Proceedings of Machine Learning Research vol 195:1–72, 2023 36th Annual Conference on Learning Theory

SGD learning on neural networks: leap complexity and saddle-to-saddle dynamics

Emmanuel Abbe
École polytechnique Fédérale de Lausanne
Enric Boix-Adserà
Department of Electrical Engineering and Computer Science, MIT
Theodor Misiakiewicz
Department of Statistics, Stanford University

Editors: Gergely Neu and Lorenzo Rosasco

Abstract

We investigate the time complexity of SGD learning on fully-connected neural networks with isotropic data. We put forward a complexity measure, *the leap*, which measures how "hierarchical" target functions are. For d-dimensional uniform Boolean or isotropic Gaussian data, our main conjecture states that the time complexity to learn a function f with low-dimensional support is

$$\tilde{\Theta}(d^{\max(\mathrm{Leap}(f),2)}) \ .$$

We prove a version of this conjecture for a class of functions on Gaussian isotropic data and 2-layer neural networks, under additional technical assumptions on how SGD is run. We show that the training sequentially learns the function support with a saddle-to-saddle dynamic. Our result departs from Abbe et al. (2022b) by going beyond leap 1 (merged-staircase functions), and by going beyond the mean-field and gradient flow approximations that prohibit the full complexity control obtained here. Finally, we note that this gives an SGD complexity for the full training trajectory that matches that of Correlational Statistical Query (CSQ) lower-bounds.

1. Introduction

Deep learning has emerged as the standard approach to exploiting massive high-dimensional datasets. At the core of its success lies its capability to learn effective features with fairly blackbox architectures without suffering from the curse of dimensionality. To explain this success, two structural properties of data are commonly conjectured: (i) a *low-dimensional* structure that SGD-trained neural networks are able to adapt to; (ii) a *hierarchical* structure that neural networks can leverage with SGD training. In particular,

From a statistical viewpoint: A line of work (Bach, 2017; Schmidt-Hieber, 2020; Kohler and Krzyżak, 2016; Bauer and Kohler, 2019) has investigated the sample complexity of learning with deep neural networks, decoupled from computational considerations. By directly considering global solutions of empirical risk minimization (ERM) problems over arbitrarily large neural networks and sparsity inducing norms, they showed that deep neural networks can overcome the curse of dimensionality on classes of functions with low-dimensional and hierarchical structures. However, this approach does not provide efficient algorithms: instead, a number of works have shown computational hardness of ERM problems (Blum and Rivest, 1988; Klivans and Sherstov, 2009; Daniely et al., 2014) and it is unclear how much this line of work can inform practical neural networks, which are trained using SGD and variants.

From a computational viewpoint: A line of work in computational learning theory has provided time- and sample-efficient algorithms for learning Boolean functions with low-dimensional structure, based on their sparse Fourier spectrum (Mansour, 1994; O'Donnell, 2014). However, these algorithms are a priori quite different from SGD-trained neural networks. While unconstrained architectures can emulate any efficient learning algorithms (Abbe and Sandon, 2020; Abbe et al., 2021b), it is unclear whether more 'standard' neural networks can succeed on these same classes of functions or whether they require additional structure that pertains to hierarchical properties.

Thus, an outstanding question emerges from the current state of affairs:

For neural networks satisfying "regularity assumptions" (e.g., fully-connected, isotropically initialized layers), are there structural properties of the data that govern the time complexity of SGD learning? How does SGD exploit these properties in its training dynamics?

Here the key points are: (i) the "regularity" assumption, which prohibits the use of unorthodox neural networks that can emulate generic PAC/SQ learning algorithms as in Abbe and Sandon (2020); Abbe et al. (2021b); (ii) the requirement on the time complexity, which prohibits direct applications of infinite width, continuous time or infinite time analyses as in Chizat and Bach (2018, 2020). We discuss in Section 1.3 the various works that made progress towards the above, in particular regarding single- and multi-index models. We now specify the setting of this paper.

IID inputs and low-dimensional latent dimension. We focus on the following class of data distributions. First of all, we consider IID inputs, i.e.,

$$\boldsymbol{x} = (x_1, \dots, x_d) \sim \mu^{\otimes d}, \tag{1}$$

and we focus on the case where μ is either $\mathcal{N}(0,1)$ or $\mathrm{Unif}(\{+1,-1\})$, although we expect that other distributions would admit a similar treatment. Incidentally, the latter distribution is of interest in reasoning tasks related to Boolean arithmetic or logic (Saxton et al., 2019; Zhang et al., 2021; Abbe et al., 2022a). We now make a key assumption on the target function, that of having a *low latent dimension*, i.e., $f(\boldsymbol{x}) = h(\boldsymbol{z})$ where $\boldsymbol{z} = \boldsymbol{M}\boldsymbol{x}$ and

$$\begin{aligned} &\text{(Gaussian case)} \quad \boldsymbol{M} \text{ a } P \times d \text{ dimensional, real-valued matrix such that } \boldsymbol{M}\boldsymbol{M}^{\top} = \boldsymbol{I}_P \\ &\text{(Boolean case)} \quad \boldsymbol{M} \text{ a } P \times d \text{ dimensional, } \{0,1\}\text{-valued matrix such that } \boldsymbol{M}\boldsymbol{M}^{\top} = \boldsymbol{I}_P \end{aligned} \tag{2}$$

with the assumption that $P = O_d(1)$. In other words, the target function has a large ambient dimension but depends only on a finite number of latent coordinates. In the Gaussian case the coordinates are not known because of a possible rotation of the input, and in the Boolean case the coordinates are not known because of a possible permutation of the input.

Data with large ambient dimension but low latent dimension have long been a center of focus in machine learning and data science. It is known that kernel methods cannot exploit low latent dimension, i.e., it was proved in Hsu et al. (2021); Hsu; Kamath et al. (2020); Abbe et al. (2022b) that any kernel method needs a number of features p or samples n satisfying

$$\min(n, p) \geq \Omega(d^D) \tag{3}$$

in order to learn a Boolean function as above with degree $D = O_d(1)$. In other words, for kernel methods D controls the sample complexity irrespective of any potential additional structural properties of f (e.g., hierarchical properties). On the other hand, it is known that this is not the limit for deep learning, which can break the d^D curse, as discussed next.

The example of staircases. Consider the following example: $\boldsymbol{x} \sim \mathrm{Unif}(\{+1,-1\}^d)$ is drawn from the hypercube and the target function is 4-sparse, either

$$h_{,1}(\boldsymbol{z}) = z_1 + z_1z_2 + z_1z_2z_3 + z_1z_2z_3z_4\,, \qquad \text{or} \qquad h_{,2}(\boldsymbol{z}) = z_1z_2z_3z_4\,.$$

The first function is called a vanilla staircase of degree 4 (Abbe et al., 2021a, 2022b). The second is a monomial of degree 4. Each of these functions induces a function class under the permutation of the variables (i.e., one can consider the class of all monomials on any 4 of the d input variables, and similarly for staircases). One can verify that these function classes have similar approximation and statistical complexity because of the low-dimensional structure, but have different computational complexity because of the hierarchical structure. For example, under the Correlational Statistical Query (CSQ) model of computation (Ben-David et al., 1995; Kearns, 1998; Bshouty and Feldman, 2002), the first class has CSQ dimension $\Theta(d)$ versus $\Theta(d^4)$ for the second class[1].

Consider now learning these two functions with online-SGD[2] on a two-layer neural network $\hat{f}_{\mathsf{NN}}(\boldsymbol{x};\boldsymbol{\Theta}) = \sum_{j\in[M]} a_j\sigma(\langle \boldsymbol{w}_j, \boldsymbol{x}\rangle + b_j)$. For online-SGD in the mean-field scaling, it was shown in Abbe et al. (2022b) that $h_{,1}$ can be learned in $\Theta_d(d)$ steps, while $h_{,2}$ cannot be learned in $O_d(d)$ steps, but it was not shown in which complexity $h_{,2}$ could be learned. How can we understand this? At initialization, the gradient of the neural network has correlation with each monomial of order $O(d^{(k-1)/2})$ inside the support and $O(d^{(k+1)/2})$ outside the support (for a degree k-monomial). In the first case, the gradient has $O(1)$ correlation with the first monomial z_1 and can learn the first coordinate, then the second coordinate becomes easier to learn using the second monomial, and so on and so forth. In the second case, the correlation is of order $d^{-3/2}$ on the support and SGD needs to align to all 4 coordinates at once, which takes more time. Indeed, we will see that to align with k coordinates at once, we need $\tilde{O}(d^{k-1})$ steps, which matches (up to logarithmic factors) the computational lower bound of CSQ algorithms.

In this paper, we treat these functions in a unified manner, with the "leap" complexity measure, where $h_{,1}$ and $h_{,2}$ are leap-1 and leap-4 functions respectively. Leap-k functions will be learned in $\tilde{\Theta}(d^{\max(k-1,1)})$ online-SGD steps. Note that going from staircase functions having leap-1 to more general functions of arbitrary (finite) leaps is highly non-trivial. This is because the mean-field gradient flow used in Abbe et al. (2022b) cannot be used beyond the scaling of $O(d)$ steps, as required for $k > 1$, because the mean-field PDE approximation breaks down (Mei et al., 2019).

1.1. The leap complexity

We now define the leap complexity. Any function in $L^2(\mu^{\otimes P})$ can be expressed in the orthogonal basis of $L^2(\mu^{\otimes P})$, i.e., the Hermite or Fourier-Walsh basis for $\mu \sim \mathsf{N}(0,1)$ and $\mu \sim \mathrm{Unif}(\{+1,-1\})$ respectively,

$$h(\boldsymbol{z}) = \sum_{S\subseteq \mathcal{Z}^P} \hat{h}(S)\chi_S(\boldsymbol{z}), \tag{4}$$

where $\mathcal{Z} = \{0,1\}$ for the Boolean case and $\mathcal{Z} = \mathsf{Z}_+$ for the Gaussian case, $\chi_S(\boldsymbol{z}) = \prod_{i\in[P]} \chi_{S_i}(z_i)$,

$$\chi_{S_i}(z_i) = \begin{cases} z_i^{S_i} & \text{(Boolean case)} \\ \mathrm{He}_{S_i}(z_i) & \text{(Gaussian case)} \end{cases} \tag{5}$$

1. See Section 2 for more details on CSQ.
2. Online-SGD means that on each SGD iteration a fresh sample $(\boldsymbol{x}^t, y^t)$ is used.

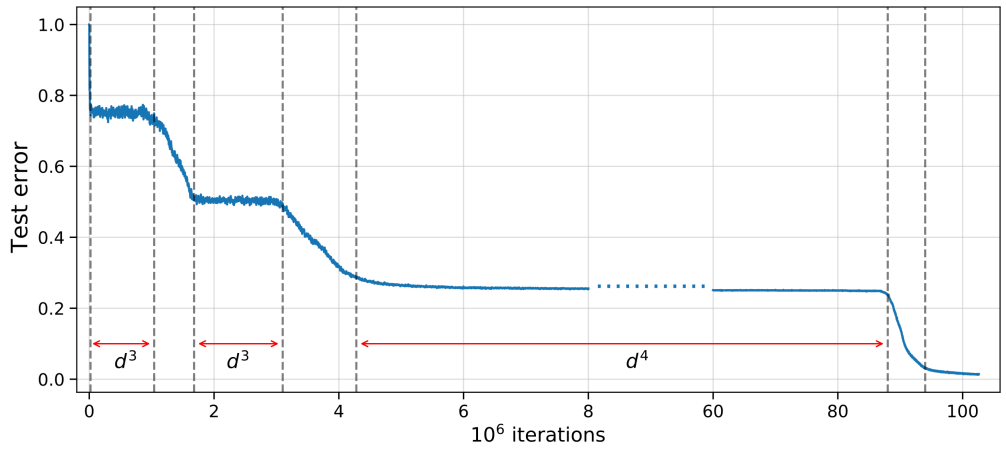


Figure 1: Test error versus the number of online-SGD steps to learn $h(\boldsymbol{z}) = z_1 + z_1z_2\cdots z_5 + z_1z_2\cdots z_9 + z_1z_2\cdots z_{14}$ in ambient dimension $d = 100$ on the hypercube. We take $M = 300$ neurons with shifted sigmoid activation and train both layers at once with constant step size $0.4/d$. The SGD dynamics follows a saddle-to-saddle dynamic and sequentially picks up the support and monomials z_1 in roughly d steps, $z_1z_2\cdots z_5$ in d^3 steps (leap of size 4), $z_1z_2\cdots z_9$ in d^3 steps (leap of size 4) and $z_1z_2\cdots z_{14}$ in d^4 steps (leap of size 5).

where He_k is the k-th Hermite polynomial, $k \in \mathsf{Z}_+$. The leap is given as follows.

Definition 1 (Leap complexity) *Let h be as before with non-zero basis elements given by the subset $\mathcal{S}(h) := \{S_1, \dots, S_m\}$, $m \in \mathsf{Z}_+$. We define the leap complexity of h as*[3]

$$\mathrm{Leap}(h) := \min_{\pi 2 \Pi_m} \max_{i 2 [m]} \left\| S_{\pi(i)} \setminus \cup_{j=0}^{i} {}^{1} S_{\pi(j)} \right\|_1 ,$$

where, for $S_j = (S_j(1), \dots, S_j(P))$ in $\{0,1\}^P$ or Z_+^P for the Boolean or Gaussian case respectively, $\|S_{\pi(i)} \setminus \cup_{j=0}^{i} {}^{1} S_{\pi(j)}\|_1 := \sum_{k 2 [P]} S_{\pi(i)}(k) \mathbb{1}\{S_{\pi(j)}(k) = 0, \forall j \in [i-1]\}$, with $S_{\pi(0)} = 0^P$. We then say that h is a leap-$\mathrm{Leap}(h)$ function.

In words, a function h is leap-k if its non-zero monomials can be ordered in a sequence such that each time a monomial is added, the support of h grows by at most k new coordinates, where each new coordinate is counted with multiplicity in the Gaussian case (and the 1-norm collapses to the cardinality of the difference set in the Boolean case). Note that the definition of leap-k functions on the hypercube generalizes the definition of functions with the merged-staircase property (leap-1 functions) from Abbe et al. (2022b).

Some examples in the Boolean case,

$$\mathrm{Leap}(z_1 + z_1z_2 + z_1z_2z_3 + z_1z_2z_3z_4) = 1 , \qquad \mathrm{Leap}(z_1 + z_2 + z_2z_3z_4) = 2 ,$$
$$\mathrm{Leap}(z_1 + z_1z_2z_3 + z_2z_3z_4z_5z_6z_7) = 4 , \qquad \mathrm{Leap}(z_1z_2z_3 + z_2z_3z_4) = 3 ,$$

3. Π_m is the symmetric group of permutations on $[m]$.

and on isotropic Gaussian data,

$$\mathrm{Leap}(\mathrm{He}_k(z_1)) = \mathrm{Leap}(\mathrm{He}_1(z_1)\mathrm{He}_1(z_2)\cdots\mathrm{He}_1(z_k)) = k\,,$$
$$\mathrm{Leap}(\mathrm{He}_{k_1}(z_1) + \mathrm{He}_{k_1}(z_1)\mathrm{He}_{k_2}(z_2) + \mathrm{He}_{k_1}(z_1)\mathrm{He}_{k_2}(z_2)\mathrm{He}_{k_3}(z_3)) = \max(k_1,k_2,k_3)\,,$$
$$\mathrm{Leap}(\mathrm{He}_2(z_1) + \mathrm{He}_2(z_2) + \mathrm{He}_2(z_3) + \mathrm{He}_3(z_1)\mathrm{He}_8(z_3)) = 2\,.$$

1.2. Summary of our contributions

Overview. This paper puts forward a general conjecture characterizing the time complexity of SGD-learning on regular neural networks with isotropic data of low latent dimension. The key quantity that emerges to govern the complexity is the *leap* (Definition 1). This gives a formal measure of "hierarchy" in target functions, going beyond spectrum sparsity and emerging from the study of SGD-trained regular networks. The paper then proves a specialization of the conjecture to a representative class of functions on Gaussian inputs, but for 2-layer neural networks and with certain technical assumptions on how SGD is run. The two main innovations of the proof are (i) a full control of the time complexity of SGD learning on a fully-connected network (without infinite width or continuous time approximations); (ii) going beyond one-step gradient analyses and showing that the leap controls the entire learning trajectory due to a sequential learning mechanism (saddle-to-saddle). We also provide experimental evidence towards the more general conjecture with vanilla SGD and derive CSQ lower-bounds for noisy GD that match our achievability bounds.

Conjecture 2 *Let $f : \mathsf{R}^d \to \mathsf{R}$ in $L_2(\mu^{\otimes d})$ for μ either $\mathcal{N}(0,1)$ or $\mathrm{Unif}\{+1,-1\}$ satisfying the low-latent-dimension hypothesis $f(\boldsymbol{x}) = h(\boldsymbol{M}\boldsymbol{x})$ in (2) for some $P = O_d(1)$. Let $\hat{f}^t_{\mathsf{NN}}$ be the output of training a fully-connected neural network with $\mathrm{poly}(d)$-edges and rotationally-invariant weight initialization with t steps of online-SGD on the square loss. Then, for all but a measure-0 set of functions (see below), the risk is bounded by*

$$R(\hat{f}^t_{\mathsf{NN}}) := \mathsf{E}_{\boldsymbol{x}}\left[\left(\hat{f}^t_{\mathsf{NN}}(\boldsymbol{x}) - f(\boldsymbol{x})\right)^2\right] \le \varepsilon \quad \textit{if and only if} \quad t = \tilde{\Omega}_d(d^{\mathrm{Leap}(h)-1})\mathrm{poly}(1/\varepsilon)\,.$$

So the total time complexity[4] *is $\tilde{\Omega}_d(d^{\mathrm{Leap}(h)-2})\mathrm{poly}(1/\varepsilon)$ for bounded width/depth networks*[5].

The "measure-0" statement in the conjecture means the following. For any set $\{S_1,\dots,S_m\}$ of nonzero (Fourier or Hermite) basis elements, the conjecture holds for all h with $\mathcal{S}(h) = \{S_1,\dots,S_m\}$ in the decomposition (4), except for a set of coefficients $\{(\hat{h}(S_i))_{i\in[m]}\} \subset \mathsf{R}^m$ of Lebesgue-measure 0. This part of the conjecture is needed for Boolean functions, since it was proved in Abbe et al. (2022b) that a measure-0 set of "degenerate" leap-1 functions on the hypercube are not learned in $\Theta(d)$ SGD-steps by 2-layer neural networks in the mean-field scaling. However, we further conjecture that in the case of Gaussian data the measure-0 modification can be removed if we instead use a rotationally invariant version of the leap. See discussion in Appendix B.2.

Remark 3 *We believe that the conjecture (in particular the time complexity scaling) holds for more general architectures than those with isotropically-initialized layers, as long as enough 'regularity' assumptions are present at initialization (prohibiting the type of 'emulation networks' used in Abbe and Sandon (2020)). Note that it is not enough to ask for only the first layer to be initialized with*

4. The total time complexity is given by computing for each neuron and SGD step, a gradient in d dimensions.
5. The polylogs in $\tilde{\Omega}$ might not be necessary in the special case of $\mathrm{Leap} = 1$ as indicated by Abbe et al. (2022b).

a rotationally-invariant distribution, as this may be handled by using emulation networks on the subsequent layers, but weaker invariances of subsequent layers (e.g., permutation subgroups) may suffice.

Formal results. In order to prove a formal result as close as possible to the general conjecture, we rely on the following specifications: (1) 2-layer NN with smooth activations, (2) layer-wise SGD, (3) projected gradient steps, and (4) a representative subclass of functions on Gaussian isotropic data. We refer to Section 3 for the details of the formal results. We also provide in Section 2 lower-bounds for kernels and CSQ algorithms on regular neural networks. In particular, our results show that SGD-training on fully-connected neural networks achieves the optimal $d^{\Theta(\mathrm{Leap}(h))}$ computational complexity of the best CSQ algorithm on this class of sparse functions, going beyond kernels.

