2}{\frac{16Ft}{3r_j}+16n}\right).$$

Using that $r_j \ge \sqrt{\log\mathscr{N}_n/n}$ by definition and choosing $t \ge t_1 := C_F\sqrt{n\log\mathscr{N}_n}$ for some large enough $C_F = C(F)$ leads to

$$P[\mathscr{T} > t] \le 2\mathscr{N}_n\exp\left(-Ct\sqrt{\frac{\log\mathscr{N}_n}{n}}\right).$$

From this one easily sees that $\mathscr{T}$ is of order $\sqrt{n\log\mathscr{N}_n}$. More precisely, using the formulas $E\mathscr{T} = \int_0^\infty P[\mathscr{T}\ge t]dt$ and $E\mathscr{T}^2 = \int_0^\infty P[\mathscr{T}^2\ge t]dt$, one obtains (check it as an exercise)

$$E\mathscr{T} \lesssim \sqrt{n\log\mathscr{N}_n}, \qquad E[\mathscr{T}^2] \lesssim n\log\mathscr{N}_n.$$

(e) Combining the points (b), (c), (d) above leads to

$$|\underbrace{R(\hat{f},f_0)}_{a} - \underbrace{\hat{R}_n(\hat{f},f_0)}_{b}| \le \underbrace{\frac{F}{n}\sqrt{2n\log\mathscr{N}_n}}_{2c}\underbrace{\sqrt{R(\hat{f},f_0)}}_{\sqrt{a}} + \underbrace{\frac{F}{n}\left\{\sqrt{\frac{\log\mathscr{N}_n}{n}}+\sqrt{2}\delta\right\}\sqrt{n\log\mathscr{N}_n}+10\delta F}_{d}.$$

Inequality (4.16) is obtained upon noting the following: for reals b,c,d and $a > 0$ such that $|a-b| \le 2\sqrt{a}c+d$, for any $\varepsilon > 0$ it holds

$$a \le (1+\varepsilon)(b+d) + \frac{(1+\varepsilon)^2}{\varepsilon}c^2,$$

obtained by using the inequality $\sqrt{a}c \le \frac{\varepsilon}{1+\varepsilon}a + \frac{1+\varepsilon}{\varepsilon}c^2$ (itself a variant of $ac \le (a^2+c^2)/2$).

Proof of (4.17). In the sequel we write Y to mean the vector of oberved Y_i's, and in slight abuse of notation also interpret it as the function that takes values Y_i's at X_i's (so as to evaluate it under the empirical norm $\|\cdot\|_n$). For any $f \in \mathscr{F}$, using the definition of the ERM,

$$\begin{aligned}\|\hat{f} - f_0\|_n^2 &= \|\hat{f} - Y\|_n^2 + \|Y - f_0\|_n^2 + 2\langle Y - f_0, \hat{f} - Y\rangle_n \\ &\leq \|f - Y\|_n^2 + \|Y - f_0\|_n^2 + 2\langle Y - f_0, \hat{f} - Y\rangle_n,\end{aligned}$$

where $\langle\cdot,\cdot\rangle_n$ is the inner–product associated to the empirical norm. Now using the definition of the model, $Y = f_0 + \varepsilon$, so that

$$\begin{aligned}\|f - Y\|_n^2 + \|Y - f_0\|_n^2 + 2\langle Y - f_0, \hat{f} - Y\rangle_n &= \|f - f_0\|_n^2 - 2\langle f - f_0, \varepsilon\rangle_n + \|\varepsilon\|_n^2 + \|\varepsilon\|_n^2 + 2\langle \varepsilon, \hat{f} - f_0 - \varepsilon\rangle_n \\ &= \|f - f_0\|_n^2 - 2\langle f, \varepsilon\rangle_n + 2\langle \varepsilon, \hat{f}\rangle_n.\end{aligned}$$

Combining the above inequalities, taking expectations and using $E\|f - f_0\|_n^2 = E[(f - f_0)(X_1)^2] = R(f, f_0)$,

$$\hat{R}_n(\hat{f}, f_0) = E\|\hat{f} - f_0\|_n^2 \leq R(f, f_0) + 2E\langle \varepsilon, \hat{f}\rangle_n,$$

where we have used that $\langle f, \varepsilon\rangle_n$ is centered, as $E[f(X_1)\varepsilon_1] = E[f(X_1)]E[\varepsilon_1] = 0$. To derive (4.17), it suffices to bound $E\langle \varepsilon, \hat{f}\rangle_n$, that is the expectation of an empirical process. We will bound this classically by replacing the quantity $\hat{f}$ by a maximum over a finite set of functions given to us by the entropy covering. We then conclude using a bound on the maximum in expectation (something sometimes called a "maximal inequality").