The characterization obtained in this paper implies a relatively simple picture for learning low-dimensional functions with SGD on neural networks:

Picking up the support. SGD sequentially learns the target function with a saddle-to-saddle dynamic. Specifically, in the first phase, the network learns the support that is reachable by the monomial(s) of lowest degree, and fits these monomials to produce a first descent in the loss. Then, iteratively, each time a new set of coordinates is added to the support, with cardinality bounded by the leap L, SGD takes at most $\tilde{\Theta}(d^{\max(L,1)})$ steps to identify the new coordinates, before escaping the saddle with another loss descent that fits the new monomials. Thus the dynamic moves from saddle points to saddle points, with plateaus of length corresponding to the leap associated with each saddle. See Figure 1 for an illustration.[6]

Computational time. Since the total training time is dominated by the time to escape the saddle with the largest leap, our results imply a $\tilde{\Theta}(d^{\mathrm{Leap}(h)-2})$ time complexity. This scaling matches the CSQ lower-bounds from Section 2, which are also exponential in the leap. Thus SGD on regular neural networks and low latent dimensional data is shown to achieve a time scaling to learn that matches that of optimal CSQ algorithms; see Section 2 for further discussions.

Curriculum learning. SGD on regular neural networks implicitly implements a form of 'adaptive curriculum' learning. SGD first picks up low-level features that are computationally and statistically easier to learn, and by picking up these low level features, it makes the learning of higher-level features in turn easier. As mentioned in the examples of Table 1: learning $z_1 \cdots z_{2k}$ takes $\widetilde{\Theta}(d^{2k-1})$ sample complexity (leap-$2k$ function). But if we add an intermediary monomial to our target to create $z_1 \cdots z_k + z_1 \cdots z_{2k}$, then it takes $\widetilde{\Theta}(d^{k-1})$ steps to learn (leap-k function). If we have a full staircase, it only requires $\Theta(d)$ (leap-1 function). This thus gives an adaptive learning process that follows a curriculum learning procedure where features of increasing complexity guide the learning.

Finally we note that we considered here the setting of online-SGD, and a natural question is to consider how the picture may change under ERM (several passes with the same batch of samples). The ERM setting is however harder to analyze. We consider this to be an important direction for future works. Note that our results imply a sample complexity equal to the number of SGD steps

6. As we will discuss, we only show a saddle-to-saddle behavior when the leaps are of increasing size as the dynamics progress, so it is theoretically open whether it holds in the more general setting where leaps can decrease.

$h\ (\boldsymbol{z}) =$	$z_1 z_2 \cdots z_{2k}$	$z_1 z_2 \cdots z_k + z_1 z_2 \cdots z_{2k}$	$z_1 + z_1 z_2 + \ldots + z_1 z_2 \cdots z_{2k}$
Kernels	$\Omega(d^{2k})$	$\Omega(d^{2k})$	$\Omega(d^{2k})$
SGD on NN	$\tilde{\Theta}(d^{2k\ \ 1})$	$\tilde{\Theta}(d^{k\ \ 1})$	$\Theta(d)$

Table 1: Sample size n to fit $f\ (\boldsymbol{x}) = h\ (\boldsymbol{M}\boldsymbol{x})$. SGD-trained neural networks implicitly implement an 'adaptive' or 'curriculum' learning scheme, by exploiting lower degree monomials to efficiently learn higher degree monomials.

$n = t = \widetilde{\Theta}(d^{\max(\text{Leap}\ \ 1,1)})$. In ERM, we reuse samples and consequently reduce the sample complexity. We conjecture in fact that $n = \widetilde{\Theta}(d^{\max(\text{Leap}/2,1)})$ is optimal for ERM. Furthermore, this paper considers the case of low-dimensional functions $P = O_d(1)$, which allows to focus on the dependency on d in the time-complexity of SGD. A natural future direction is to extend these results to larger P. See Appendix B.4 for further discussion.

1.3. Related works

A string of works (Allen-Zhu and Li, 2019; Li et al., 2020; Daniely and Malach, 2020; Allen-Zhu and Li, 2020; Suzuki and Akiyama, 2020; Ba et al., 2022; Ghorbani et al., 2019; Telgarsky, 2022) has explored the power of learning with neural networks beyond neural tangent kernels (Jacot et al., 2018). In particular, much attention has been devoted to learning multi-index models (Chen et al., 2020; Abbe et al., 2022b; Nichani et al., 2022; Barak et al., 2022; Damian et al., 2022; Mousavi-Hosseini et al., 2022; Bietti et al., 2022; Refinetti et al., 2021), i.e., functions that only depend on a small number of (unknown) relevant directions $\mathsf{E}[y|\boldsymbol{x}] = h\ (\langle \boldsymbol{u}_1, \boldsymbol{x}\rangle, \ldots, \langle \boldsymbol{u}_P, \boldsymbol{x}\rangle)$. These functions offer a simple setting where we expect to see a large benefit of non-linear 'feature learning' (aligning the weights of the neural networks with the sparse support), compared to fixed-feature methods (kernel methods). The conjectural picture described in our paper offers a unified framework to understand learning multi-index functions with SGD-trained regular neural networks on square loss. For example, Mousavi-Hosseini et al. (2022) considers learning monotone single-index functions, which is a special case of learning leap-1 functions, and shows that they can be learned in $\tilde{\Theta}(d)$ online-SGD steps. Damian et al. (2022) considers learning a low-rank polynomial on Gaussian data, with null Hermite-1 coefficients and full rank Hessian, which implies that the polynomial is a leap-2 function. They show that it can be learned in $n = \Theta(d^2)$ samples with one-gradient descent step on the first layer weights, while we conjecture (and show for a subset of those polynomials) that $\tilde{\Theta}(d)$ online-SGD steps is sufficient. Barak et al. (2022) considers learning degree-k monomials on the hypercube and shows that $n = d^{O(k)}$ samples are sufficient, using one gradient descent step on the first layer, while we conjecture (and prove in the Gaussian case) a tighter scaling of $\tilde{\Theta}(d^{k\ \ 1})$ online-SGD steps[7]. Bietti et al. (2022) considers a single-index leap-k function on Gaussian data and obtains the tight scaling $\tilde{\Theta}(d^{k\ \ 1})$ with a neural network where all first layer weights are equal. An important innovation of our work compared with these previous results is that we show a *sequential learning mechanism* with several learning phases, which prevents the use of

7. Note that a kernel method can learn degree-k monomials with $n = \Theta(d^k)$ samples, and this tighter analysis is necessary to obtain a separation here.

single-index models (Bietti et al., 2022) or one gradient-descent step analysis (Daniely and Malach, 2020; Barak et al., 2022; Damian et al., 2022).

In parallel, several works have studied the dynamics of SGD in simpler non-convex models in high dimensions (Ge et al., 2015; Tan and Vershynin, 2019; Chen et al., 2019; Arous et al., 2021). Our analysis relies on a similar drift plus martingale decomposition of online-SGD as in Tan and Vershynin (2019); Arous et al. (2021). In particular, the leap complexity is related to the *information-exponent* introduced in Arous et al. (2021). The latter considers a single-index model trained with online-SGD on a non-convex loss and the information exponent captures the scaling of the correlation between the model at a typical initialization and the global solution. Arous et al. (2021) showed that, with information exponent k, online-SGD requires $\widetilde{\Theta}(d^{k-1})$ steps to converge, similarly to the scaling presented in this paper. However our analysis and the definition of the leap complexity differ from Arous et al. (2021) in two major ways. First, our model is not a single parameter model, so a much more involved analysis is required for the dynamics. Second, the information exponent is a coefficient that only applies at initialization, while the leap-complexity is a measure of targets that controls the entire learning trajectory (our neural networks visit several saddles during training).

See Appendix B.1 for further references.

2. Lower bounds on learning leap functions

Linear methods such as kernel methods suffer exponentially in the degree of the target function, and cannot use the "hierarchical" structure to learn faster. This was proved in Abbe et al. (2022b) for the Boolean case, and this work extends the result to the Gaussian case:

Proposition 4 (Lower bound for linear methods; informal statement of Propositions 27 and 28) *Let h be a degree-D polynomial over the Boolean hypercube (resp., Gaussian measure). Then there are $c_h, \varepsilon_h > 0$, such that any linear method needs $c_h\, d^D$ samples to learn $f(\boldsymbol{x}) = h(\boldsymbol{M}\boldsymbol{x})$ to less than $\varepsilon_h > 0$ error, where $\boldsymbol{M}$ is an unknown permutation (resp., rotation) as in (2).*

Consider now the Correlational Statistical Query (CSQ) model of computation (Ben-David et al., 1995; Bshouty and Feldman, 2002). A CSQ algorithm accesses the data via expectation queries, plus additive noise. We show that for CSQ methods the query complexity scales exponentially in the *leap* of the target function, which can be much less than the degree.

Proposition 5 (Lower bound for CSQ methods; informal statement of Propositions 30 and 31) *In the setting of Proposition 4, the CSQ complexity of learning f to less than ε_h error is at least $c_h\, d^{\mathrm{Leap}(h)}$ in the Boolean case, and at least $c_h\, d^{\mathrm{Leap}(h)/2}$ in the Gaussian case.*

The scaling $d^{\mathrm{Leap}(h)}$ for Boolean functions in Proposition 5 matches the total time complexity scaling in Conjecture 2. For Gaussian functions, we only prove $d^{\mathrm{Leap}(h)/2}$ scaling. However we conjecture that the same scaling as the Boolean case should hold. See Appendix F for details.

Remark 6 *We note that the above lower-bounds are for CSQ models or noisy population-GD models, and not for online-SGD since the latter takes a single sample per time step. Our proof does show a correspondence between online-SGD and population-GD, but without the additional noise. It is however intriguing that the regularity in the network model for online-SGD appears to act*

comparatively in terms of constraints to a noisy population-GD model (on possibly non-regular architectures), and we leave potential investigations of such correspondences to future work (see also discussion in Appendix B.5). Further, we note that the correspondence to CSQ may not hold beyond the finite P regime. First there is the 'extremal case' of learning the full parity function, which is efficiently learnable in CSQ (with 0 queries) but not necessarily with online-SGD on regular networks: Abbe and Boix-Adsera (2022) shows it is efficiently learnable by a i.i.d. Rademacher$(1/2)$ initialization, but not necessarily by a Gaussian isotropic initialization. Further, the positive result of the Rademacher initialization may disappear under proper hyperparameter 'stability' assumptions. Beyond this extremal case, a more important nuance arises for large P: the fitting of the function on the support may become costly for regular neural networks in certain cases. For example, let $g : [P] \to \{0, 1\}$ be a known function and consider learning $f(\boldsymbol{x})$ which depends on P unknown coordinates as $h(\boldsymbol{z}) = \sum_{i=1}^{P} iz_i + \prod_{i=1}^{P} z_i^{g(i)}$. This is a leap-1 function where the linear part reveals the support and the permutation, and with a parity term on the indices such that $g(i) = 1$. In this case, SGD on a regular network would first pick up the support, and then have to express a potentially large degree monomial on that support, which may be hard if P is large (i.e., $P \gg 1$). The latter part may be non trivial for SGD on a regular network, while, since g is known, it would require 0 queries for a CSQ algorithm once the permutation was determined from learning the linear coefficients.

3. Learning leap functions with SGD on neural networks

Let $h : \mathsf{R}^P \to \mathsf{R}$ be a degree-D polynomial. We consider learning $f(\boldsymbol{x}) = h(\boldsymbol{z})$ on isotropic Gaussian data, where $\boldsymbol{z} = \boldsymbol{M}\boldsymbol{x}$ is the covariate projected on a P-dimensional latent subspace, using online-SGD on a two-layer neural network. At each step t, we get a new fresh (independent) sample $(\boldsymbol{x}^t, y^t)$ where $\boldsymbol{x}^t \sim \mathsf{N}(0, \mathbf{I}_d)$ and $y^t = h(\boldsymbol{z}^t) + \varepsilon^t$, with additive noise ε^t independent and K-sub-Gaussian. For the purpose of our analysis, we assume that $\boldsymbol{z}$ is a subset of P coordinates of $\boldsymbol{x}$ instead of a general subspace. This limitation of our analysis is because of an entrywise projection step that we perform during training for technical reasons, and which makes the training algorithm non-rotationally equivariant (see description of the algorithm below). Since we assume that $\boldsymbol{z}$ is a subset of the coordinates, without loss of generality we choose $\boldsymbol{z}$ to be the first P coordinates of $\boldsymbol{x}$.

3.1. Algorithm

We use a 2-layer neural network with M neurons and weights $\boldsymbol{\Theta} = (a_j, b_j, \boldsymbol{w}_j)_{j \in [M]} \in \mathsf{R}^{M(d+2)}$:

$$\hat{f}_{\mathsf{NN}}(\boldsymbol{x}; \boldsymbol{\Theta}) = \sum_{j \in [M]} a_j \sigma(\langle \boldsymbol{w}_j, \boldsymbol{x} \rangle + b_j)\,, \tag{6}$$

We consider the following assumption on the activation function:[8]

Assumption 7 *Let $\sigma : \mathsf{R} \to \mathsf{R}$ be an activation function that satisfies the following conditions. There exists a constant $K > 0$ such that σ is $(D+3)$-differentiable with $\|\sigma^{(k)}\|_1 \le K$ for $k = 0, \dots, (D+3)$ and $|\mu_k(\sigma)| > 1/K$ for $k = 0, \dots, D$, where $\mu_k(\sigma) = \mathsf{E}_G[\mathrm{He}_k(G)\sigma(G)] = \mathsf{E}_G[\sigma^{(k)}(G)]$ is the k-th Hermite coefficient of σ and $G \sim \mathsf{N}(0, 1)$.*

8. This is satisfied, for example, by the shifted sigmoid $\sigma(z) = 1/(1 + e^{z+c})$ for almost all shifts c.

We train $\hat{f}_{\mathsf{NN}}$ using online-SGD on the squared loss $\ell(y,\hat{y}) = \frac{1}{2}(y-\hat{y})^2$, with the goal of minimizing the population risk:

$$R(\boldsymbol{\Theta}) = \mathsf{E}_{\boldsymbol{x}}\Big[\ell\big(f\ (\boldsymbol{x}), \hat{f}_{\mathsf{NN}}(\boldsymbol{x};\boldsymbol{\Theta})\big)\Big]\,. \tag{7}$$

For the purposes of the analysis, we make two modifications to SGD training. First, we train layerwise: training $\{\boldsymbol{w}_j\}_{j2[M]}$ and then $\{a_j\}_{j2[M]}$, while keeping the biases $\{b_j\}_{j2[M]}$ frozen during the whole training. Second, during the training of the first layer weights $\{\boldsymbol{w}_j\}_{j2[M]}$, we project the weights in order to ensure that they remain bounded in magnitude. See Algorithm 1 for pseudocode, and see below for a detailed explanation. These modifications are not needed in practice for SGD to learn, as we demonstrate in our experiments in Figure 1 and Appendix A.

Algorithm 1: Layerwise online-SGD with init scales $\kappa, \rho > 0$, learning rates $\eta_1, \eta_2 > 0$, step counts $\overline{T}_1, \overline{T}_2 > 0$, second-layer ridge-regularization $\lambda_a > 0$, and projection params $r, \Delta > 0$

$a_j^0 \sim \mathrm{Unif}(\{\pm\kappa\})$, $b_j^0 \sim \mathrm{Unif}([-\rho,+\rho])$, $\boldsymbol{w}_j^0 \sim \mathrm{Unif}(\{\pm 1/\sqrt{d}\}^d)$ `// Initialization`

for $t = 0$ ***to*** $\overline{T}_1 - 1$, ***and all*** $j \in [M]$ **do** `// Train first layer with projected SGD`

- $\widetilde{\boldsymbol{w}}_j^{t+1} = \boldsymbol{w}_j^t - \eta_1 \cdot \mathrm{grad}_{\boldsymbol{w}_j^t}\ell\big(y^t, \hat{f}_{\mathsf{NN}}(\boldsymbol{x}^t;\boldsymbol{\Theta}^t)\big)$, where grad is spherical gradient in (9)
- $\boldsymbol{w}_j^{t+1} =$ projection of $\widetilde{\boldsymbol{w}}_j^{t+1}$ defined in (10)
- $a_j^{t+1} = a_j^t$, $b_j^{t+1} = b_j^t$

for $t = \overline{T}_1$ ***to*** $\overline{T}_1 + \overline{T}_2 - 1$, ***and all*** $j \in [M]$ **do** `// Train second layer with SGD`

- $\boldsymbol{w}_j^{t+1} = \boldsymbol{w}_j^t$, $b_j^{t+1} = b_j^t$
- $a_j^{t+1} = (1-\lambda_a)a_j^t - \eta_2 \cdot \frac{\partial}{\partial a_j^t}\ell\big(y^t, \hat{f}_{\mathsf{NN}}(\boldsymbol{x}^t;\boldsymbol{\Theta}^t)\big)$

Analyzing layerwise training is a fairly standard tool in the theoretical literature to obtain rigorous analyses; it is used in a number of works, including Daniely and Malach (2020); Barak et al. (2022); Damian et al. (2022); Abbe et al. (2022b). In our setting, layerwise training allows us to analyze the complicated dynamics of neural network training, but it also leads to a major issue. During the training of the first layer, the target function f is not fully fitted because we do not train the second layer concurrently. Therefore the first-layer weights continue to evolve even after they pick up the support of f. This is a challenge since we must train the first-layer weights for a large number of steps, and so they can potentially grow to a very large magnitude, leading to instability.[9]

We correct the issue by projecting each neuron's first-layer weights $\boldsymbol{w}_j$ to ensure that the coordinates do not blow up. First, we keep the "small" coordinates of $\boldsymbol{w}_j$ on the unit sphere, i.e., for some parameter $r > 0$, we define the "small" coordinates for neuron j at time t by $S_{j,0} = [d]$ and

$$S_{j,t} = \{i \in [d] : |\tilde{w}_{j,i}^{t^0}| < r \text{ for all } 1 \le t^0 \le t\}.$$

We project these coordinates on the unit sphere using the operator P_j^t defined by

$$\big(\mathsf{P}_j^t \boldsymbol{w}_j^t\big)_i = w_{j,i}^t \quad \text{if } i \notin S_{j,t}; \qquad \big(\mathsf{P}_j^t \boldsymbol{w}_j^t\big)_i = \frac{w_{j,i}^t}{\|\mathcal{S}_t(\boldsymbol{w}_j^t)\|_2} \quad \text{if } i \in S_{j,t}, \tag{8}$$

9. We emphasize that this problem is due to layerwise training, since in practice if we train both layers at the same time the residual quickly goes to zero after the support is picked up, and so the first-layer weights stop evolving and remain bounded in magnitude (see Appendix A).

and use the spherical gradient with respect to the sphere $\|\mathcal{S}_t(\boldsymbol{w}_j^t)\|_2 = 1$, i.e., for any function f,

$$\text{grad}_{\boldsymbol{w}_j^t} f(\boldsymbol{w}_j^t) = \nabla_{\boldsymbol{w}_j^t} f(\boldsymbol{w}_j^t) - \mathcal{S}_t(\boldsymbol{w}_j^t)\langle \mathcal{S}_t(\boldsymbol{w}_j^t), \nabla_{\boldsymbol{w}_j^t} f(\boldsymbol{w}_j^t)\rangle \,. \tag{9}$$

In order to keep the "large" coordinates $i \notin S_{j,t}$ from growing too large, we project them onto the ℓ_1 ball of radius Δ for some $\Delta > r$, and denote this projection by P_1. In summary, the projection performed in the training of the first layer can be written compactly as

$$\boldsymbol{w}_j^{t+1} = \mathsf{P}_j^{t+1}\mathsf{P}_1 \widetilde{\boldsymbol{w}}_j^{t+1} \,. \tag{10}$$

In the second phase, the training of the second layer weights $\boldsymbol{a}$ is by standard SGD (without projection) with added ridge-regularization term $\frac{\lambda_a}{2}\|\boldsymbol{a}\|^2$ to encourage low-norm solutions.

3.2. Learning a single monomial

We first consider the case of learning a single monomial with Hermite exponents $k_1, \dots, k_P \geq 1$:

$$h(\boldsymbol{z}) = \text{He}_{k_1}(z_1)\text{He}_{k_2}(z_2)\cdots\text{He}_{k_P}(z_P) \,.$$

We assume $D = k_1 + \ldots + k_P \geq 2$ (the case $D = 1$ is straightforward). h is a leap-D function. We start by proving that, during the first phase, the first layer weights grow in the directions of $z_1, \dots, z_P$ which are the variables in the support of the target function.

Theorem 8 (First layer training, single monomial, sum of monomials) *Assume σ satisfy Assumption 7. Then for $0 < r < \Delta$ sufficiently small (depending on D, K) and $\rho \leq \Delta$ the following holds. For any constant $C > 0$, there exist C_i for $i = 0, \dots, 6$, that only depend on D, K and C such that*

$$\overline{T}_1 = C_0 d^{D-1}\log(d)^{C_0} \,, \qquad \eta_1 = \frac{1}{C_1\kappa d^{D/2}\log(d)^{C_1}} \,, \qquad \kappa \leq \frac{1}{C_2 d^{C_2}} \,,$$

and for d large enough that $r \geq C_0\log(d)^{C_0}/\sqrt{d}$, the following event holds with probability at least $1 - Md^{-C}$. For any neuron $j \in [M]$,

(a) Early stopping: $|w_{j,i}^t - w_{j,i}^0| \leq C_3/\sqrt{d\log(d)}$ for all $i \in [d]$ and $t \leq \overline{T}_1/(C_4\log(d)^{C_4})$.

And for any neuron $j \in [M]$ such that $a_j^0\mu_D(\sigma)(w_{j,1}^0)^{k_1}\cdots(w_{j,P}^0)^{k_P} > 0$,

(b) On the support: $\left|w_{j,i}^{\overline{T}_1} - \text{sign}(w_{j,i}^0)\cdot\Delta\right| \leq C_5/\sqrt{d\log(d)}$ for $i = 1, \dots, P$.

(c) Outside the support: $|w_{j,i}^{\overline{T}_1} - w_{j,i}^0| \leq C_6 r^2/\sqrt{d}$ for $i = P+1, \dots, d$, and $\sum_{i>P}(w_{j,i}^{\overline{T}_1})^2 = 1$.

Theorem 8 shows that after the end of the first phase, the coordinates $\boldsymbol{w}_j^{\overline{T}_1}$ aligned with the support $\boldsymbol{z}$ are all close to $\pm\Delta$ with the same signs as $w_{j,1}^0, \dots, w_{j,P}^0$ as long as $(w_{j,1}^0)^{k_1}\cdots(w_{j,P}^0)^{k_P} > 0$ has the same sign as $a_j^0\mu_D(\sigma)$ at initialization. Furthermore, the correlation with the support only appears at the end of the dynamics, and does not appear if we stop early.

The proof of Theorem 8 follows a similar proof strategy as Arous et al. (2021), namely a decomposition of the dynamics into a drift and martingale terms with information exponent D. However,

our problem is multi-index, and the analysis will require a tighter control of the different contributions to the dynamics as the dynamics move from saddle to saddle. An heuristic explanation for this result can be found in Appendix B.3. The complete proof of Theorem 8 is deferred to Appendix C.

The second layer weights training amounts to studying SGD on a linear model and is standard. The typical strategy consists in showing that the target function can be fitted with low-norm second-layer weights $\|\boldsymbol{a}\|_2$ (see for example Daniely and Malach (2020); Barak et al. (2022); Damian et al. (2022); Abbe et al. (2022b)). Because of the way we prove alignment of the first layer weights (weights $\pm\Delta$ on the support coordinates), we only prove this fitting for two specific monomials[10].

Corollary 9 (Second layer training, single monomial) *Let $h(\boldsymbol{z}) = z_1 \cdots z_D$ or $h(\boldsymbol{z}) = \mathrm{He}_D(z_1)$ and assume σ satisfies Assumption 7. For any constants $C > 0$ and $\varepsilon > 0$, there exist C_i for $i = 0, \dots, 11$, that only depend on D, K and C such that taking width $M = C_0\varepsilon^{C_0}$, bias initialization scale $\rho = \varepsilon^{C_1}/C_1$, and $\Delta = \varepsilon^{C_1}/C_1$ and second-layer initialization scale $\kappa = \frac{1}{C_2 M d^{C_2}}$, and second-layer regularization $\lambda_a = M\varepsilon/C_3$, and , and $r = \varepsilon^{C_4}/C_4$, and*

$$\overline{T}_1 = C_5 d^{D-1}\log(d)^{C_5}\,, \qquad \eta_1 = \frac{1}{C_6\kappa d^{D/2}\log(d)^{C_6}}\,,$$
$$\overline{T}_2 = C_7\varepsilon^{-C_7}\,, \qquad \eta_2 = 1/(C_8 M\varepsilon^{C_8})\,,$$

for $d \geq C_9\varepsilon^{-C_9}$ we have with probability at least $1 - d^{-C} - \varepsilon$:

(a) At the end of the dynamics,

$$R(\boldsymbol{\Theta}^{\overline{T}_1+\overline{T}_2}) \leq \varepsilon\,.$$

(b) If we train the first layer weights for $\overline{T}_1^0 \leq \overline{T}_1/(C_{10}\log(d)^{C_{10}})$ steps and for $M \leq C_{10}\log(d)$, then we cannot fit f using the second-layer weights, i.e.,

$$\min_{\boldsymbol{a}\in\mathbb{R}^M} \mathsf{E}_{\boldsymbol{x}}\Big[\Big(f(\boldsymbol{x}) - \sum_{j\in[M]} a_j\sigma(\langle \boldsymbol{w}_j^{\overline{T}_1^0}, \boldsymbol{x}\rangle)\Big)^2\Big] \geq 1 - \frac{\log(d)}{d^D}\,.$$

This result suggests that the dynamics of SGD with one monomial can be decomposed into a 'search phase' (plateau in the learning curve) and a 'fitting phase' (rapid decrease of the loss) similarly to Arous et al. (2021). SGD progressively aligns the first layer weights with the support, with little progress, and as soon as SGD picks up the support, the second layer weights can drive the risk quickly to 0. Because of the layer-wise training, we only show in Corollary 9.(b) that with early stopping on the training of the first layer weights, we cannot approximate the function f at all using the second layer weights (hence, we cannot learn it even with infinite number of samples). The proof of Corollary 9 is in Appendix E.1.

3.3. Learning multiple monomials

We now consider h with several monomials in its decomposition. In order to simplify the statement and the proofs, we will specifically consider the case of nested monomials

$$h(\boldsymbol{z}) = \sum_{l=1}^{L}\prod_{s\in[P_l]}\mathrm{He}_{k_s}(z_s)\,, \tag{11}$$

10. For general monomials, we would require more diversity on the first layer weights (for example, adding randomness on the ℓ_1 projection such that the weights are $\beta\Delta$ with $\beta \sim \mathrm{Unif}([1/2, 3/2])$). Again, these caveats are due to our proof technique to show alignment of the first layer weights.

where $0 =: P_0 < P_1 < P_2 < \ldots < P_L =: P$ and $k_1, \ldots, k_P$ are positive integers. For $l \in [L]$, we denote $D_l = k_{P_{l-1}+1} + \ldots + k_{P_l}$, and $D = \max_{l \in [L]} D_l$ the size of the biggest leap (such that h is a leap-D function), $\overline{D}_l = D_1 + \ldots + D_L$ and $\overline{D} := D_L$ the total degree of the polynomial h. We will assume that $\min_{l \in [L]} D_l \geq 2$ (i.e., leap of size at least 2 between monomials). This specific choice for h allows for a more compact proof, similar to Theorem 8. However, the compositionality of h is not a required structure for the sequential alignment to hold and we describe in Appendix D.2 how to modify the analysis for more general[11] h.

We first prove that the first-layer weights grow in the relevant directions during training.

Theorem 10 (First layer training) *Let $h : \mathsf{R}^P \to \mathsf{R}$ be defined as in Eq. (11) and assume σ satisfy Assumption 7. Then with the same choice of hyperparameters as in Theorem 8, with D now corresponding to the biggest leap, we have with probability at least $1 - Md^{-C}$: for any neuron $j \in [M]$ that satisfies $a^0 \mu_{\overline{D}_j}(\sigma)(w^0_{j,1})^{k_1} \cdots (w^0_{j,P_l})^{k_{P_l}} > 0$ for all $l \in [L]$,*

(a) On the support: $\left|w^{\overline{T}_1}_{j,i} - \text{sign}(w^0_{j,i}) \cdot \Delta\right| \leq C_5/\sqrt{d \log(d)}$ for $i = 1, \ldots, P$.

(b) Outside the support: $|w^{\overline{T}_1}_{j,i} - w^0_{j,i}| \leq C_6 r^2/\sqrt{d}$ for $i = P+1, \ldots, d$ and $\sum_{i>P}(w^{\overline{T}_1}_{j,i})^2 = 1$.

The proof follows by showing the sequential alignment of the weights to the support: with high probability and for each neurons satisfying the sign condition at initialization, it takes between $d^{\frac{D_l+D}{2}-1}/(C\log(d)^C)$ and $d^{\frac{D_l+D}{2}-1} C \log(d)^C$ steps to align with coordinates $[P_l]$, after having picked up coordinates $[P_{l-1}]$. The proof can be found in Appendix D.

While Theorem 10 captures the tight scaling in overall number of steps, it does not capture the number of steps for smaller leaps $D_l < D$ shown in Figure 1 in the case of increasing leaps. In Appendix D.2.1, we show that the scaling of $d^{D_l - 1}$ steps to align to the next monomial can be obtained by varying the step size, in the case of increasing leaps. Note that in practice, neural networks with constant step size seem to achieve this optimal scaling for escaping each saddle (such as in Figure 1). Hence, there might be a mechanism in the SGD training that can implicitly control the martingale part of the dynamics, without rescaling the step sizes. However, understanding such a mechanism would require to study the joint training of both layers, which is currently out of reach of our proof techniques.

As in the single monomial case, we consider fitting the second layer weights only for a specific class of functions (where all monomials are multilinear):

$$h(\boldsymbol{z}) = z_1 \cdots z_{P_1} + z_1 \cdots z_{P_2} + \ldots + z_1 \cdots z_{P_L}\,. \tag{12}$$

We require extra assumptions on the activation function to prove that the fitting is possible. The following is an informal statement, and we leave the formal statement and proof to Appendix E.2.

Corollary 11 (Second layer training, sum of monomials; informal statement) *Let $h(\boldsymbol{z})$ be as in (12). Then there is an activation function σ satisfying Assumption 7 such that for any $\varepsilon > 0$ there are choices of hyperparameters for SGD on a $\text{poly}(1/\epsilon)$-width two-layer network (Algorithm 1) such that for step counts $\overline{T}_1 = \tilde{\Theta}(d^{\text{Leap}(h)-1})$ and $\overline{T}_2 = \text{poly}(1/\epsilon)$ we have with high probability $R(\boldsymbol{\Theta}^{\overline{T}_1+\overline{T}_2}) \leq \varepsilon$.*

11. However, our current proof techniques do not allow for fully general leap functions: e.g., $h(\boldsymbol{z}) = \text{He}_2(z_1)\text{He}_3(z_2)\ \text{He}_3(z_1)\text{He}_2(z_2)$ has its two monomials pushing the $\boldsymbol{w}_j$'s in two opposite directions.

4. Discussion

Summary of contributions In this work, we have considered learning multi-index functions over the hypercube or the Gaussian measure. For classical linear methods, the complexity of the task is determined by the *degree* of the target function (Proposition 4). However, for neural networks, we have conjectured that the complexity is determined by the *leap complexity* of the target function (introduced in Definition 1). This would generalize the result of Abbe et al. (2022b), which shows the conjecture in the case of $\mathrm{Leap}(h\ ) = 1$. As evidence for this conjecture, we have proved lower-bounds showing $d^{\Omega(\mathrm{Leap}(h\ ))}$ complexity of learning in the Correlational Statistical Query (CSQ) framework (Proposition 5). Conversely, we have proved that $d^{O(\mathrm{Leap}(h\ ))}$ samples and runtime suffices for a modified version of SGD to successfully learn the relevant indices in the case of "leap-staircase" functions of the form (11), and to fit the function in the case of "multilinear leap-staircase" functions of the form (12).

Future work One direction for future work is to remove the modifications to vanilla SGD used in the analysis (layerwise training and the projection step). Another direction is to prove the conjecture by extending our analysis of the training dynamics to general functions, beyond those of the form (11). Another direction is to study extensions of the leap complexity measure beyond isotropic input distributions.

Acknowledgments

Part of this work was supported by the NSF-Simons Research Collaborations on the Mathematical and Scientific Foundations of Deep Learning (MoDL) Award and the EPFL PhD Exchange Fellowship. EB was also generously supported by Apple with an AI/ML fellowship. TM also acknowledges the NSF grant CCF-2006489 and the ONR grant N00014-18-1-2729.

References

Emmanuel Abbe and Enric Boix-Adsera. On the non-universality of deep learning: quantifying the cost of symmetry. *arXiv:2208.03113*, 2022.

Emmanuel Abbe and Colin Sandon. On the universality of deep learning. *Advances in Neural Information Processing Systems*, 33:20061–20072, 2020.

Emmanuel Abbe, Enric Boix-Adsera, Matthew S Brennan, Guy Bresler, and Dheeraj Nagaraj. The staircase property: How hierarchical structure can guide deep learning. *Advances in Neural Information Processing Systems*, 34:26989–27002, 2021a.

Emmanuel Abbe, Pritish Kamath, Eran Malach, Colin Sandon, and Nathan Srebro. On the power of differentiable learning versus pac and sq learning. *Advances in Neural Information Processing Systems*, 34:24340–24351, 2021b.

Emmanuel Abbe, Samy Bengio, Elisabetta Cornacchia, Jon Kleinberg, Aryo Lotfi, Maithra Raghu, and Chiyuan Zhang. Learning to reason with neural networks: Generalization, unseen data and boolean measures. *arXiv preprint arXiv:2205.13647*, 2022a.

Emmanuel Abbe, Enric Boix-Adsera, and Theodor Misiakiewicz. The merged-staircase property: a necessary and nearly sufficient condition for sgd learning of sparse functions on two-layer neural networks. In *Conference on Learning Theory*, pages 4782–4887. PMLR, 2022b.

Emmanuel Abbe, Elisabetta Cornacchia, Jan Hazla, and Christopher Marquis. An initial alignment between neural network and target is needed for gradient descent to learn. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 33–52. PMLR, 17–23 Jul 2022c.

Zeyuan Allen-Zhu and Yuanzhi Li. What can resnet learn efficiently, going beyond kernels? *Advances in Neural Information Processing Systems*, 32, 2019.

Zeyuan Allen-Zhu and Yuanzhi Li. Backward feature correction: How deep learning performs deep learning. *arXiv:2001.04413*, 2020.

Gerard Ben Arous, Reza Gheissari, and Aukosh Jagannath. Online stochastic gradient descent on non-convex losses from high-dimensional inference. *J. Mach. Learn. Res.*, 22:106–1, 2021.

Jimmy Ba, Murat A Erdogdu, Taiji Suzuki, Zhichao Wang, Denny Wu, and Greg Yang. High-dimensional asymptotics of feature learning: How one gradient step improves the representation. *arXiv:2205.01445*, 2022.

Francis Bach. Breaking the curse of dimensionality with convex neural networks. *The Journal of Machine Learning Research*, 18(1):629–681, 2017.

Boaz Barak, Benjamin L Edelman, Surbhi Goel, Sham Kakade, Eran Malach, and Cyril Zhang. Hidden progress in deep learning: Sgd learns parities near the computational limit. *arXiv:2207.08799*, 2022.

Benedikt Bauer and Michael Kohler. On deep learning as a remedy for the curse of dimensionality in nonparametric regression. 2019.

Shai Ben-David, Alon Itai, and Eyal Kushilevitz. Learning by distances. *Information and Computation*, 117(2):240–250, 1995.

Alberto Bietti, Joan Bruna, Clayton Sanford, and Min Jae Song. Learning single-index models with shallow neural networks. *arXiv:2210.15651*, 2022.

Avrim Blum and Ronald Rivest. Training a 3-node neural network is np-complete. *Advances in neural information processing systems*, 1, 1988.

Nader H Bshouty and Vitaly Feldman. On using extended statistical queries to avoid membership queries. *Journal of Machine Learning Research*, 2(Feb):359–395, 2002.

Emmanuel J Candes, Thomas Strohmer, and Vladislav Voroninski. Phaselift: Exact and stable signal recovery from magnitude measurements via convex programming. *Communications on Pure and Applied Mathematics*, 66(8):1241–1274, 2013.

Minshuo Chen, Yu Bai, Jason D Lee, Tuo Zhao, Huan Wang, Caiming Xiong, and Richard Socher. Towards understanding hierarchical learning: Benefits of neural representations. *Advances in Neural Information Processing Systems*, 33:22134–22145, 2020.

Sitan Chen and Raghu Meka. Learning polynomials in few relevant dimensions. In *Conference on Learning Theory*, pages 1161–1227. PMLR, 2020.

Yuxin Chen, Yuejie Chi, Jianqing Fan, and Cong Ma. Gradient descent with random initialization: Fast global convergence for nonconvex phase retrieval. *Mathematical Programming*, 176:5–37, 2019.

Lénaïc Chizat and Francis Bach. On the global convergence of gradient descent for over-parameterized models using optimal transport. *Advances in Neural Information Processing Systems*, 31:3036–3046, 2018.

Lenaic Chizat and Francis Bach. Implicit bias of gradient descent for wide two-layer neural networks trained with the logistic loss. In *Conference on Learning Theory*, pages 1305–1338. PMLR, 2020.

Jeremy M Cohen, Simran Kaur, Yuanzhi Li, J Zico Kolter, and Ameet Talwalkar. Gradient descent on neural networks typically occurs at the edge of stability. *arXiv preprint arXiv:2103.00065*, 2021.

Alexandru Damian, Jason Lee, and Mahdi Soltanolkotabi. Neural networks can learn representations with gradient descent. In *Conference on Learning Theory*, pages 5413–5452. PMLR, 2022.

Amit Daniely and Eran Malach. Learning parities with neural networks. *Advances in Neural Information Processing Systems*, 33:20356–20365, 2020.

Amit Daniely, Nati Linial, and Shai Shalev-Shwartz. From average case complexity to improper learning complexity. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 441–448, 2014.

Anindya De, Elchanan Mossel, and Joe Neeman. Is your function low dimensional? In *Conference on Learning Theory*, pages 979–993. PMLR, 2019.

Rishabh Dudeja and Daniel Hsu. Learning single-index models in gaussian space. In *Conference On Learning Theory*, pages 1887–1930. PMLR, 2018.

Rong Ge, Furong Huang, Chi Jin, and Yang Yuan. Escaping from saddle points—online stochastic gradient for tensor decomposition. In *Conference on learning theory*, pages 797–842. PMLR, 2015.

Behrooz Ghorbani, Song Mei, Theodor Misiakiewicz, and Andrea Montanari. Limitations of lazy training of two-layers neural network. *Advances in Neural Information Processing Systems*, 32, 2019.

Behrooz Ghorbani, Song Mei, Theodor Misiakiewicz, and Andrea Montanari. Linearized two-layers neural networks in high dimension. 2021.

Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In David S. Johnson, editor, *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washington, USA*, pages 25–32. ACM, 1989. doi: 10.1145/73007.73010. URL https://doi.org/10.1145/73007.73010.

Daniel Hsu. Dimension lower bounds for linear approaches to function approximation.

Daniel Hsu, Clayton Sanford, Rocco A Servedio, and Emmanouil-Vasileios Vlatakis-Gkaragkounis. On the approximation power of two-layer networks of random relus. *arXiv preprint arXiv:2102.02336*, 2021.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

Prateek Jain, Sham M Kakade, Rahul Kidambi, Praneeth Netrapalli, Venkata Krishna Pillutla, and Aaron Sidford. A markov chain theory approach to characterizing the minimax optimality of stochastic gradient descent (for least squares). *arXiv:1710.09430*, 2017.

Pritish Kamath, Omar Montasser, and Nathan Srebro. Approximate is Good Enough: Probabilistic Variants of Dimensional and Margin Complexity. In Jacob Abernethy and Shivani Agarwal, editors, *Proceedings of Thirty Third Conference on Learning Theory*, volume 125 of *Proceedings of Machine Learning Research*, pages 2236–2262. PMLR, 09–12 Jul 2020. URL https://proceedings.mlr.press/v125/kamath20b.html.

Michael Kearns. Efficient noise-tolerant learning from statistical queries. *Journal of the ACM (JACM)*, 45(6):983–1006, 1998.