Let j^* be a random index such that $\|\hat{f} - f_j\|_\infty \leq \delta$. Let us denote, for an integer $j \leq \mathscr{N}_n$,

$$\xi_j = \frac{1}{\sqrt{n}} \sum_{i=1}^n \frac{\varepsilon_i (f_j - f_0)(X_i)}{\|f_j - f_0\|_n}.$$

This is a centered variable, whose distribution given the X_i's is standard normal. By writing

$$E\left[\max_{1 \leq j \leq \mathscr{N}_n} \xi_j^2\right] = E\left[E\left[\max_{1 \leq j \leq \mathscr{N}_n} \xi_j^2 | (X_i)_i\right]\right],$$

we see that to bound the first expectation it is enough to bound the conditional expectation on the right hand side, that is the expectation of the maximum of chi-squared(1) variables, so that using Lemma 4.14 below, the last display is at most $3\log \mathscr{N}_n + 1$.

Recalling the definition of j^* and using the triangle inequality,

$$\begin{aligned}\left|E\langle \varepsilon, \hat{f}\rangle_n\right| &= \left|\frac{1}{n} E \sum_{i=1}^n \varepsilon_i (\hat{f}(X_i) - f_0(X_i))\right| \\ &\leq \delta E \frac{1}{n} \sum_{i=1}^n |\varepsilon_i| + \left|\frac{1}{n} E \sum_{i=1}^n \varepsilon_i (f_{j^*}(X_i) - f_0(X_i))\right| \\ &\leq \delta + \frac{1}{\sqrt{n}} E\left[|\xi_{j^*}| \|f_{j^*} - f_0\|_n\right].\end{aligned}$$

One further bounds $\|f_{j^*} - f_0\|_n \leq \|f_{j^*} - \hat{f}\|_n + \|\hat{f} - f_0\|_n \leq \delta + \|\hat{f} - f_0\|_n$ and via Cauchy-Schwarz,

$$\begin{aligned}E\left[|\xi_{j^*}| \|f_{j^*} - f_0\|_n\right] &\leq \sqrt{2E\|\hat{f} - f_0\|_n^2 + 2\delta^2} \sqrt{E\left[\max_{1 \leq j \leq \mathscr{N}_n} \xi_j^2\right]} \\ &\leq \sqrt{2}\left[\sqrt{\hat{R}(\hat{f}, f_0)} + \delta\right] \sqrt{3\log \mathscr{N}_n + 1},\end{aligned}$$

where we have used the property on maxima mentioned above. Deduce

$$\left|E\langle\varepsilon,\hat{f}\rangle_n\right| \le \delta + \sqrt{2}\frac{\sqrt{4n}}{\sqrt{n}}\delta + \sqrt{2}\sqrt{\hat{R}(\hat{f},f_0)\cdot\frac{4\log\mathcal{N}_n}{n}} \le 5\delta + 4\sqrt{\hat{R}(\hat{f},f_0)\frac{\log\mathcal{N}_n}{n}},$$

where we use the assumption $1 \le \log\mathcal{N}_n \le n$. One obtains

$$\hat{R}(\hat{f},f_0) \le R(f,f_0) + 4\sqrt{\hat{R}(\hat{f},f_0)\frac{\log\mathcal{N}_n}{n}} + 5\delta.$$

To conclude, one uses a similar argument as at the end of the proof of (4.16), so that one moves both $\hat{R}(\hat{f},f_0)$ terms to the left hand side of the inequality, which concludes the proof of (4.17).

Lemma 4.14. *Let $\xi_1,\dots,\xi_N$ be standard normal variables (but not necessarily independent). Then, for all $N \ge 1$,*

$$E\left[\max_{1\le j\le N}\xi_j^2\right] \le 3\log N + 1.$$

This is standard (see [SH20a], Lemma C.1, or e.g. [BLM13] Corollary 2.6 for a more general result for sub-exponential variables); the way to understand it: for Gaussian variables the maximum is of order at most $\sqrt{\log N}$ so the squares of N Gaussians have their maximum at most of size $(\log N)$.

4.4 Compositional structures: towards solving the curse of dimensionality

Discovering a hidden 'structure'. The 'raw' regression data collected by the statistician takes the form, in the setting model (4.1), of n vectors of size $d+1$: the n pairs (X_i^T, Y_i) with $X_i \in [0,1]^d$ and Y_i a real, with the dimension d possibly large (think for instance of e.g. $d = 10$ or 20). The unknown regression function $f_0(x_1,\dots,x_d)$ depends on d of variables, and we have seen that if d is larger than a few units this may lead to a slow uniform convergence rate of the form $n^{-2\beta/(2\beta+d)}$ for the prediction risk. It is often the case though that the problem is effectively of smaller dimension than d. We give a number of frequently encountered examples

1. f_0 in fact depends on just one variable (but we do not know it a priori), for instance

   $$f_0(x_1,\dots,x_d) = g(x_1),$$

   for some $g : [0,1] \to \mathbb{R}$. In this case it seems reasonable to expect a rate $n^{-2\beta/(2\beta+1)}$, since the f_0 effectively depends on 1 variable only. More generally, f_0 may depend on a small number $t \le d$ of variables, although we do not know a priori which ones, e.g.

   $$f_0(x_1,\dots,x_d) = g(x_2,x_3,x_d),$$

   in which case the effective dimension should be 3, so we expect a rate $n^{-2\beta/(2\beta+3)}$.