Adam R Klivans and Alexander A Sherstov. Cryptographic hardness for learning intersections of halfspaces. *Journal of Computer and System Sciences*, 75(1):2–12, 2009.

Adam R Klivans, Ryan O'Donnell, and Rocco A Servedio. Learning intersections and thresholds of halfspaces. *Journal of Computer and System Sciences*, 68(4):808–840, 2004.

Michael Kohler and Adam Krzyżak. Nonparametric regression based on hierarchical interaction models. *IEEE Transactions on Information Theory*, 63(3):1620–1630, 2016.

Eyal Kushilevitz and Yishay Mansour. Learning decision trees using the fourier spectrum. *SIAM Journal on Computing*, 22(6):1331–1348, 1993. doi: 10.1137/0222080. URL https://doi.org/10.1137/0222080.

Yuanzhi Li, Tengyu Ma, and Hongyang R Zhang. Learning over-parametrized two-layer neural networks beyond ntk. In *Conference on learning theory*, pages 2613–2682. PMLR, 2020.

Eran Malach and Shai Shalev-Shwartz. The implications of local correlation on learning some deep functions. *Advances in Neural Information Processing Systems*, 33:1322–1332, 2020.

Yishay Mansour. *Learning Boolean Functions via the Fourier Transform*, pages 391–424. Springer US, Boston, MA, 1994. ISBN 978-1-4615-2696-4. doi: 10.1007/978-1-4615-2696-4_11. URL https://doi.org/10.1007/978-1-4615-2696-4_11.

Song Mei, Andrea Montanari, and Phan-Minh Nguyen. A mean field view of the landscape of two-layer neural networks. *Proceedings of the National Academy of Sciences*, 115(33):E7665–E7671, 2018.

Song Mei, Theodor Misiakiewicz, and Andrea Montanari. Mean-field theory of two-layers neural networks: dimension-free bounds and kernel limit. In *Conference on Learning Theory*, pages 2388–2464. PMLR, 2019.

Alireza Mousavi-Hosseini, Sejun Park, Manuela Girotti, Ioannis Mitliagkas, and Murat A Erdogdu. Neural networks efficiently learn low-dimensional representations with sgd. *arXiv:2209.14863*, 2022.

Eshaan Nichani, Yu Bai, and Jason D Lee. Identifying good directions to escape the ntk regime and efficiently learn low-degree plus sparse polynomials. *arXiv preprint arXiv:2206.03688*, 2022.

Ryan O'Donnell. *Analysis of Boolean Functions*. Cambridge University Press, 2014. doi: 10.1017/CBO9781139814782.

Maria Refinetti, Sebastian Goldt, Florent Krzakala, and Lenka Zdeborová. Classifying high-dimensional gaussian mixtures: Where kernel methods fail and neural networks succeed. In *International Conference on Machine Learning*, pages 8936–8947. PMLR, 2021.

Lev Reyzin. Statistical queries and statistical algorithms: Foundations and applications. *arXiv preprint arXiv:2004.00557*, 2020.

Itay Safran and Jason Lee. Optimization-based separations for neural networks. In *Conference on Learning Theory*, pages 3–64. PMLR, 2022.

David Saxton, Edward Grefenstette, Felix Hill, and Pushmeet Kohli. Analysing mathematical reasoning abilities of neural models. *arXiv preprint arXiv:1904.01557*, 2019.

Anselm Johannes Schmidt-Hieber. Nonparametric regression using deep neural networks with relu activation function. *Annals of statistics*, 48(4):1875–1897, 2020.

Taiji Suzuki and Shunta Akiyama. Benefit of deep learning with non-convex noisy gradient descent: Provable excess risk bound and superiority to kernel methods. *arXiv:2012.03224*, 2020.

Yan Shuo Tan and Roman Vershynin. Online stochastic gradient descent with arbitrary initialization solves non-smooth, non-convex phase retrieval. *arXiv:1910.12837*, 2019.

Matus Telgarsky. Feature selection with gradient descent on two-layer networks in low-rotation regimes. *arXiv:2208.02789*, 2022.

Santosh S Vempala. A random-sampling-based algorithm for learning intersections of halfspaces. *Journal of the ACM (JACM)*, 57(6):1–14, 2010.

Santosh S Vempala and Ying Xiao. Structure from local optima: Learning subspace juntas via higher order pca. *arXiv:1108.3329*, 2011.

Chiyuan Zhang, Maithra Raghu, Jon M. Kleinberg, and Samy Bengio. Pointer value retrieval: A new benchmark for understanding the limits of neural network generalization. *ArXiv*, abs/2107.12580, 2021.

Tong Zhang. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In *Proceedings of the twenty-first international conference on Machine learning*, page 116, 2004.

Appendix A. Additional numerical simulations

In Figures 2, 3, 4 and 5 we plot the risk versus number of samples for SGD training of 5-layer ResNets with fully-connected layers for various different target functions and for Boolean and Gaussian data. In these plots, the saddle-to-saddle dynamics are visible, which are caused by the neural network sequentially picking up the support using the hierarchical structure of the monomials in the function. In Figures 6 and 7, we study learning a leap-1 function (merged-staircase function), and we experiment with the effect of adding depth to see its effect on fitting. There is also an interesting edge-of-stability behavior during the "second-layer fitting" part, where the loss does not decrease monotonically (Cohen et al., 2021). We leave understanding this to future work.

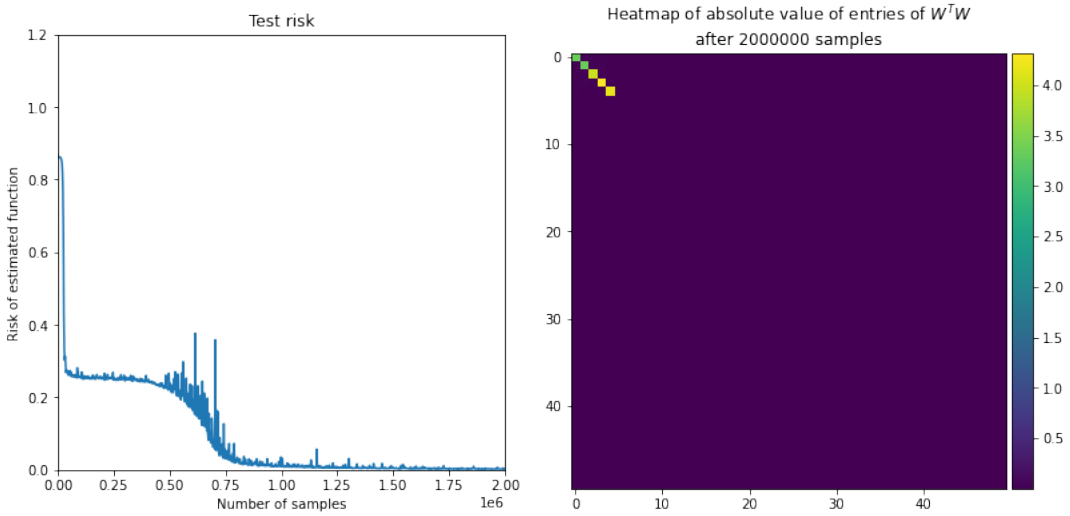
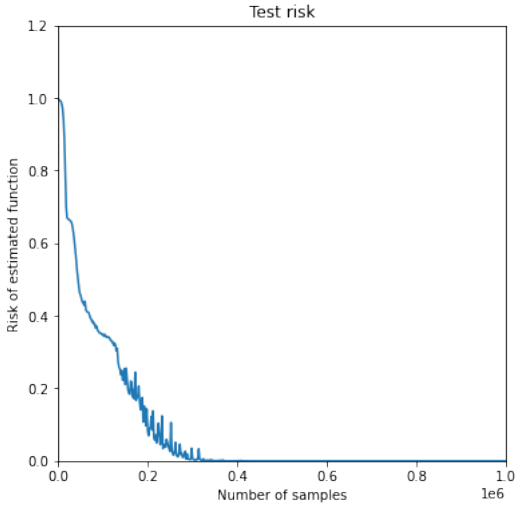
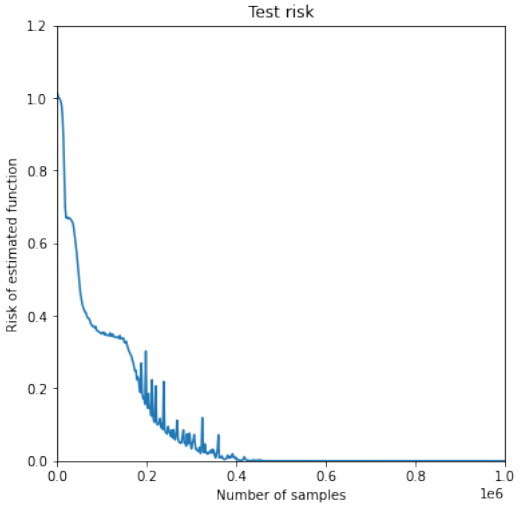


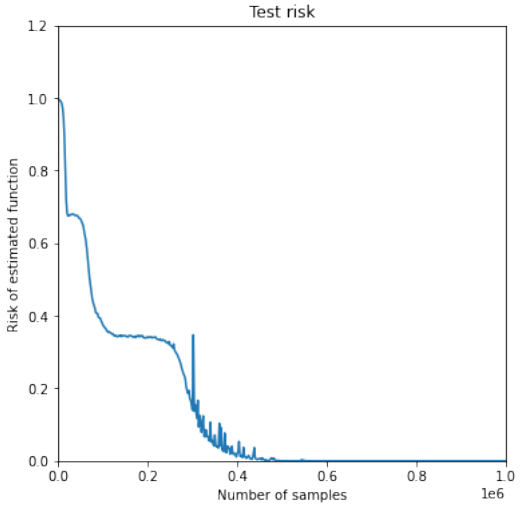
Figure 2: In this figure we consider training a 5-layer ResNet with fully-connected layers with SGD the leap-3 function $h(\boldsymbol{z}) = 2 \cdot \prod_{i=1}^{2} \tanh(z_i) + 5 \cdot \prod_{i=1}^{5} \tanh(z_i)$ with data $\boldsymbol{x} \sim \mathsf{N}(0, I_d)$ and $d = 50$. While our paper considered bounded degree polynomials, the leap complexity, which drives the sequential alignment to the support, also holds for non-polynomial functions. In this case, the leap depends on the first non-zero monomials in the Hermite decomposition. For h considered in this plot, we have first a leap of size 2 to align with x_1, x_2 followed by a leap of size 3 to align with $x_3x_4x_5$. In the plot of test risk over time, we indeed see first a short saddle to align with x_1, x_2, followed by a quick decrease of the loss (corresponding to the neural networks fitting $2\tanh(z_1)\tanh(z_2)$). This is followed by a plateau while SGD slowly picks up x_3, x_4, x_5 (saddle) and a sharp decrease in the loss when the neural network fit the remainder of h . We also plot the heatmap of the absolute value of the entries of $\boldsymbol{W}^{\top}\boldsymbol{W} \in \mathsf{R}^{d\ d}$ where $\boldsymbol{W}$ is the first-layer matrix after training. This shows that the first layer indeed picks up the relevant coordinates (first 5 coordinates) in the support after training.



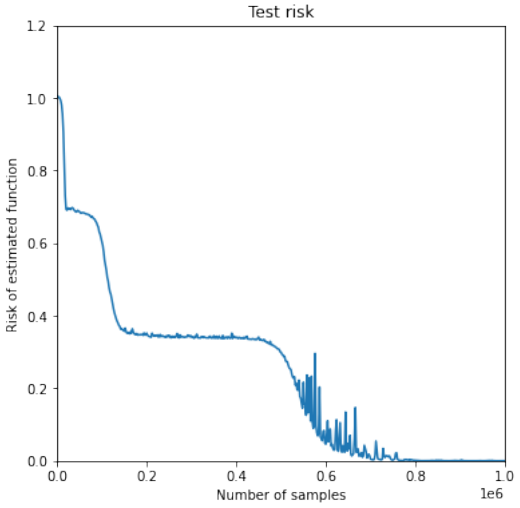
(a) Ambient dimension $d = 50$



(b) Ambient dimension $d = 100$

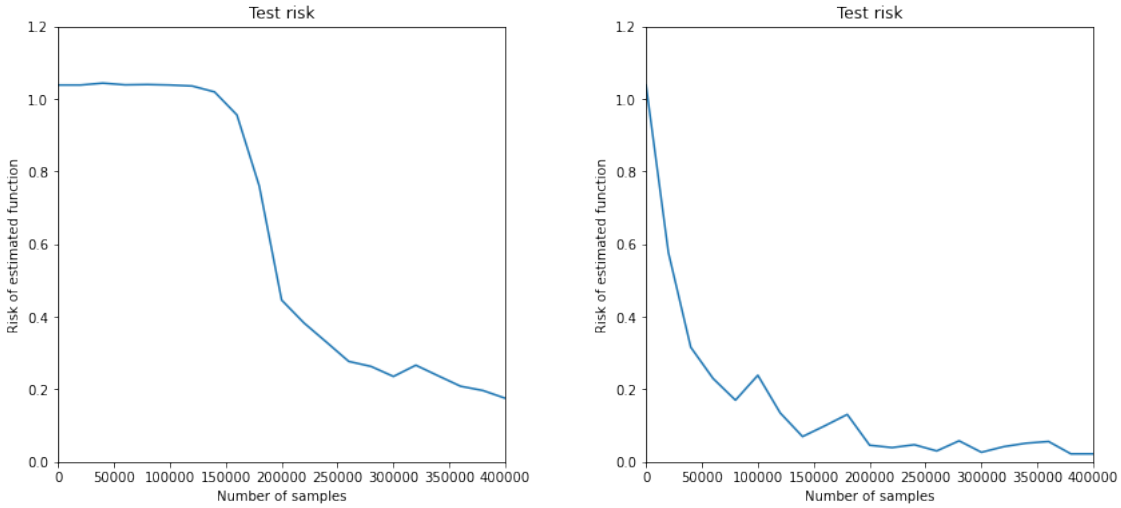


(c) Ambient dimension $d = 200$



Ambient dimension $d = 400$

Figure 3: In (a)-(d) we show the evolution of the risk for training a 5-layer ResNet with fully-connected layers with SGD to learn the leap-3 function $h(\boldsymbol{z}) = z_1 + z_1 z_2 z_3 + z_1 z_2 z_3 z_4 z_5 z_6$ with binary hypercube data in ambient dimension $d = 50, 100, 200, 400$, respectively. Notice that the evolution of the risk follows a saddle-to-saddle dynamic. This dynamic becomes more salient as the ambient dimension increases and escaping the saddles dominates the SGD trajectory.



(a) Leap-3 function $h\ (\boldsymbol{z}) = \mathrm{He}_3(z_1)$ (b) Leap-1 function $h\ (\boldsymbol{z}) = \mathrm{He}_1(z_1) + \mathrm{He}_3(z_1)$

Figure 4: We consider training a 5-layer ResNet with fully-connected layers with SGD on covariate distribution $\boldsymbol{x} \sim \mathsf{N}(0, I_d)$ with $d = 500$. In (a) we show the risk from learning the leap-3 function $h\ (\boldsymbol{z}) = \mathrm{He}_3(z_1)$, and in (b) we show the risk from learning the leap-1 function $h\ (\boldsymbol{z}) = \mathrm{He}_1(z_1) + \mathrm{He}_3(z_1)$. Notice that the leap-3 task is much more difficult for SGD, and it gets stuck in a saddle where the loss plateaus. On the other hand, the $\mathrm{He}_1(z_1)$ term in the leap-1 task means that SGD is not stuck in a saddle.

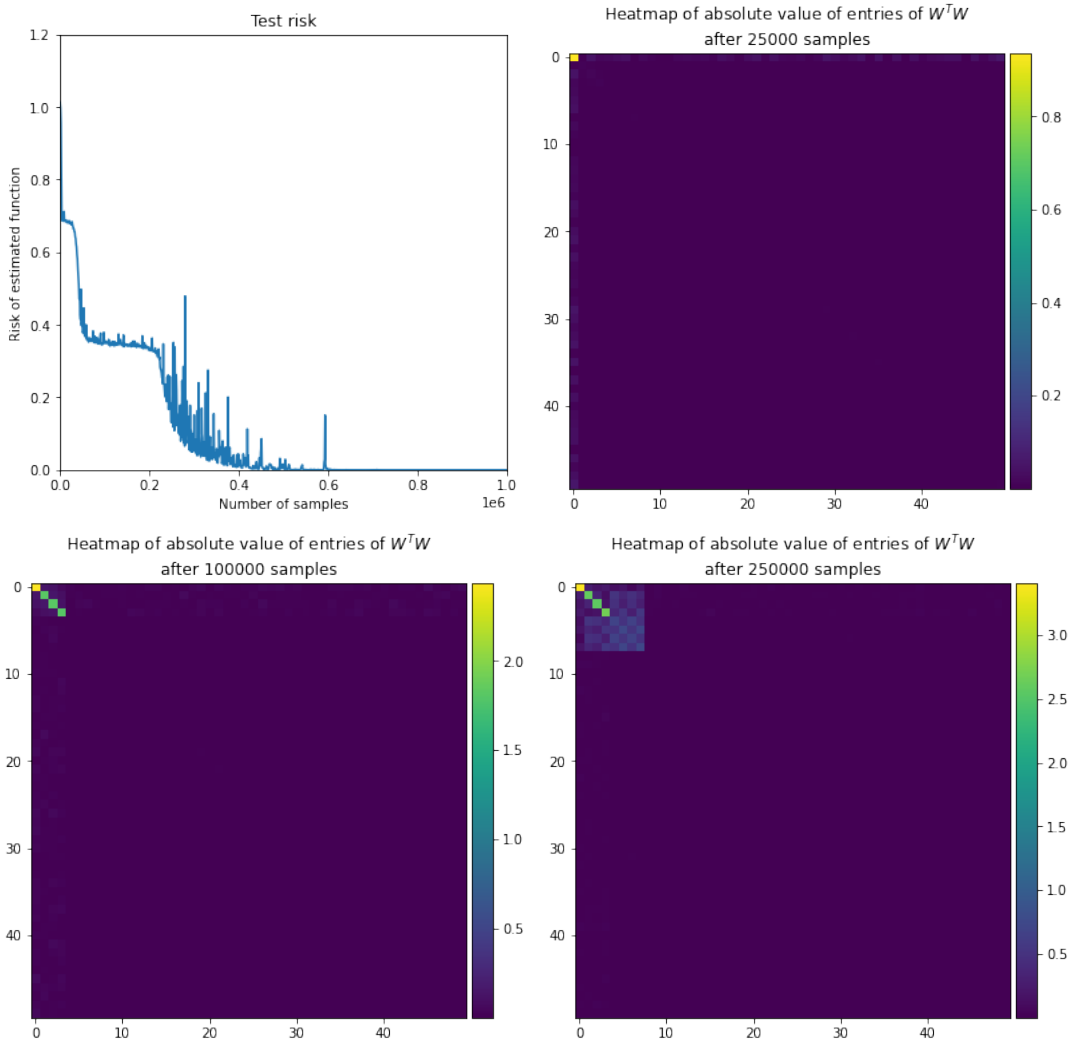


Figure 5: A width-1000 5-layer ResNet network with ReLU activation trained with one-pass SGD with mini-batch size 100 and step size 0.1. The data is $\boldsymbol{x} \sim \{+1, -1\}^d$ for ambient dimension $d = 50$, and $h\ (z) = z_1 + z_1z_2z_3z_4 + z_1z_2z_3z_4z_5z_6z_7z_8$, which is a leap-4 function. We observe saddle-to-saddle dynamics. And we observe that the first layer picks up the relevant support iteratively.

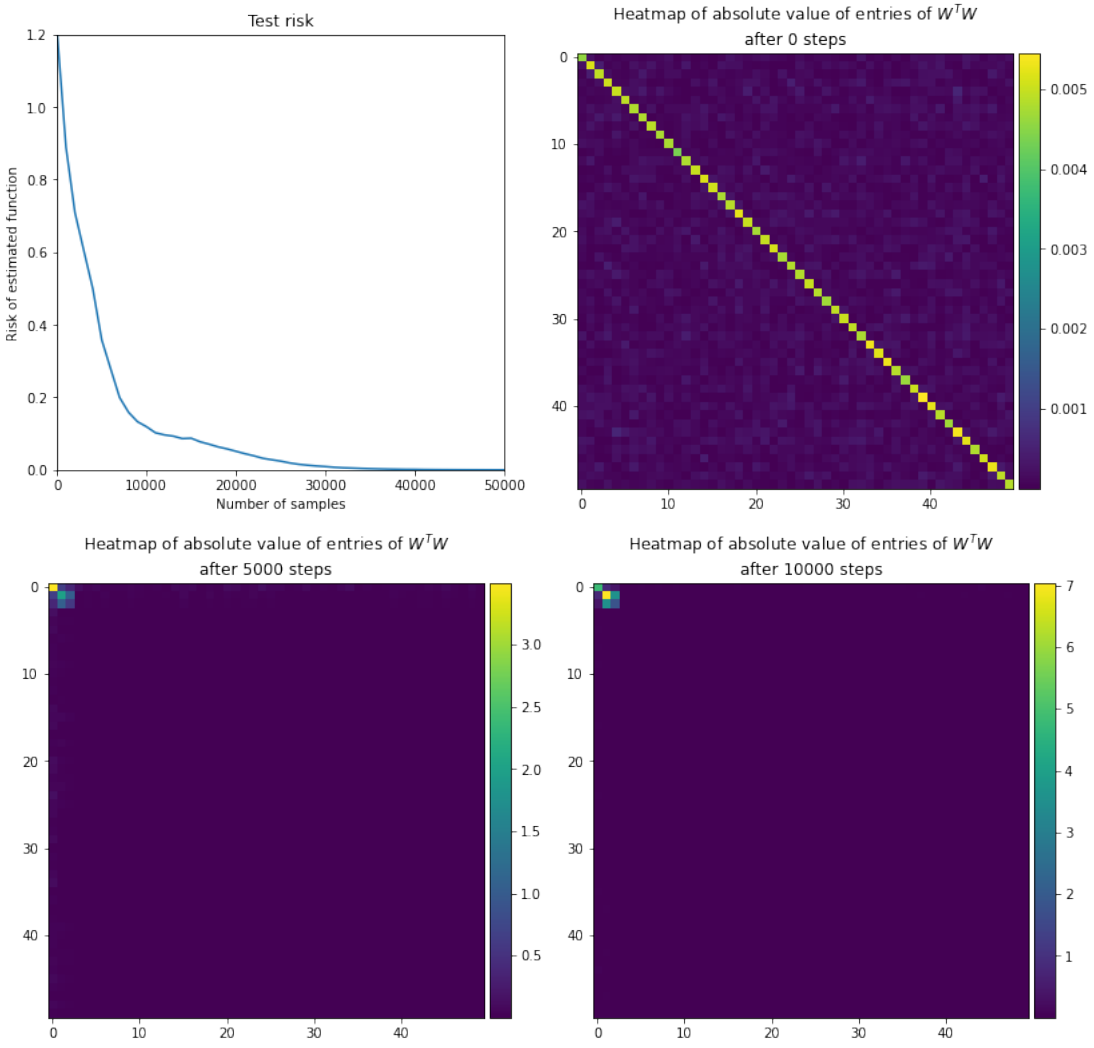
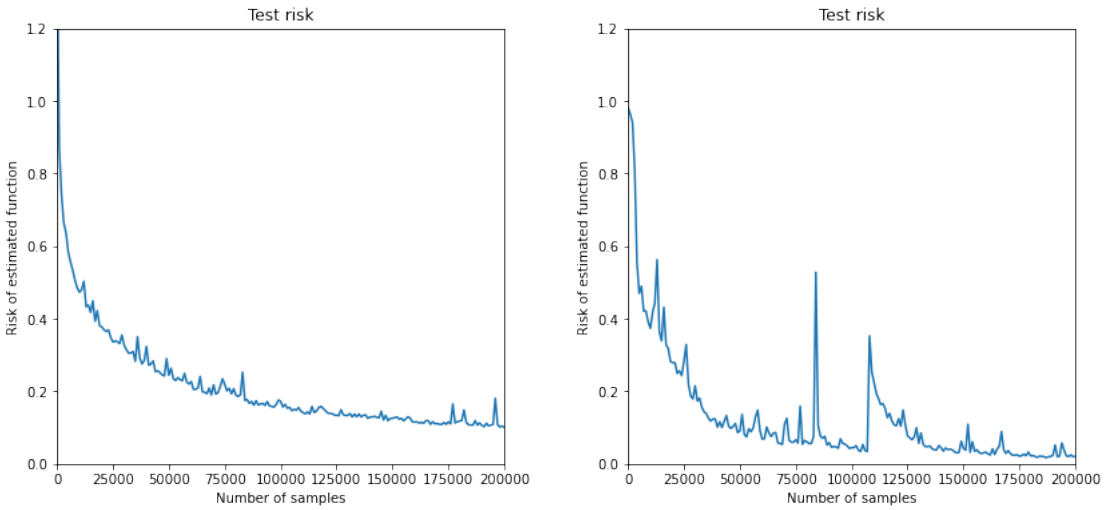


Figure 6: We train a width-1000, 2-layer network with sigmoid activation with mini-batch size 100 and learning rate 0.5. The data is from the Boolean hypercube with ambient dimension $d = 50$, and the target function is $h\ (z) = z_1 + z_1z_2 + z_1z_2z_3$, which is a leap-1 function. Notice that the weights quickly align to the support of the function (no saddles) after less than 5000 steps.



(a) 2-layer network

(b) depth-5 ResNet

Figure 7: We consider either (a) training a width-1000, 2-layer network with sigmoid activation or (b) training a width-1000, 5-layer ResNet network with ReLU activation and fully-connected layers. Our ambient dimension is $d = 50$, our data is $\boldsymbol{x} \sim \mathsf{N}(0, I_d)$ and our target function is $h\ (z) = z_1 + z_1 z_2 + z_1 z_2 z_3$. This is a leap-1 function, so the weights quickly align to the coordinates x_1, x_2, x_3 after a small number of steps. However, the two-layer neural network struggles to fit the different monomials in h . This can be mitigated by training a deeper network which finds a better fit faster. Hence, besides the alignment phenomenon to the low-dimensional support explored in this paper, it is an interesting question for future work to understand why depth helps in this situation. Note that this is a different phenomenon than the one explored in depth-separation papers such as Safran and Lee (2022), which considers learning functions which cannot be efficiently approximated by 2-layer neural networks (here, h can be easily approximated with a two-layer network).

Appendix B. Additional discussion from the main text

B.1. Additional references

In addition to the references listed in the main text, we further review other relevant papers.

A line of work in computational learning theory studied the complexity of learning Boolean functions under the uniform input distribution. It was realized that functions with concentrated Fourier spectrum can be learned efficiently, both in sample and time complexity using the sparse Fourier algorithm (Mansour, 1994). Namely, under knowledge of a set of basis elements $\mathcal{S}$ such that $\sum_{S\not\in\mathcal{S}} f^2(S) \geq 1-\epsilon/2$ for all $f \in \mathcal{F}$, one can learn $\mathcal{F}$ with error ϵ, sample complexity $O((1/\epsilon)|\mathcal{S}|\log(|\mathcal{S}|/\delta))$ and polynomial time complexity if $|\mathcal{S}|$ is polynomial using the sparse Fourier algorithm that estimates the coefficients in $\mathcal{S}$. Many interesting classes of functions fall under this setting, such as juntas, low degree functions, bounded-size or -depth decision trees (O'Donnell, 2014). While $\mathcal{S}$ has to be known[12] under the random sample model, no degree constraints are imposed. In particular, the low-degree assumption (degree at most k) is just a special case that provides this knowledge (with order d^k time complexity), monomials of degree k or $d-k$ are equivalent in the eye of the sparse Fourier algorithm. This is not necessarily the case for SGD-trained neural networks.

A line of work has considered SGD learning on 'unconstrained' neural networks (besides polynomial size) and shows that we can emulate any efficient PAC or SQ algorithm (Abbe and Sandon, 2020; Abbe et al., 2021b; Malach and Shalev-Shwartz, 2020). Such networks are far from the practical neural networks used in applications. Against this state of affairs, several works have attempted to derive computational lower bounds on learning with regular neural networks. For example, Abbe et al. (2022c) shows that for fully connected 2-layer networks, if the initial alignment (INAL) of a network with a Boolean target function (measured by the maximal expected correlation between target and neurons) is not significant, then noisy-GD cannot amplify the correlation to any significant level. This is achieved by showing that a low INAL implies a large minimal degree in the target function (thus a large leap) under some additional conditions. Another work (Abbe and Boix-Adsera, 2022) uses the permutation, sign-flip, or rotational equivariance of noisy-GD training of fully-connected neural networks to show a lower bound on the number of gradient descent steps required for global convergence, when we have access to population gradients with an additive Gaussian noise. In particular, for leap-L functions on the hypercube and the hypercube, $\Omega(d^L\tau^2)$ steps are to shown to be required, where τ^2 is the Gaussian noise variance. This roughly matches the conjecture in this paper in its exponential dependence on the leap – however, the computational model is different (noisy-GD versus online-SGD).

Finally let's remark that a large body of work in the statistics and machine learning literature has studied the problem of learning multi-index models. These include for example phase retrieval (Candes et al., 2013), intersection of halfspaces (Klivans et al., 2004; Vempala, 2010) and subspace juntas (Vempala and Xiao, 2011; De et al., 2019). We refer to Dudeja and Hsu (2018); Chen and Meka (2020) and references therein for an overview of this line of work. In particular, it is well understood that in order to break the "curse of dimensionality", the algorithm needs to estimate the low-dimensional support. In contrast with this line of work, we consider learning these multi-index functions with generic SGD on regular neural networks, with no a priori information on the target

12. The set knowledge can be relaxed under the query access model using the Kushilevitz-Mansour algorithm (based on the Goldreich-Levin algorithm) that uses a divide-and-conquer procedure to identify the coefficients to be estimated Kushilevitz and Mansour (1993); Goldreich and Levin (1989).

function. Surprisingly, we show that this generic algorithm can nearly match the computational complexity of the best CSQ algorithm. Note that specialized algorithms can achieve better sample and computational complexity: for example, Chen and Meka (2020) showed an algorithm that can learn low-rank Gaussian polynomials in $\widetilde{O}_d(d)$ samples and $\widetilde{O}_d(d^3)$ runtime, regardless of the leap-complexity, by going beyond CSQ algorithms.

B.2. Discussion on the definition of the leap complexity

It was noted in Abbe et al. (2022b) that some "degenerate" leap-1 functions on the hypercube are not learned in $\Theta(d)$ SGD-steps. Take for example $h(\boldsymbol{z}) = z_1 + z_2 + z_3 + z_1z_2z_3$: by permutation symmetry on the support of h, $O(d)$ steps of SGD will learn first layer weights $\boldsymbol{w}_j$ aligned with $(1,1,1)$ on the support (z_1, z_2, z_3). SGD will require many more steps to break this symmetry[13] and fit h. Abbe et al. (2022b) circumvents this difficulty under a smoothed complexity analysis, and shows that the set of degenerate leap-1 functions has $\{\hat{h}(S)\}_{S2S}$ of Lebesgue-measure 0. Alternatively, a possible approach to learn these degenerate cases (for "axis-aligned" sparse functions) is to use different random learning rates for each coordinates and break the symmetry in learning.

On the other hand, Gaussian data offers a more natural setting to understand these degenerate functions: indeed, the decomposition (4) depends on the specific coordinate axis used to define the product of Hermite polynomials. In particular, $\mathrm{Leap}(h)$ will depend on the specific basis for this expansion. By rotational symmetry of the Gaussian distribution and equivariance of neural networks with isotropic initialization, the time complexity of SGD will be driven by the following "isotropic leap" complexity:

$$\mathrm{isoLeap}(h) = \max_{R2O_P} \mathrm{Leap}(h, R)\,,$$

where $\mathrm{Leap}(h, R)$ corresponds to the leap complexity of Definition 1, where we made the dependency on the specific choice of axis R for the Hermite expansion explicit. Adapting the proofs of Abbe et al. (2022b), we can show that if $\mathrm{isoLeap}(h) > 1$, then h cannot be learned in $\Theta(d)$ SGD-steps in the mean-field regime, while if $\mathrm{isoLeap}(h) = 1$, then the span of $\boldsymbol{w}_j$'s covers the entire support of h and not a subspace as for degenerate functions[14]. Consider for example the case of $h(\boldsymbol{z}) = z_1 + z_2 + z_1z_2$: in this coordinate basis, h is a leap-1 function[15]. However, we can consider the following rotation $(u_1, u_2) = (z_1 + z_2, z_1 - z_2)/\sqrt{2}$ and $h(\boldsymbol{z}) = u_1 + \mathrm{He}_2(u_1)/\sqrt{8} - \mathrm{He}_2(u_2)/\sqrt{8}$ in this basis. Therefore $\mathrm{isoLeap}(h) = 2$ and h cannot be learned in $\Theta(d)$ SGD steps in the mean-field regime.

For technical reasons, we consider in Section 3 the support and the decomposition of h aligned with the canonical basis of $\boldsymbol{x}$. This can be seen as a smoothed complexity setting similar to Abbe et al. (2022b): almost surely over the Hermite coefficients, the basis that maximizes the leap is the original basis in Eq. (4), i.e., fixing the axis coordinates R, then $\mathrm{isoLeap}(h) = \mathrm{Leap}(h, R)$ almost surely over the Hermite coefficients $\{\hat{h}(S)\}_{S2S}$ in the basis aligned with R. However, we stress here that isoLeap is the right measure for SGD-complexity in the case of general (not axis-aligned) low-dimensional support.

13. We conjecture $\Theta(d\log(d)^C)$ steps are required, see following discussion in the Gaussian case.
14. Proving full learnability in $\Theta(d)$ steps of functions with $\mathrm{isoLeap}(h) = 1$ on Gaussian data is technically challenging and not the purpose of the current paper, and would require a separate analysis, see Abbe et al. (2022b) for details on what it would entail.
15. Note that $\mathrm{He}_1(x) = x$ and we can rewrite $h(\boldsymbol{z}) = \mathrm{He}_1(z_1) + \mathrm{He}_1(z_2) + \mathrm{He}_1(z_1)\mathrm{He}_1(z_2)$.

B.3. Intuition for the proof of Theorem 8

In this section, we give some intuition behind the proof of Theorem 8. The complete proof can be found in Appendix C.

We first consider a simple SGD dynamics, with no projection step, and neglect the biases. We later discuss our choice of algorithm and how the analysis needs to be modified to control the projection step. The dynamics on the first layer weights is now simply given by

$$\boldsymbol{w}_j^{t+1} = \boldsymbol{w}_j^t + \eta_1 a_j^0 (y^t - \hat{f}_{\mathsf{NN}}(\boldsymbol{x}^t; \boldsymbol{\Theta}^t)) \sigma^{\emptyset}(\langle \boldsymbol{w}_j^t, \boldsymbol{x}^t \rangle) \boldsymbol{x}^t \,.$$

Reduction to a correlation flow: Recall that we initialize the second layer weights $|a_j^0| = \kappa$. By Assumption 7, we have

$$|\hat{f}_{\mathsf{NN}}(\boldsymbol{x}^t; \boldsymbol{\Theta}^t)| \le \sum |a_j^0| |\sigma(\langle \boldsymbol{w}_j^t, \boldsymbol{x}^t \rangle)| \le MK\kappa \,.$$

With high probability over a polynomial number of steps, $\|\boldsymbol{x}^t\|_1 \le C \log(d)$ with C constant chosen sufficiently large. Hence,

$$\boldsymbol{w}_j^{t+1} = \boldsymbol{w}_j^t + \eta_1 a_j^0 y^t \sigma^{\emptyset}(\langle \boldsymbol{w}_j^t, \boldsymbol{x}^t \rangle) \boldsymbol{x}^t + O(M \eta_1 \kappa^2 \log(d)) \,,$$

and we can chose κ, η_1 with $\eta_1 \kappa^2$ sufficiently small, while keeping $\eta_1 \kappa$ constant, so that we can neglect the interaction term between the different neurons and get:

$$\boldsymbol{w}_j^{t+1} \approx \boldsymbol{w}_j^t + \eta a_j^0 y^t \sigma^{\emptyset}(\langle \boldsymbol{w}_j^t, \boldsymbol{x}^t \rangle) \boldsymbol{x}^t \,.$$

This means that in this scaling and with high probability, the derivative of the square loss $\ell(y, \hat{y}) = \frac{1}{2}(y - \hat{y})^2$ in $\hat{y}$ is well-approximated by $\ell^{\emptyset}(y, \hat{y}) \approx -y$. Under this approximation, the different neurons do not interact while the first-layer weights are being fit, and so the neuron dynamics can be analyzed individually.

Heuristic derivation of $\overline{T}_1$ and η_1: Let us directly consider the correlation loss and track the dynamics of a unique neuron $(a, \boldsymbol{w})$. We assume that $w_1^0 = \ldots = w_d^0 = 1/\sqrt{d}$, $a^0 = \kappa$ and $\mu_D(\sigma) > 0$. We further make the following heuristic simplification: we assume the dynamics is described by only two parameters

$$\alpha_1^t = w_1^t = \ldots = w_P^t \,, \qquad \alpha_2^t = w_{P+1}^t = \ldots = w_d^t \,,$$

with SGD updates $g_1^t = y^t \sigma^{\emptyset}(\langle \boldsymbol{w}^t, \boldsymbol{x}^t \rangle) x_1^t$ and $g_2^t = y^t \sigma^{\emptyset}(\langle \boldsymbol{w}^t, \boldsymbol{x}^t \rangle) x_{P+1}^t$, i.e.,

$$\alpha_1^{t+1} = \alpha_1^t + \eta_1 \kappa g_1^t \,,$$
$$\alpha_2^{t+1} = \alpha_2^t + \eta_1 \kappa g_2^t \,.$$

The computation follows from a similar strategy as in Tan and Vershynin (2019); Arous et al. (2021): namely, we will decompose the dynamics into a drift term (deterministic) and a martingale term. Introduce the population gradient $\overline{g}_i^t = \mathsf{E}_{y^t, \boldsymbol{x}^t}[g_i^t]$, $i \in \{1, 2\}$. We can decompose the dynamics into a sum of population gradients (deterministic drift) and the martingale difference between the stochastic gradients and population gradients (recall that the (y^t, x^t)'s are independent):

$$\alpha_i^{t+1} = \alpha_i^t + \eta_1 \kappa \overline{g}_i^t + \eta_1 \kappa (g_i^t - \overline{g}_i^t) = \alpha_i^0 + \eta_1 \kappa \sum_{s=0}^{t} \overline{g}_i^t + \eta_1 \kappa \sum_{s=0}^{t} (g_i^t - \overline{g}_i^t) \,. \tag{13}$$

By using that for Hermite polynomials $\mathsf{E}_G[\mathrm{He}_k(G)f(G)] = \mathsf{E}_G[f^{(k)}(G)]$ with $G \sim \mathsf{N}(0,1)$, we can show that

$$\mathsf{E}[\mathrm{He}_D(g_1)f(\boldsymbol{u}^T\boldsymbol{g})g_1] = Du_1^{D-1}\mathsf{E}[f^{(D-1)}(\boldsymbol{u}^T\boldsymbol{g})] + u_1^{D+1}\mathsf{E}[f^{(D+1)}(\boldsymbol{u}^T\boldsymbol{g})]\,,$$
$$\mathsf{E}[\mathrm{He}_D(g_1)f(\boldsymbol{u}^T\boldsymbol{g})g_2] = u_1^D u_2\mathsf{E}[f^{(D+1)}(\boldsymbol{u}^T\boldsymbol{g})]\,,$$

where $\boldsymbol{u} = (u_1, u_2)$ and $\boldsymbol{g} = (g_1, g_2) \sim \mathsf{N}(0, \mathbf{I}_2)$. We deduce that to leading term (assuming $\|\boldsymbol{w}^t\|_2 \approx 1$)

$$\overline{g}_1^t = \mathsf{E}[h'(\boldsymbol{z})\sigma'(\langle \boldsymbol{w}^t, \boldsymbol{x}\rangle)x_1] \approx (\alpha_1^t)^{D-1}\mathsf{E}[\sigma^{(D)}(\|\boldsymbol{w}^t\|G)] \approx \mu_D(\sigma)(\alpha_1^t)^{D-1}\,,$$
$$\overline{g}_2^t = \mathsf{E}[h'(\boldsymbol{z})\sigma'(\langle \boldsymbol{w}^t, \boldsymbol{x}\rangle)x_{P+1}] \approx (\alpha_1^t)^D\alpha_2^t\mathsf{E}[\sigma^{(D+2)}(\|\boldsymbol{w}^t\|G)] \approx \mu_{D+2}(\sigma)(\alpha_1^t)^D\alpha_2^t\,.$$

Let us now control the different contributions to the dynamics:

(i) **Martingale part:** By Doob's maximal inequality for martingales, we have with high probability

$$\sup_{1 \le t \le \overline{T}_1} \left|\eta_1\kappa\sum_{s=0}^{t}(g_i^t - \overline{g}_i^t)\right| \lesssim \eta_1\kappa\sqrt{\overline{T}_1}\,.$$

We choose $\eta_1\kappa$ so that we can neglect the martingale contribution during the entire dynamics by taking $\eta_1\kappa\sqrt{\overline{T}_1} \lesssim \alpha_i^0 = d^{-1/2}$.