2. In the preceding example, the function effectively depends on a small number of the original variables x_i, but it could depend on few variables only after transformation of the variables, for instance

   $$f_0(x_1,\dots,x_d) = g(x_1+x_2+\cdots+x_d).$$

   In this case $f_0(x_1,\dots,x_d) = g(x')$ only depends on 'one' variable $x' = x_1+\cdots+x_d$, so one expect a rate $n^{-2\beta/(2\beta+1)}$.

3. *Additive models.* It may be possible to write f_0 in an additive form

$$f_0(x_1,\dots,x_d) = \sum_{i=1}^{d} f_i(x_i),$$

for some functions $f_1,\dots,f_d$ depending on one variable only. If all functions f_i are at least β–Hölder, one expects a rate $d\cdot n^{-2\beta/(2\beta+1)}$ that is $n^{-2\beta/(2\beta+1)}$ if d is a fixed constant.

4. *Generalised additive models.* It may be possible to write f_0 in the form

$$f_0(x_1,\dots,x_d) = h\left(\sum_{i=1}^{d} f_i(x_i)\right),$$

for some real-valued functions $f_1,\dots,f_d$ (that are, as before, say all β–Hölder) and an unknown real 'link' function h that is γ–Hölder. One expects the rate to depend on β,γ, but not (too much) on the dimension d.

Class of compositions. In all the settings of the previous paragraph, one may note that the original function f_0 can be written as a composition of functions

$$f_0 = g_q\circ\cdots\circ g_1\circ g_0,$$

for some integer $q\geq 1$. For instance, in the case of additive models one can set $g_0(x_1,\dots,x_d) = (f_1(x_1),\dots,f_d(x_d))$ (note that g_0 is then $\mathbb{R}^d$–valued) and $g_1(y_1,\dots,y_d) = y_1+\cdots+y_d$. For each of the examples in the above list, *if one knew beforehand* that f_0 is in one class of the other, one could certainly develop a specific estimation method using the special structure at hand. In practice, however, it would be desirable to have a method that is able to automatically 'learn the structure'. We are going to see that this is achieved by deep ReLU estimators.

Let us introduce the class, for $d=(d_0,\dots,d_{q+1}), t=(t_0,\dots,t_q), \beta=(\beta_0,\dots,\beta_q)$,

$$\begin{aligned}\mathcal{G}(q,d,t,\beta,K) = \Big\{ f = g_q\circ\cdots\circ g_0 :\quad & g_i = (g_{ij})_j : [a_i,b_i]^{d_i}\to[a_{i+1},b_{i+1}]^{d_{i+1}},\\ & g_{ij}\in \mathcal{C}^{\beta_i}_{t_i}([a_i,b_i]^{t_i},K),\quad |a_i|,|b_i|\leq K\Big\},\end{aligned} \tag{4.18}$$

where we denoted $\mathcal{C}^{\beta_i}_{t_i}$ for the Hölder ball over t_i variables to insist on the fact that these functions depend on t_i variables only (at most). The coefficients t_i can be interpreted as the maximal number of variables each function g_{ij} is allowed to depend on. In particular, this number is always at most d_i, but may actually be much smaller. Let us note that the decomposition of f_0 as a composition is typically not unique, but this is not of concern us here because we are interested in estimation of f_0 itself only.

Note that for $f_0 = g_1\circ g_0$ with $d_1=d_0=t_1=t_0=1$ and $\beta_0,\beta_1\leq 1$, it follows from the definition of the Hölder class that f_0 has regularity $\beta_0\beta_1$, so that one expects a convergence rate of order $n^{-\frac{\beta_0\beta_1}{1+2\beta_0\beta_1}}$. It turns out that the actual (or 'effective') regularity depends on whether $\beta_i\leq 1$ or not. Let us define the following new 'regularity' parameter

$$\beta_i^* = \beta_i\prod_{\ell=i+1}^{q}(\beta_\ell\wedge 1). \tag{4.19}$$

Convergence result for compositions. Given d,t,β as before, let us define the rate

$$\varepsilon_n^* = \max_{0\leq i\leq q}\left\{ n^{-\frac{\beta_i^*}{2\beta_i^*+t_i}}\right\}. \tag{4.20}$$