(ii) **Drift part for α_1:** We now neglect the martingale term and write for all $0 \le t \le \overline{T}_1 - 1$

$$\alpha_1^{t+1} \approx \alpha_1^0 + \eta_1\kappa\sum_{s=0}^{t}\overline{g}_1^t \approx \alpha_1^0 + \eta_1\kappa\mu_D(\sigma)\sum_{s=0}^{t}(\alpha_1^s)^{D-1}\,. \tag{14}$$

We can study this sequence (see Arous et al. (2021)) and show that

$$\alpha_1^t \approx \frac{1}{\big((\alpha_1^0)^{-(D-2)} - \eta_1\kappa\mu_D(\sigma)t\big)^{1/(D-2)}}\,.$$

In order for $\alpha_1^{\overline{T}_1} \approx 1$, we need to take $\eta_1\kappa\mu_D(\sigma)\overline{T}_1 \gtrsim (\alpha_1^0)^{-(D-2)} = d^{D/2-1}$.

(iii) **Drift part for α_2:** Again, by neglecting the martingale contribution and for $0 \le t \le \overline{T}_1 - 1$,

$$\alpha_2^{t+1} \approx \alpha_2^0 + \eta_1\kappa\mu_{D+2}(\sigma)\sum_{s=0}^{t}(\alpha_1^s)^D\alpha_2^s\,.$$

We can show that this sequence is bounded by

$$\begin{aligned}\ln\left(\frac{\alpha_2^{t+1}}{\alpha_2^0}\right) &\lesssim \eta_1\kappa\mu_{D+2}(\sigma)\sum_{s=0}^{t}(\alpha_1^s)^D \\ &\le \frac{\mu_{D+2}(\sigma)}{\mu_D(\sigma)}\eta_1\kappa\mu_D(\sigma)\sum_{s=0}^{t}(\alpha_1^s)^{D-1} \le \frac{\mu_{D+2}(\sigma)}{\mu_D(\sigma)}\alpha_1^{t+1}\,,\end{aligned} \tag{15}$$

where we used Eq. (14) in the last inequality.

We deduce from Eq. (15) that for $\overline{T}_1$ chosen such that $\alpha_1^{\overline{T}_1} \approx 1$, then $\ln(\alpha_2^{t+1}/\alpha_2^0) \;.\; 1$. Hence, during the dynamics, the weights α_2^t not aligned with the support of h remain small, of order $1/\sqrt{d}$, while the weights α_1^t aligned with the support of h become of order 1. From the bounds in (i) and (ii), we need to choose η_1 and $\overline{T}_1$ such that $\eta_1\kappa\sqrt{\overline{T}_1} \;.\; d^{\;1/2}$ (martingale part) and $\eta_1\kappa\overline{T}_1 \;\&\; d^{D/2\;\;1}$ (drift part), i.e., we can take

$$\overline{T}_1 \approx d^{D\;\;1}\,, \qquad \eta_1 \approx \frac{1}{\kappa d^{D/2}}\,,$$

which matches the scaling in Theorem 8.

Adding a projection step: While the above heuristic derivation was useful to get intuitions, the assumption that the weights remain equal (or approximately equal) on and outside the support is not valid. Because of the statistical fluctuations over $\widetilde{\Theta}(d^{D\;\;1})$ steps, different coordinates over different neurons will grow to be order 1 on the support at a stochastic time (with high probability between $d^{D\;\;1}/(C\log(d)^C)$ and $d^{D\;\;1}C\log(d)^C$ for some large enough constant C). To prevent these coordinates to continue growing (because we neglected the interaction term in the dynamics, which could otherwise prevent this growth), we introduce the projection step

$$\begin{cases} \widetilde{\boldsymbol{w}}^{t+1} = & \boldsymbol{w}^t + ay^t\eta_1 \cdot \mathrm{grad}_{\boldsymbol{w}^t}\sigma(\langle \boldsymbol{w}^t, \boldsymbol{x}^t\rangle)\,, \\ \boldsymbol{w}^{t+1} = & \mathsf{P}^{t+1}\mathsf{P}_{\mathit{1}}\,\widetilde{\boldsymbol{w}}^{t+1}\,, \end{cases} \tag{16}$$

where $\mathsf{P}^{t+1}\mathsf{P}_{\mathit{1}}$ is the projection step defined in Eq. (10), and we use the spherical gradient defined in Eq. (9). Note that because of the choice $\Delta > r$ and the definition of the set $S_{j,t}$ on which we do the projection on the sphere, P^{t+1} and P_{1} commute.

Thanks to the spherical gradient, we can show that the projection steps only have a negligible impact on the dynamics (similarly to the analysis in Arous et al. (2021)). By carefully arranging these additional terms, we can essentially recover the drift plus martingale analysis presented heuristically above.

B.4. Going beyond sparsity

In the $P = O_d(1)$ regime the complexity scaling in d is dominated by the 'hard' part of learning the low-dimensional latent space on which the function depends, and the complexity of fitting the function on the support is secondary and only results in constants. This also makes the conjecture fairly general in terms of architecture choices as long as there is enough expressivity to fit the function on the support. One could also consider functions that depend on a finite number of basis elements, without necessarily involving a finite number of coordinates. For instance the full parity $\prod_{i2[d]} x_i$ function is such an example. For SQ algorithms, the class of monomials of degree 0 (more generally k) has equivalent complexity to the class of monomials of degree d (more generally degree $d-k$), and the SQ-dimension is symmetrical for these dual cases. However for SGD learning on regular nets, this is not exactly the case. It is true that the full parity can be learned by regular nets under a specific setting; Abbe and Boix-Adsera (2022) provides a regular 2-layer neural net that can learn the full parity if the weight measure of the first layer at initialization is i.i.d. Rademacher(1/2) and the activation is a ReLU. A constant number of step can also be sufficient in such cases, as for the 0-degree monomial. It is however conjectured that this is not achievable with a polynomial number of steps for weights that have a Gaussian initialization. Thus, for isotropic layers, it is

possible that the full parity is not polytime learnable. This means that the generalized notion of leap to non-coordinate sparse may depend on more specific choices of the parameters. Further, in the non-isotropic case where the full parity is efficiently learnable, one may define the leap with basis sets that can either grow from the 0-monomial or descend from the full-monomial, with the mirror symmetry as for SQ algorithms.

Another notion to factor in when considering non-coordinate sparse function is the fitting of the function once the support is learned. First of all, there may be a non-polynomial number of coefficients to handle, although one can probably cover enough interesting cases with functions that are well-approximated by polynomially many coefficients (O'Donnell, 2014). Further, there is the fitting of the function by the neural net that may now turn non-trivial. Consider even a function with few basis elements, $h\ (\boldsymbol{z}) = \sum_{i=1}^{P} ix_i + \prod_{i=1}^{P} x_i^{g(i)}$, where $g : [P] \to \{0, 1\}$ is an arbitrary, but known function, and $P \gg 1$ is large. SGD on a regular neural network would first pick up the P coordinates in the support and then learn the monomial $\prod_{i=1}^{P} z_i^{g(i)}$ based on that support. The latter part may not be trivial for SGD on a regular net, while it would require 0 queries for an SQ algorithm (once the linear part is learned, the permutation is identified and the coefficients in front of each variable would allow us to calculate $g(i)$). Thus the complexity of learning the second monomial on the detected support set is likely to factor in for such cases, and this is likely going to depend more on the model hyperparameters and architecture choice. In less contrived cases, the naive generalization of the leap applied verbatim to non-constant P remains likely relevant.

B.5. Lower-bounds: beyond noisy GD

Note that the CSQ and noisy-GD models do not exactly match the SGD learning model; we do prove in this paper that the drift of the population gradient dominates the dynamic on the considered horizon, but the CSQ model also has noise added to the query outputs. It is nonetheless interesting that the regularity of the network model drives us to an achievability result that matches that of CSQ lower-bounds. Since it is known how to go beyond the CSQ/SQ lower-bounds with non-regular networks Abbe and Sandon (2020), e.g., learning dense parities by emulating matrix inversions with irregular networks, our results raise an intriguing question: may the model "regularity" act comparably to a CSQ constraint? We leave this to future work.

A result in Abbe et al. (2022b) does derive a lower-bound that does not require additive noise and that applies to online-SGD. This work requires however a few restrictions: the mean-field parametrization (and not just any isotropic distribution) and a linear sample complexity (e.g., finite number of time steps with linear batches). These are used to derive a specialization of the mean-field PDE approximation (Mei et al., 2018) to the coordinate sparse setting. In turn, this allows to show that the sample complexity for SGD learning on functions that do not satisfy the merged-staircase property ($\mathrm{Leap} = 1$) cannot be learned with a linear sample complexity.

Appendix C. Proof of Theorem 8: alignment with a single monomial

In this appendix, we prove the alignment of the first layer's weights with the support of one monomial. The proof will follow from a similar proof strategy as in Tan and Vershynin (2019); Arous et al. (2021), namely decomposing the dynamics into drift and martingale terms. However, it will differ in a key aspect: while Arous et al. (2021) considers a single-index model, we will need to track for each neuron P parameters (the first P coordinates of $\boldsymbol{w}_j$) and show that the $d-P$ other parameters remain well behaved along their whole trajectories, which requires a tighter control of the different contributions to the dynamics.

Recall that we denote by K a constant that only depends on σ (Assumption 7) and the sub-Gaussianity of the label noise ε. Throughout the proofs, we will write $C, c > 0$ for generic constants that only depend on D and K. The values of these constants are allowed to change from line to line or within the same line.

C.1. Preliminaries

In the proof, we will consider $0 < r \le \Delta \le 1$ to be small enough constants that can depend on D and K, but are independent of d. We will track the dependency in r, Δ when necessary, and otherwise use that they are bounded by 1 (in particular, the constants c, C in the proof will be independent of r, Δ). These constants r, Δ will be fixed in Theorem 9.

We will show that we can take initialization scale κ of second layer weights a^0 and step size η such that the dynamics of the first layer training can be approximated by a correlation dynamics, with no interactions between the neurons, so that we can analyze each neuron independently. We consider below an arbitrary neuron $(a_j, b_j, \boldsymbol{w}_j)$ for $j \in [M]$. In the case that $a_j^0 \mu_D(\sigma(\cdot + b_j))(w_{j,1})^{k_1} \cdots (w_{j,P}^0)^{k_P} > 0$ we prove that the event claimed in Theorem 8.(b) and (c) holds with probability at least $1 - d^{-C}$ for neuron j. Theorem 8.(a) will follow from a similar analysis. The result for all neurons follows by a union bound.

We further consider $|b_j^0| \le \rho \le \Delta$ small enough such that $1/2 \le |\mu_k(\sigma(\cdot + b_j))|/|\mu_k(\sigma)| \le 3/2$ for $k = 0, \dots, D+2$ (see comments below Lemma 13). Hence, the biases will not impact the training of the first layer weights and for the simplicity, we will fix $b_j = 0$ in the proof.

Nonnegative first layer weights Without loss of generality, we assume that all of the first-layer coordinates of neuron j have positive sign at initialization $w_{j,1}^0 = \ldots = w_{j,d}^0 = 1/\sqrt{d}$ (and therefore $a^0 \mu_D(\sigma) > 0$ by our choice of $(a_j^0, \boldsymbol{w}_j^0)$). To see why, define $s_0 = \prod_{i 2 [P]} \text{sign}(w_{j,i}^0)^{k_i}$ and consider instead initializing the network at $\breve{\boldsymbol{\Theta}}^0 = (\breve{\boldsymbol{a}}^0, \breve{\boldsymbol{W}}^0)$ where $\breve{\boldsymbol{a}}^0 = s_0 \boldsymbol{a}^0$ and $\breve{\boldsymbol{w}}_{j^0}^0 = \boldsymbol{w}_{j^0} \odot \text{sign}(\boldsymbol{w}_j^0)$ for all j^0. Then consider training the network with samples $(\breve{y}^t, \breve{\boldsymbol{x}}^t)$ where $\breve{\boldsymbol{x}}^t = \boldsymbol{x}^t \odot \text{sign}(\boldsymbol{w}_j^0)$ and $\breve{y}^t = f\ (\breve{\boldsymbol{x}}^t) + \varepsilon_t$. The distribution of data $(\breve{y}^t, \breve{x}^t)$ is the same as that of $(y^t, \boldsymbol{x}^t)$, and the the training dynamics $\breve{\boldsymbol{\Theta}}$ match those of $\boldsymbol{\Theta}$ up to sign flips, and $\breve{\boldsymbol{w}}_j^0 = [1/\sqrt{d}, \dots, 1/\sqrt{d}]$.

For brevity of notation, let us drop the subscript j in the remainder of the analysis of this section, and write $(a, \boldsymbol{w})$ to denote $(a_j, \boldsymbol{w}_j)$ whenever it can be inferred from context.

Stopping times on the dynamics: Recall that the loss is the square loss $\ell(y, \hat{y}) = \frac{1}{2}(y - \hat{y})^2$. We denote $\boldsymbol{v}^t$ the (negative) stochastic gradient at time t:

$$\boldsymbol{v}^t := -\nabla_{\boldsymbol{w}} \ell\big(y^t, \hat{f}_{\mathsf{NN}}(\boldsymbol{x}^t; \boldsymbol{\Theta}^t)\big)\big) = (y^t - \hat{f}_{\mathsf{NN}}(\boldsymbol{x}^t; \boldsymbol{\Theta}^t)) a^0 \sigma^0(\langle \boldsymbol{w}^t, \boldsymbol{x}^t \rangle) \boldsymbol{x}^t \,,$$

Further recall that we constrain the dynamics of $\boldsymbol{w}^t$ in two ways: the ℓ_1 constraint $\|\boldsymbol{w}^t\|_1 \le \Delta$ and the spherical constraint $\|\mathcal{S}_t(\boldsymbol{w}^t)\|_2 = 1$, where $\mathcal{S}_t(\boldsymbol{w}^t) = (w_i^t \mathbb{1}_{i \in S_t})_{i \in [d]}$ is the projection on the subset of coordinates S_t which contains all coordinates that verify $|\widetilde{w}_i^{t'}| < r$ for all times $t' \le t$. Let us define $\widetilde{\boldsymbol{v}}^t$ the spherical gradient update on support S_t:

$$\widetilde{\boldsymbol{v}}^t = -\operatorname{grad}_{\boldsymbol{w}^t} \ell(y^t, \hat{f}_{\mathsf{NN}}(\boldsymbol{x}^t; \boldsymbol{\Theta}^t)) = \boldsymbol{v}^t - \mathcal{S}_t(\boldsymbol{w}^t)\langle \mathcal{S}_t(\boldsymbol{w}^t), \boldsymbol{v}^t \rangle$$

The update equations are given by

$$\begin{cases} \widetilde{\boldsymbol{w}}^{t+1} = & \boldsymbol{w}^t + \eta_1 \widetilde{\boldsymbol{v}}^t \,, \\ \overline{\boldsymbol{w}}^{t+1} = & \mathsf{P}_1 \, \widetilde{\boldsymbol{w}}^{t+1} \,, \\ \boldsymbol{w}^{t+1} = & \mathsf{P}^{t+1} \overline{\boldsymbol{w}}^{t+1} \,, \end{cases}$$

where we recall that we defined

$$(\mathsf{P}^{t+1} \overline{\boldsymbol{w}}^{t+1})_i = \begin{cases} \overline{w}_i^{t+1} & \text{if } i \notin S_{t+1} \,, \\ \frac{\overline{w}_i^{t+1}}{k S_{t+1}(\overline{\boldsymbol{w}}^t) k_2} & \text{if } i \in S_{t+1} \,, \end{cases}$$

with $S_{t+1} = S_t \setminus \{i \in [d] : |\widetilde{w}_i^{t+1}| \ge r\}$.

Let us introduce the following stopping times on the dynamics:

$$\begin{aligned} \tau^+ &= \inf\Big\{ t \ge 0 : \max_{i=P+1,\dots,d} \{|\widetilde{w}_i^{t+1}| \vee |w_i^{t+1}|\} \ge 3/(2\sqrt{d}) \Big\} \,, \\ \tau &= \inf\Big\{ t \ge 0 : \min_{i \in [d]} \{|\widetilde{w}_i^{t+1}| \wedge |w_i^{t+1}|\} \le 1/(2\sqrt{d}) \Big\} \,, \\ \tau^0 &= \inf\Big\{ t \ge 0 : \max(\|\boldsymbol{v}^t/a^0\|_1 \,, \|\widetilde{\boldsymbol{v}}^t/a^0\|_1 \,, |y^t| \,, \|\boldsymbol{x}^t\|_1) \ge C_0 \log(d)^{C_0} \Big\} \,. \end{aligned}$$

Note that $\{\tau = t\} \in \mathcal{F}_t := \sigma\big(\boldsymbol{\Theta}^0, \{\boldsymbol{x}^s, y^s\}_{s \le t}\big)$ for $\tau \in \{\tau^+, \tau, \tau^0\}$ and $\sigma(\boldsymbol{w}^{t+1}), \sigma(S_{t+1}) \subseteq \mathcal{F}_t$. For $t \le \tau^+$ and $r \ge 3/(2\sqrt{d})$, we have $\{P+1, \dots, d\} \subseteq S_t$, and $\|\boldsymbol{w}^t\|_2 \le \|\boldsymbol{w}^t_{1:P}\|_2 + \|\mathcal{S}_t(\boldsymbol{w}^t)\|_2 \le \sqrt{P}\Delta + 1$. We further define for all $i \in [d]$,

$$\begin{aligned} \tau_i^r &= \inf\Big\{ t \ge 0 : |\widetilde{w}_i^{t+1}| \ge r \Big\} \,, \\ \tau_i^\Delta &= \inf\Big\{ t \ge 0 : |w_i^{t+1}| \ge \Delta - |a^0| \eta_1 C_0 \log(d)^{C_0} \Big\} \,, \\ \tau^r &= \sup_{i \in [P]} \tau_i^r \,, \qquad\qquad \tau^\Delta = \sup_{i \in [P]} \tau_i^\Delta \,, \end{aligned}$$

where C_0 is a constant that will be chosen large enough. In particular, at time $\tau_i^r + 1$, the i-th coordinate is removed from the set on which we do the projection, i.e., $\{i\} \subseteq S_{\tau_i^r} \setminus S_{\tau_i^r + 1}$. We will show in the proof that $\tau^+ \wedge \tau \wedge \tau^0 > \overline{T}_1$ with high probability.

By concentration of polynomials of Gaussian variables, we have:

Lemma 12 *Assume that $\Delta \le 1$. Then for any $C > 0$, there exists C_0 large enough that only depends on C, D and K, such that for $d \ge 2$,*

$$\mathsf{P}(\tau^0 \le \tau^+ \wedge d^D) \le d^{-C} \,.$$

Proof [Proof of Lemma 12] For $t \le \tau^+$, we must have $\|\boldsymbol{w}^t\|_2 \le \sqrt{P}+1$. Using the bounds (53) in Lemma 16 and a union bound, there exists a constant C_0 such that

$$\begin{aligned}\mathsf{P}(\tau^0 \le \tau^+ \wedge d^D) &\le \sum_{t \ \tau^+ \wedge d^D} \mathsf{P}_{\boldsymbol{w}^t}\big(\max(\|\boldsymbol{v}^t/a^0\|_{\mathit{1}}, \|\widetilde{\boldsymbol{v}}^t/a^0\|_{\mathit{1}}, |y^t|, \|\boldsymbol{x}^t\|_{\mathit{1}}) > z \ + c\big) \\ &\le c d^{D+1} \exp(-C(z\ )^{2/(3D+3)}) \le d^{\ -C}\ ,\end{aligned}$$

where $z\ = C_0 \log(d)^{C_0} - c$. ∎

Reducing to the correlation flow We now show that for second-layer initialization scale κ small enough, the updates $\boldsymbol{v}^t$ and $\tilde{\boldsymbol{v}}^t$ mostly come from correlation term in the square loss, and the self-interaction term contributes negligibly. Define the gradient and spherical gradient from the correlation term as:

$$\boldsymbol{u}^t = a^0 y^t \sigma^0(\langle \boldsymbol{w}^t, \boldsymbol{x}^t\rangle)\boldsymbol{x}^t \qquad \text{and} \qquad \tilde{\boldsymbol{u}}^t = \boldsymbol{u}^t - \mathcal{S}_t(\boldsymbol{w}^t)\langle \mathcal{S}_t(\boldsymbol{w}^t), \boldsymbol{u}^t\rangle\,.$$

Then $\boldsymbol{v}^t$ is close to $\boldsymbol{u}^t$ and $\tilde{\boldsymbol{v}}^t$ is close to $\tilde{\boldsymbol{u}}^t$. For $t < \tau^0$, we have the following bounds. Use (a) $\|\sigma\|_{\mathit{1}}, \|\sigma^0\|_{\mathit{1}} \le K$ and $\|\boldsymbol{a}^0\|_{\mathit{1}} = \kappa$, and (b) $\|\mathcal{S}_t(\boldsymbol{w}^t)\|_{\mathit{1}} < r < 1$ and $\|\mathcal{S}_t(\boldsymbol{w}^t)\|_1 < d$,

$$\begin{aligned}\|\boldsymbol{v}^t - \boldsymbol{u}^t\|_{\mathit{1}}/|a^0| &\overset{(a)}{\le} \|\hat{f}_{\mathsf{NN}}(\boldsymbol{x}^t; \boldsymbol{\Theta}^t)\sigma^0(\langle \boldsymbol{w}^t, \boldsymbol{x}^t\rangle)\boldsymbol{x}^t\|_{\mathit{1}} \le \kappa K^2 M C_0 \log(d)^{C_0} \le \tilde{\kappa} \\ \|\tilde{\boldsymbol{v}}^t - \tilde{\boldsymbol{u}}^t\|_{\mathit{1}}/|a^0| &= \|\boldsymbol{v}^t - \boldsymbol{u}^t - \mathcal{S}_t(\boldsymbol{w}^t)\langle \mathcal{S}(\boldsymbol{w}^t), \boldsymbol{v}^t - \boldsymbol{u}^t\rangle\|_{\mathit{1}}/|a^0| \\ &\overset{(b)}{\le} (1+d)\|\boldsymbol{v}^t - \boldsymbol{u}^t\|_{\mathit{1}} \le \tilde{\kappa}\,,\end{aligned} \tag{17}$$

for $\tilde{\kappa} = 2\kappa d K^2 M C_0$.

Simplifying the update equations: Note that for $t < \tau_i^\Delta \wedge \tau_0$, we have $|\widetilde{w}_i^{t+1}| = |w_i^t + \eta_1 \widetilde{v}_i^t| \le \Delta$, and therefore $\overline{w}_i^{t+1} = \widetilde{w}_i^{t+1}$. Let us introduce the truncated spherical gradient $\boldsymbol{g}^t$ defined by

$$g_i^t := \begin{cases} \widetilde{v}_i^t & \text{for } t < \tau_i^\Delta \wedge \tau_0\ , \\ \gamma_i^t \widetilde{v}_i^t & \text{for } t \ge \tau_i^\Delta \wedge \tau_0\ , \end{cases}$$

where $\gamma_i^t \in [0,1]$ is a multiplicative factor that models the projection step $\overline{\boldsymbol{w}}^{t+1} = \mathsf{P}_{\mathit{1}}\,\widetilde{\boldsymbol{w}}^{t+1}$,

$$\gamma_i^t = \min\Big(\frac{\Delta - \operatorname{sign}(\widetilde{v}_i^t) w_i^t}{\eta_1 |\widetilde{v}_i^t|}, 1\Big)\,.$$

It is easy to check that $\sigma(\boldsymbol{g}^t) \subseteq \mathcal{F}_t$ and $\overline{\boldsymbol{w}}^{t+1} = \mathsf{P}_{\mathit{1}}\,(\boldsymbol{w}^t + \eta_1 \widetilde{\boldsymbol{v}}^t) = \boldsymbol{w}^t + \eta_1 \boldsymbol{g}^t$ for all $t \ge 0$. With these notations, our dynamics are now simply given by

$$\begin{cases} \overline{\boldsymbol{w}}^{t+1} = & \boldsymbol{w}^t + \eta_1 \boldsymbol{g}^t\,, \\ \boldsymbol{w}^{t+1} = & \mathsf{P}^{t+1} \overline{\boldsymbol{w}}^{t+1}\,. \end{cases} \tag{18}$$

Population gradients: Let us define the population spherical gradient $\overline{\boldsymbol{g}}^t = \mathsf{E}_{\boldsymbol{x}^t,y^t}[\boldsymbol{g}^t]$. We have the following formula on $\overline{g}_i^t$ for $t < \tau_i^\Delta \wedge \tau_0$:

Lemma 13 *Denote $\chi\ (\boldsymbol{w}^t) = \prod_{j \in [P]} (w_j^t)^{k_j}$. For $i \in [P]$ and $t < \tau_i^\Delta \wedge \tau_0$, we have the following formulas that approximate the population gradient: if $t \le \tau_i^r$ (i.e., $i \in S_t$), then $\overline{g}_i^t = \mathsf{E}_{\boldsymbol{x}^t,y^t}\left[\widetilde{v}_i^t\right]$ and*

$$\left| \overline{g}_i^t - a^0 \frac{\chi\ (\boldsymbol{w}^t)}{w_i^t} \left(k_i - (w_i^t)^2 \sum_{j \in S_t \setminus [P]} k_j \right) \mathsf{E}_G\left[\sigma^{(D)}(\|\boldsymbol{w}^t\|_2 G)\right] \right| \le |a^0| \tilde{\kappa}\,, \tag{19}$$

while if $t > \tau_i^r$ (i.e., $i \notin S_t$), then $\overline{g}_i^t = \mathsf{E}_{\boldsymbol{x}^t,y^t}[v_i^t]$ and

$$\left| \overline{g}_i^t - a^0 \frac{\chi\ (\boldsymbol{w}^t)}{w_i^t} \left(k_i \mathsf{E}_G\left[\sigma^{(D)}(\|\boldsymbol{w}^t\|_2 G)\right] + (w_i^t)^2 \mathsf{E}_G\left[\sigma^{(D+2)}(\|\boldsymbol{w}^t\|_2 G)\right] \right) \right| \le |a^0| \tilde{\kappa}\,. \tag{20}$$

For $i > P$ and $t < \tau^+ \wedge \tau_0$,

$$\left| \overline{g}_i^t + a^0 w_i^t \chi\ (\boldsymbol{w}^t) \left(\sum_{j \in S_t \setminus [P]} k_j \right) \mathsf{E}_G\left[\sigma^{(D)}(\|\boldsymbol{w}^t\|_2 G)\right] \right| \le |a^0| \tilde{\kappa}\,. \tag{21}$$

Proof [Proof of Lemma 13] We recall the following useful identities (where $G \sim \mathsf{N}(0,1)$)

$$\mathsf{E}_G[\mathrm{He}_k(G) g(G)] = \mathsf{E}_G[g^{(k)}(G)]\,, \qquad x\mathrm{He}_k(x) = \mathrm{He}_{k+1}(x) + k\mathrm{He}_{k\ \ 1}(x)\,.$$

In particular, by integration by parts, we have

$$\mathsf{E}\Big[\prod_{j \in [P]} \mathrm{He}_{v_j}(x_j) \sigma^{()}(\langle \boldsymbol{w}^t, \boldsymbol{x} \rangle) \Big] = \Big(\prod_{j \in [P]} (w_j^t)^{v_j} \Big) \cdot \mathsf{E}_G\big[\sigma^{(1+v_1+\ldots+v_P)}(\|\boldsymbol{w}^t\|_2 G)\big]\,.$$

Furthermore, if $i \in [P]$,

$$x_i f\ (\boldsymbol{x}) = \Big(\prod_{j \in [P], j \neq i} \mathrm{He}_{k_j}(\boldsymbol{x}_j) \Big) \big\{ k_i \mathrm{He}_{k_i\ \ 1}(x_i) + \mathrm{He}_{k_i+1}(x_i) \big\}\,.$$

Hence, for $i \in [P]$ and $i \in S_t$, we have $|\overline{g}_i^t - \mathsf{E}_{\boldsymbol{x}^t,y^t}[\tilde{u}_i^t]| \le |a^0| \tilde{\kappa}$ by (17), and

$$\begin{aligned}
\mathsf{E}_{\boldsymbol{x}^t,y^t}[\tilde{u}_i^t] &= \mathsf{E}_{\boldsymbol{x}^t,\varepsilon^t}\Big[a^0 (f\ (\boldsymbol{x}^t) + \varepsilon^t) \Big\{ x_i^t - w_i^t \sum_{j \in S_t} w_j^t x_j^t \Big\} \sigma^{()}(\langle \boldsymbol{w}^t, \boldsymbol{x}^t \rangle) \Big] \\
&= a^0 \frac{\chi\ (\boldsymbol{w}^t)}{w_i^t} \Big\{ k_i - (w_i^t)^2 \sum_{j \in S_t \setminus [P]} k_j \Big\} \mathsf{E}_G\big[\sigma^{(D)}(\|\boldsymbol{w}^t\|_2 G)\big] \\
&\quad + a^0 \chi\ (\boldsymbol{w}^t) \Big\{ w_i^t - w_i^t \sum_{j \in S_t} (w_j^t)^2 \Big\} \mathsf{E}_G\big[\sigma^{(D+2)}(\|\boldsymbol{w}^t\|_2 G)\big]\,,
\end{aligned}$$

which gives Eq. (19) by using $\|\mathcal{S}_t(\boldsymbol{w}^t)\|_2^2 = 1$. Eqs. (20) and (21) are obtained similarly. ∎

From Assumption 7, we can choose Δ small enough and depending only on K and D such that that for all $u, v \in [-P\Delta, P\Delta]$ and $0 \le k \le D$

$$\left|\mathsf{E}_G[\mathrm{He}_k(G)\sigma((1+v)G+u)] - \mu_k(\sigma)\right| \le \frac{|\mu_k(\sigma)|}{2}\,. \tag{22}$$

and for $k = D+1$ or $D+2$,

$$\left|\mathsf{E}_G[\mathrm{He}_k(G)\sigma((1+v)G+u)]\right| \le 2K\,. \tag{23}$$

We further assume that Δ is chosen small enough such that $\Delta^2 \le 1/(2D)$ and $\Delta^2 \le 1/(4K^2)$. With this choice of Δ, there exist constants $C, c > 0$ that only depend on D, K such that for all $t < \tau_i^\Delta \wedge \tau^+ \wedge \tau$, if $i \in [P]$,

$$ca^0 \frac{\chi(\boldsymbol{w}^t)}{w_i^t}\mu_D(\sigma) - |a^0|\tilde{\kappa} \le \overline{g}_i^t \le Ca^0 \frac{\chi(\boldsymbol{w}^t)}{w_i^t}\mu_D(\sigma) + |a^0|\tilde{\kappa}\,,$$

where we recall that we are now assuming that $\mathrm{sign}(w_i^0) = 1$ and therefore $\mathrm{sign}(w_i^t) = 1$ for $t < \tau$, and $a^0\chi(\boldsymbol{w}^0)\mu_D(\sigma) > 0$. Because $t < \tau$, and because κ is chosen small enough so that $\tilde{\kappa} \ll (1/(2\sqrt{d}))^{D+1}$, we have

$$ca^0 \frac{\chi(\boldsymbol{w}^t)}{w_i^t}\mu_D(\sigma) \le \overline{g}_i^t \le Ca^0 \frac{\chi(\boldsymbol{w}^t)}{w_i^t}\mu_D(\sigma)\,. \tag{24}$$

And if $i > P$,

$$|\overline{g}_i^t| \le Ca^0\mu_D(\sigma)w_i^t\chi(\boldsymbol{w}^t)\Big\{\sum_{j \in S_t \setminus [P]} k_j\Big\} + |a^0|\tilde{\kappa}\,.$$

Notice that $|\overline{g}_i^t| \le |a^0|\tilde{\kappa}$ for all $i > P$ and $t \ge \tau^r + 1$ (i.e., $S_t \cap [P] = \emptyset$).

C.2. Bounding the different contributions to the dynamics

The following lemma tracks the contribution of the projection on the sphere $\boldsymbol{w}^{t+1} = \mathsf{P}^{t+1}\overline{\boldsymbol{w}}^{t+1}$:

Lemma 14 *Assume that $C_0 \log(d)^{C_0}/\sqrt{d} \le r \le \Delta/2 \le 1/(\sqrt{8P})$, $|a^0| \le 1$ and $\eta_1 \le 1/d$. Then there exist constants $C, C^\emptyset > 0$ (that only depend on D, K and C_0) such that for $d \ge C^\emptyset$ and all $t < \tau^0 \wedge \tau^+$, if $S_{t+1} = S_t$,*

$$\frac{1}{2} \le 1 - C\eta_1^2\|\boldsymbol{g}^t\|_2^2 \le \frac{1}{\|\mathcal{S}_t(\overline{\boldsymbol{w}}^{t+1})\|_2} \le 1 + C\eta_1^2\|\boldsymbol{g}^t\|_2^2\,, \tag{25}$$

and if $S_{t+1} \neq S_t$, then

$$\frac{1}{2} \le 1 - Cr^2 \le \frac{1}{\|\mathcal{S}_t(\overline{\boldsymbol{w}}^{t+1})\|_2} \le 1 + Cr^2\,. \tag{26}$$

Proof [Proof of Lemma 14] First consider the case $S_{t+1} = S_t$. We have $\mathcal{S}_{t+1}(\overline{\boldsymbol{w}}^{t+1}) = \mathcal{S}_t(\boldsymbol{w}^t) + \eta_1\mathcal{S}_t(\boldsymbol{g}^t)$. Note that on $i \in S_t$, we have $t < \tau_i^\Delta$ and therefore $\gamma_i^t(\boldsymbol{w}^t) = 1$ and $\mathcal{S}_t(\boldsymbol{g}^t) = \mathcal{S}_t(\widetilde{\boldsymbol{v}}^t)$. We therefore have

$$\|\mathcal{S}_{t+1}(\overline{\boldsymbol{w}}^{t+1})\|_2^2 = 1 + \eta_1^2\|\mathcal{S}_t(\widetilde{\boldsymbol{v}}^t)\|_2^2\,, \tag{27}$$

where we used that $\|\mathcal{S}_t(\boldsymbol{w}^t)\|_2^2 = 1$ and $\langle \mathcal{S}_t(\boldsymbol{w}^t), \mathcal{S}_t(\widetilde{\boldsymbol{v}}^t)\rangle = 0$ by definition of the spherical gradient. Furthermore, $\eta_1^2\|\mathcal{S}_t(\widetilde{\boldsymbol{v}}^t)\|_2^2 \le \eta_1^2\|\boldsymbol{g}^t\|_2^2 \le d^{\ 2}\|\boldsymbol{g}^t\|_2^2 \le d^{\ 1}\|\boldsymbol{g}^t\|_{\boldsymbol{1}}^2 \le Cd^{\ 1}\log(d)^C \le 1/4$ for $t < \tau_0$. Therefore, there exists a constant $C > 0$ such that bound (25) holds.

In the case $S_{t+1} \neq S_t$, we have $|S_t \setminus S_{t+1}| \le P$ for $t < \tau^+$ and the coordinates that are removed at time $t+1$ satisfy $w_i^t + \eta_1 g_i^t \le r + C_0 d^{\ 1/2}\log(d)^{C_0} \le 2r$. Hence

$$-4Pr^2 + \|\mathcal{S}_t(\overline{\boldsymbol{w}}^{t+1})\|_2^2 \le \|\mathcal{S}_{t+1}(\overline{\boldsymbol{w}}^{t+1})\|_2^2 \le 4Pr^2 + \|\mathcal{S}_t(\overline{\boldsymbol{w}}^{t+1})\|_2^2 .$$

We can then use Eq. (27) and that $\eta_1^2\|\mathcal{S}_t(\widetilde{\boldsymbol{v}}^t)\|_2^2 \le Pr^2$ to derive Eq. (26). ■

Let us decompose the different contributions to the dynamics. We define $\boldsymbol{m}^t = \boldsymbol{g}^t - \overline{\boldsymbol{g}}^t$ the martingale updates. Let us bound the change of a coordinate after one update. For $t < \tau^0 \wedge \tau^+ \wedge \tau$, if $S_{t+1} = S_t$, then by Eq. (25), we have for $i \in S_{t+1}$

$$\begin{aligned} w_i^{t+1} &= \frac{w_i^t + \eta_1 g_i^t}{\|\mathcal{S}_{t+1}(\boldsymbol{w}^t + \eta_1 \boldsymbol{g}^t)\|_2} \ge w_i^t + \eta_1 g_i^t - C\eta_1^2\|\boldsymbol{g}^t\|_2^2|w_i^t| - C\eta_1^3\|\boldsymbol{g}^t\|_2^2|g_i^t| , \\ w_i^{t+1} &\le w_i^t + \eta_1 g_i^t + C\eta_1^2\|\boldsymbol{g}^t\|_2^2|w_i^t| + C\eta_1^3\|\boldsymbol{g}^t\|_2^2|g_i^t| . \end{aligned} \tag{28}$$

(Note that for $t < \tau$, we have $\mathrm{sign}(w_i^{t+1}) = \mathrm{sign}(w_i^t) = 1$.) For $t < \tau^0$, we have $\|\boldsymbol{g}^t\|_{\boldsymbol{1}} \le C|a^0|\log(d)$ and $|w_i^t|/|w_i^{t+1}| \le C$ because $\eta_1 \le 1/d$. Hence, we can rearrange Eqs. (28) and obtain

$$\begin{aligned} \big(1 + C|a^0|^2\eta_1^2 d\log(d)^C\big) w_i^{t+1} &\ge w_i^t + \eta_1 g_i^t - C\eta_1^3|a^0|^3 d\log(d)^C , \\ \big(1 - C|a^0|^2\eta_1^2 d\log(d)^C\big) w_i^{t+1} &\le w_i^t + \eta_1 g_i^t + C\eta_1^3|a^0|^3 d\log(d)^C , \end{aligned} \tag{29}$$

On the other hand, if $S_{t+1} \neq S_t$, by Eq. (26), we have for $i \in S_{t+1}$,

$$\begin{aligned} \frac{w_i^{t+1}}{1 - Cr^2} &\ge w_i^t + \eta_1 g_i^t \ge w_i^t + \eta_1 g_i^t - C\eta_1^2\|\boldsymbol{g}^t\|_2^2|w_i^t| - C\eta_1^3\|\boldsymbol{g}^t\|_2^2|g_i^t| , \\ \frac{w_i^{t+1}}{1 + Cr^2} &\le w_i^t + \eta_1 g_i^t + C\eta_1^2\|\boldsymbol{g}^t\|_2^2|w_i^t| + C\eta_1^3\|\boldsymbol{g}^t\|_2^2|g_i^t| C . \end{aligned}$$

Rearranging these equations, we obtain for $t < \tau^0 \wedge \tau$ and $S_{t+1} \neq S_t$ (using $|Cr^2| \le 1/2$),

$$\begin{aligned} \frac{1 + C|a^0|^2\eta_1^2 d\log(d)^C}{1 - Cr^2} \cdot w_i^{t+1} &\ge w_i^t + \eta_1 g_i^t - C\eta_1^3|a^0|^3 d\log(d)^C , \\ \frac{1 - C|a^0|^2\eta_1^2 d\log(d)^C}{1 + Cr^2} \cdot w_i^{t+1} &\le w_i^t + \eta_1 g_i^t + C\eta_1^3|a^0|^3 d\log(d)^C , \end{aligned} \tag{30}$$