Example. For $d_0 = d_1 = t_0 = t_1 = q = 1$ and $f = g_1 \circ g_0$ with $\beta_1, \beta_0 \leq 1$, we have $\beta_0^* = \beta_0(\beta_1 \wedge 1) = \beta_0\beta_1$ and $\beta_1^* = \beta_1$, and the rate ε_n^* equals, since $\beta_0\beta_1 \leq \beta_1$,

$$\max\left(n^{-\frac{\beta_1}{2\beta_1+1}}, n^{-\frac{\beta_0\beta_1}{2\beta_0\beta_1+1}}\right) = n^{-\frac{\beta_0\beta_1}{2\beta_0\beta_1+1}},$$

which gives the rate announced above for this example. One may check that the formula (4.20) also gives the expected rate in the other examples above.

Theorem 4.15 (Convergence of ReLU DNNs for compositions)**.** *Suppose $f_0 \in \mathcal{G}(q, d, t, \beta, K)$ for arbitrary $\beta > 0$ and $K > 0$, integer q and vector of integers d, t. Let $\hat{f} = \hat{f}^{ReLU}$ be the estimator in (4.5) with $\mathcal{F} = \mathcal{F}(L, N, s, F)$ the class of realisations of neural networks with depth L, width vector $N = (N_l)_{1\leq l\leq L}$, sparsity s and uniform bound F. Suppose $F \geq K \vee 1$ and a choice of parameters, for ε_n^* as in (4.20),*

$$\log n \leq L \leq n\varepsilon_n^{*2}, \qquad n\varepsilon_n^{*2} \leq \min_{1\leq l\leq L} N_l \leq \max_{1\leq l\leq L} N_l \leq n^2, \qquad s \asymp (\log n)\left(n\varepsilon_n^{*2}\right).$$

Then there exists $C = C(q, d, t, \beta, F)$ such that

$$\sup_{f_0 \in \mathcal{G}(q,d,t,\beta,K)} R(\hat{f}, f_0) \leq CL(\log n)^2 \varepsilon_n^{*2}.$$

In particular, if $L \asymp \log n$, the maximum risk is bounded by $C(\log n)^3 \varepsilon_n^{*2}$.

This result as a similar interpretation as Theorem 4.3: for well chosen parameters, the deep ReLU ERM achieves the rate ε_n^{*2} in prediction risk (up to a logarithmic factor). Moreover, this rate is optimal from the minimax perspective, as the next Theorem shows (under a mild condition on the dimensions). The remarkable point here is that, provided its parameters are well chosen, the deep ReLU estimator is able to *automatically* obtain the best possible rate, *without* being given any information beforehand on the underlying type of composition structure. Another setting somewhat different from compositional classes (but in the same spirit of a 'hidden structure') is that of data sitting on (or near) a geometric object, e.g. a manifold. One can show that deep ReLU ERM estimators again perform well in such settings, naturally 'adapting' to the unknown underlying geometric structure.

Theorem 4.16 (Minimax optimality for compositions)**.** *Consider the regression model (4.1), where the X_is are drawn from a distribution with density on $[0,1]^d$ which is bounded from above and below by positive constants. For arbitrary $\beta > 0$, integer q and vector of integers d, t, suppose $t_i \leq \min(d_0, \ldots, d_{i-1})$ for all i. Then for large enough K,*

$$\inf_{\hat{f}} \sup_{f_0 \in \mathcal{G}(q,d,t,\beta,K)} R(\hat{f}, f_0) \geq c\varepsilon_n^{*2},$$

where the infimum is taken over all possible estimators $\hat{f}$ of f in model (4.1).

Proof of Theorem 4.15. . The proof is based again on the two key ingredients viewed in Section 4.2.3. These, combined with the Lemma below, enable to obtain the result. The proof is then very similar to that of Theorem 4.3. We sketch the proof now.

Step 1 (shift–and–rescale). One rewrites the composition $f_0 = g_q \circ \cdots \circ g_0$ as

$$f_0 = h_q \circ \cdots \circ h_0,$$

where now the function $h_{0j} \in \mathcal{C}_{t_0}^{\beta_0}([0,1]^{t_0}, 1)$, for $1 \leq i \leq q-1$ we have $h_{ij} \in \mathcal{C}_{t_i}^{\beta_i}([0,1]^{t_i}, (2K)^{\beta_i})$ and $h_{qj} \in \mathcal{C}_{t_q}^{\beta_q}([0,1]^{t_q}, K(2K)^{\beta_q})$. This follows by shift–and–rescale by setting

$$h_0 = \frac{1}{2} + \frac{g_0}{2K}, \qquad h_i = \frac{1}{2} + \frac{g_i(2K\cdot - K)}{2K}, \qquad h_q = g_q(2K\cdot - K), \tag{4.21}$$