On the other hand, if $i \notin S_{t+1}$, then

$$w_i^{t+1} = w_i^t + \eta_1 g_i^t .$$

Define $X_t = \#\{i \in [P] : \tau_i^r < t\}$, and

$$\underline{p}^t := \frac{(1 - C|a^0|^2\eta_1^2 d\log(d)^C)^t}{(1 + Cr^2)^{X_t}} , \qquad \overline{p}^t := \frac{(1 + C|a^0|^2\eta_1^2 d\log(d)^C)^t}{(1 - Cr^2)^{X_t}} .$$

Note $\sigma(\underline{p}^t), \sigma(\overline{p}^t) \in \mathcal{F}_{t \quad 1}$, so that $\underline{p}^t \boldsymbol{m}^t$ and $\overline{p}^t \boldsymbol{w}^t$ are still martingale updates. By induction on Eqs (29) and (30), we deduce that for $t < \tau^0 \wedge \tau^+ \wedge \tau \quad$,

$$\begin{aligned}
\underline{p}^{t \wedge \tau_i^r} w_i^t &\le w_i^0 + \eta_1 \sum_{s=0}^{t \quad 1} \underline{p}^{s \wedge \tau_i^r} \overline{g}_i^s + \eta_1 \sum_{s=0}^{t \quad 1} \underline{p}^{s \wedge \tau_i^r} m_i^s + C(t \wedge \tau_i^r) \underline{p}^{t \wedge \tau_i^r} \eta_1^3 |a^0|^3 d \log(d)^C \,, \\
\overline{p}^{t \wedge \tau_i^r} w_i^t &\ge w_i^0 + \eta_1 \sum_{s=0}^{t \quad 1} \overline{p}^{s \wedge \tau_i^r} \overline{g}_i^s + \eta_1 \sum_{s=0}^{t \quad 1} \overline{p}^{s \wedge \tau_i^r} m_i^s - C(t \wedge \tau_i^r) \overline{p}^{t \wedge \tau_i^r} \eta_1^3 |a^0|^3 d \log(d)^C \,.
\end{aligned} \tag{31}$$

Let us introduce the following quantities:

$$D_i^{t,t^0} = \sum_{s=t}^{t^0 \quad 1} \overline{g}_i^s \,, \qquad \underline{D}_i^{t,t^0} = \sum_{s=t}^{t^0 \quad 1} \underline{p}^{s \wedge \tau_i^r} \overline{g}_i^s \,, \qquad \overline{D}_i^{t,t^0} = \sum_{s=t}^{t^0 \quad 1} \overline{p}^{s \wedge \tau_i^r} \overline{g}_i^s \,,$$

and

$$M_i^{t,t^0} = \sum_{s=t}^{t^0 \quad 1} m_i^s \,, \qquad \underline{M}_i^{t,t^0} = \sum_{s=t}^{t^0 \quad 1} \underline{p}^{s \wedge \tau_i^r} m_i^s \,, \qquad \overline{M}_i^{t,t^0} = \sum_{s=t}^{t^0 \quad 1} \overline{p}^{s \wedge \tau_i^r} m_i^s \,.$$

The term D^{t,t^0}, which is the sum of population gradients, plays the role of a drift term, while the term M^{t,t^0} is a martingale and corresponds to the comparison between the stochastic and the population gradients.

We can choose C^0 a constant large enough, depending only on K, D, such that for

$$\eta_1^2 T \le \frac{1}{C^0 |a^0|^2 \log(d)^{C^0} d} \,, \tag{32}$$

we have

$$1 - \frac{1}{\log(d)} \le (1 - C|a^0|^2 \eta_1^2 d \log(d)^C)^T \le (1 + C|a^0|^2 \eta_1^2 d \log(d)^C)^T \le 1 + \frac{1}{\log(d)} \,.$$

In particular, this implies that for any $t < T \wedge \tau^+ \wedge \tau \quad$ and r constant sufficiently small,

$$\frac{1}{\overline{p}^t} \ge 1 - Cr^2 - \frac{C}{\log(d)} \,, \qquad \frac{1}{\underline{p}^t} \le 1 + Cr^2 + \frac{C}{\log(d)} \,. \tag{33}$$

Similarly, we can choose C^0 a constant large enough, depending only on K, D, such that for

$$\eta_1^3 T \le \frac{1}{C^0 |a^0|^3 \log(d)^{C^0} d^{3/2}} \,, \tag{34}$$

we have

$$\sup_{t < T \wedge \tau^+ \wedge \tau} C(t \wedge \tau_i^r) \overline{p}^{t \wedge \tau_i^r} \eta_1^3 |a^0|^3 d \log(d)^C \le \frac{1}{\sqrt{d} \log(d)} \,.$$

Hence for η_1 and T satisfying Eqs (32) and (34), we get the following bounds on the trajectory for $t < T \wedge \tau^+ \wedge \tau \quad \wedge \tau^0$: for $i \ge P+1$ or $i \in [P]$, $t \le \tau_i^r$,

$$\begin{aligned}
w_i^t &\ge \left(1 - Cr^2 - \frac{C}{\log(d)}\right) \left[(1 - \log(d) \quad ^1) w_i^0 + \eta_1 \overline{D}_i^{0,t} + \eta_1 \overline{M}_i^{0,t}\right] \,, \\
w_i^t &\le \left(1 + Cr^2 + \frac{C}{\log(d)}\right) \left[(1 + \log(d) \quad ^1) w_i^0 + \eta_1 \underline{D}_i^{0,t} + \eta_1 \underline{M}_i^{0,t}\right] \,,
\end{aligned} \tag{35}$$

while for $i \in [P]$ and $t > \tau_i^r$,

$$w_i^t = w_i^{\tau_i^r} + \eta_1 D_i^{\tau_i^r, t} + \eta_1 M_i^{\tau_i^r, t}. \tag{36}$$

We prove the following bounds on the martingale part:

Lemma 15 (Martingale part M^t) *Assume $\Delta \le 1$ and r are chosen as in Lemma 14. Fix $T \le d^D$ and $C > 0$. There exists a constant C that only depends on D, K, and C, such that if we choose*

$$\eta_1^2 T \le \frac{1}{C|a^0|^2 \log(d)^C d}, \tag{37}$$

then with probability at least $1 - d^{-C}$, we have

$$\max_{t<t^0 \le T \wedge \tau^+} \max_{i \in [d]} \left\{ |\eta_1 M_i^{t,t^0}| \vee |\eta_1 \overline{M}_i^{t,t^0}| \vee |\eta_1 \underline{M}_i^{t,t^0}| \right\} \le \frac{1}{\sqrt{d}\log(d)}. \tag{38}$$

Proof [Proof of Lemma 15] We will show the theorem for $\max_{0<t \le T \wedge \tau^+} \underline{M}_i^{0,t}$. The result for $\max_{t<t^0 \le T \wedge \tau^+} \underline{M}_i^{t,t^0}$ will follow from an union bound on all $t \le T$, which are also martingales (the proofs for $\overline{M}_i^{t,t^0}$ and M_i^{t,t^0} will follow by the same argument).

Denote $\underline{M}_i^t := \underline{M}_i^{0,t}$. We will use a truncation argument. For some $\widetilde{C}$, define for all $t \ge 1$ and $i \in [d]$,

$$U_i^t = \sum_{s=0}^{t-1} \underline{p}^s \left\{ g_i^t \mathbb{1}_{|g_i^t/a^0| < \widetilde{C}\log(d)^{\widetilde{C}}} - \mathsf{E}\left[g_i^t \mathbb{1}_{|g_i^t/a^0| < \widetilde{C}\log(d)^{\widetilde{C}}} \right] \right\},$$

so that

$$|U_i^t - \underline{M}_i^t| \le (1 - Cr^2)^P \sum_{s=0}^{t-1} \left\{ |g_i^t| \mathbb{1}_{|g_i^t/a^0| \ge \widetilde{C}\log(d)^{\widetilde{C}}} + \mathsf{E}\left[|g_i^t| \mathbb{1}_{|g_i^t/a^0| \ge \widetilde{C}\log(d)^{\widetilde{C}}} \right] \right\}.$$

For $t \le \tau^+$, we have $\|\boldsymbol{w}\|_2 \le \sqrt{P}\Delta + 1$ and we can use Lemma 16 to choose $\widetilde{C}$ that only depends on D, K, C such that

$$\mathsf{P}\big(\exists t \le T \wedge \tau^+, \exists i \in [d], |g_i^t/a^0| \ge \widetilde{C}\log(d)^{\widetilde{C}}\big) \le d^{-C}/2,$$

and for all $t \le T \wedge \tau^+$ and $i \in [d]$

$$\eta_1 (1 - Cr^2)^P \mathsf{E}\left[|g_i^t| \mathbb{1}_{|g_i^t/a^0| \ge \widetilde{C}\log(d)^{\widetilde{C}}} \right] \le d^{-D} \cdot \frac{1}{2\sqrt{d}\log(d)}.$$

Hence with probability at least $1 - d^{-C}/2$,

$$\max_{t \le T \wedge \tau^+} \max_{i \in [d]} |\eta_1 \underline{M}_i^t| \le \frac{1}{2\sqrt{d}\log(d)} + \max_{t \le T \wedge \tau^+} \max_{i \in [d]} |\eta_1 U_i^t|.$$

Let us now apply Doob's maximal inequality on U_i^t: the increments are bounded by $2|a^0|\widetilde{C}\log(d)^{\widetilde{C}}$, hence we have

$$\mathsf{P}\Big(\max_{1 \le t \le T} |\eta_1 U_i^t| \ge \varepsilon \Big) \le 2\exp\left\{ -\frac{\varepsilon^2}{C(\eta_1 a^0 \widetilde{C}\log(d)^{\widetilde{C}})^2 T} \right\}.$$

Choosing $\varepsilon = 1/(2\sqrt{d}\log(d))$ and η_1 as in Eq. (37), as well as a union bound, yields the result. ■

C.3. Proof of Theorem 8

Step 0: Bounds on the dynamics.

We consider the dynamics (18) up to time $T \wedge \overline{\tau}$ where $\overline{\tau} := \tau^0 \wedge \tau^+ \wedge \tau_*$. We assume T and η_1 satisfy conditions (32), (34) and (37). In particular, with probability at least $1 - d^{-C}$, the dynamics of w_i^t satisfies the bounds in Eqs (35) and (36), with $M_i^{t,t^0}, \overline{M}_i^{t,t^0}, \underline{M}_i^{t,t^0}$ satisfying the bounds (38). In the rest of the proof, we show that on this high probability event, we can choose η_1 and T such that $T < \overline{\tau}$ and Theorem 8.(a), (b) and (c) are satisfied.

Step 1: Controlling the coordinates $i \in [P]$ at the end of the dynamics.

Let us first show that as soon as $t > \tau_i^\Delta$, then w_i^t stays close to Δ. Note that we can choose $C > 0$ constant large enough independent of d, such that if $w_i^t \le \Delta - |a^0| C\eta_1 \log(d)^C$, then by (24)

$$\begin{aligned} \mathsf{E}_{\boldsymbol{x}}[g_i^t] = \mathsf{E}_{\boldsymbol{x}}[\gamma_i^t \widetilde{v}_i^t] &\ge \mathsf{E}_{\boldsymbol{x}}[\widetilde{v}_i^t] - \mathsf{E}_{\boldsymbol{x}}[|\widetilde{v}_i^t|^2]^{1/2} \mathsf{P}\big\{|v_i^t| \ge |a^0| C\eta_1 \log(d)^C\big\} \\ &\ge Cd^{-(D-1)/2} - Cd^{-D} > 0\,. \end{aligned} \tag{39}$$

Hence, for any $\overline{\tau} > t > \tau_i^\Delta$, consider $t^0 = \sup\{t^0 \le t : w_i^{t^0} \ge \Delta - |a^0| C\eta_1 \log(d)^C\}$ (in particular, $t^0 \ge \tau_i^\Delta + 1$). From Eq. (36) and by Lemma 15, we have

$$w_i^t = w_i^{t^0+1} + \eta_1 D_i^{t^0+1,t} + \eta_1 M_i^{t^0+1,t} \ge \Delta - |a^0| C\eta_1 \log(d)^C - \frac{1}{\sqrt{d \log d}} \ge \Delta - \frac{2}{\sqrt{d \log(d)}}\,,$$

where we used that $w_i^s < \Delta - |a^0| C\eta_1 \log(d)^C$ for $t^0 + 1 \le s < t$ and therefore by Eq. (39), we have $D_i^{t^0+1,t} \ge 0$. We deduce that

$$\inf_{\tau_i^\Delta < t < \overline{\tau} \wedge T} w_i^t \ge \Delta - \frac{2}{\sqrt{d \log(d)}}\,. \tag{40}$$

Similarly, we show that for any $t \ge \tau_i^r + 1$, we have $w_i^t \ge r/2$. Indeed, for any $\tau_i^r + 1 \le t \le \tau_i^\Delta \wedge \overline{\tau} \wedge T$, we have

$$w_i^t = w_i^{\tau_i^r+1} + \eta_1 D_i^{\tau_i^r+1,t} + \eta_1 M_i^{\tau_i^r+1,t} \ge r - \frac{1}{\sqrt{d \log(d)}}\,,$$

where we used that $w_i^{\tau_i^r+1} \ge r$ by definition of τ_i^r, and $\overline{g}_i^t \ge 0$ for all $s \le \tau_i^\Delta \wedge \overline{\tau} \wedge T$. We deduce that

$$\inf_{\overline{\tau} \wedge T > t > \tau_i^r} w_i^t \ge r - \frac{1}{\sqrt{d \log(d)}} \ge \frac{r}{2}\,. \tag{41}$$

Step 2: Bounding the growth of w_i^t for $i \in [P]$.

Define $\alpha_t = \min\{w_i^t : i \in S_t \cap [P]\}$ (i.e., the minimum of w_i^t that have $\tau_i^r \ge t$). Note that $w_i^t \ge r/2$ for $i \notin S_t$ by Eq. (41) and therefore $w_i^t \ge \alpha_t/2$. By Eq. (33) and Lemma 13, we have for $s \le \tau_i^r \wedge \overline{\tau} \wedge T$,

$$\underline{p}^s \overline{g}_i^s \ge Ca^0 \frac{\chi_*(\boldsymbol{w}^t)}{w_i^t} \mu_D(\sigma) - |a^0| \tilde{\kappa} \ge Ca^0 \mu_D(\sigma) \alpha_t^{D-1} - |a^0| \tilde{\kappa}\,.$$

Combining this lower bound with Eq. (35) and the bound on the martingale in Lemma 15, we get that for all $t \leq \tau_i^r \wedge \overline{\tau} \wedge T$

$$\begin{aligned}\alpha_t &\geq \frac{1 - Cr^2}{\sqrt{d}} - \eta_1 t |a^0| \tilde{\kappa} + C\eta_1 a^0 \mu_D(\sigma) \sum_{s=0}^{t} \alpha_t^{D \quad 1} \\ &\geq \frac{1}{2\sqrt{d}} + C\eta_1 a^0 \mu_D(\sigma) \sum_{s=0}^{t} \alpha_t^{D \quad 1},\end{aligned} \tag{42}$$

where we have used that r is sufficiently small, and $\eta_1 t |a^0| \tilde{\kappa} \leq \tilde{\kappa} / (C^0 \eta_1 |a^0| \log(d)^{C^0} d) \leq 1/(4\sqrt{d})$ for all $t \leq T$ since we take κ sufficiently small. We can use the bound on this sequence derived in Lemma 17: for $D > 2$,

$$\alpha_t \geq r \wedge \left\{ \frac{1}{\left((4d)^{D/2 \quad 1} - C\eta_1 a^0 \mu_D(\sigma) t\right)^{\frac{1}{k \quad 2}}} \right\},$$

and we deduce that we must have

$$\tau^r \wedge \overline{\tau} \leq \frac{C d^{D/2 \quad 1}}{\eta_1 a^0 \mu_D(\sigma)}.$$

For $D = 2$, we use that

$$\alpha_t \geq \frac{1}{2\sqrt{d}} \left(1 + C\eta_1 a^0 \mu_D(\sigma)\right)^t,$$

and deduce that

$$\tau^r \wedge \overline{\tau} \leq \frac{C \log(d)}{\eta_1 a^0 \mu_D(\sigma)}.$$

Similarly, consider $\alpha_t = \min\{w_i^t : i \in [P], \tau_i^\Delta \geq t\}$ for all $\tau^r + 1 \leq t \leq \tau^\Delta \wedge \overline{\tau}$. By Eq. (40), we have $w_i^t \geq \Delta/2$ for $t > \tau_i^\Delta$. Hence by Eq. (36), we get

$$\alpha_t \geq \frac{r}{2} + C\eta_1 a^0 \mu_D(\sigma) \sum_{s=0}^{t} \alpha_t^{D \quad 1},$$

and we deduce that if $\tau^\Delta < \overline{\tau}$ then

$$\tau^\Delta - \tau^r \leq \frac{C}{\eta_1 a^0 \mu_D(\sigma)}.$$

Combining the above bounds, we deduce that

$$\begin{aligned}&\text{for } D = 2: \qquad \tau^\Delta \wedge \overline{\tau} \leq \frac{C \log(d)}{\eta_1 a^0 \mu_D(\sigma)}, \\ &\text{for } D > 2: \qquad \tau^\Delta \wedge \overline{\tau} \leq \frac{C d^{D/2 \quad 1}}{\eta_1 a^0 \mu_D(\sigma)},\end{aligned} \tag{43}$$

On the other hand, consider $\beta_t = \max\{w_i^t : i \in [P]\}$ and let us lower bound the time $\bar{t} = \inf\{t : \beta_t \geq 2/\sqrt{d}\}$. For $t \leq \overline{\tau} \wedge \bar{t}$, we have $S_t = [d]$ and $\frac{1}{\underline{p}^t} \leq 1 + \frac{C}{\log(d)}$. Hence by Eq. (35),

$$\beta_t \leq (1 + C/\sqrt{\log(d)})\frac{2}{\sqrt{d}} + C\eta_1 a^0 \mu_D(\sigma) \sum_{s=0}^{t} \beta_s^{D-1},$$

and therefore by Lemma 17, we get for $D > 2$,

$$\beta_t \leq \frac{1}{\left((d/2(1 + C/\sqrt{\log d}))^{D/2-1} - C\eta_1 a^0 \mu_D(\sigma) t\right)^{\frac{1}{D-2}}}. \tag{44}$$

We deduce that

$$\bar{t} \geq \overline{\tau} \wedge \frac{d^{D/2-1}}{C\eta_1 a^0 \mu_D(\sigma)}. \tag{45}$$

For $D = 2$, we have by Lemma 17,

$$\beta_t \leq \frac{3}{\sqrt{d}}\big(1 + C\eta_1 a^0 \mu_D(\sigma)\big)^t, \tag{46}$$

and therefore

$$\bar{t} \geq \overline{\tau} \wedge \frac{1}{C\eta_1 a^0 \mu_D(\sigma)}. \tag{47}$$

Step 3: Bounding the coordinates $P + 1 \leq i \leq d$.

From Eq. (35) and Lemma 13, we have for all $i \geq P + 1$ and $t < \overline{\tau} \wedge T$,

$$w_i^t \geq (1 - Cr^2)\frac{1}{\sqrt{d}} - C|\eta_1 \overline{D}_i^{0,t\wedge(\tau^r+1)}| - Ct\eta_1 |a_i^0|\tilde{\kappa},$$
$$w_i^t \leq (1 + Cr^2)\frac{1}{\sqrt{d}} + C|\eta_1 \underline{D}_i^{0,t\wedge(\tau^r+1)}| + Ct\eta_1 |a_i^0|\tilde{\kappa}.$$

We have

$$|\eta_1 \overline{D}_i^{0,t\wedge(\tau^r+1)}| \vee |\eta_1 \underline{D}_i^{0,t\wedge(\tau^r+1)}| \leq C\eta_1 a^0 \mu_D(\sigma) \sum_{s=0}^{t\wedge(\tau^r+1)-1} w_i^s \chi(\boldsymbol{w}^s) + Ct\eta_1 |a_i^0|\tilde{\kappa}. \tag{48}$$

Consider $j \in [P]$ such that $\tau_j^r = \tau^r$. Then by Eq. (35), we have, for any $t < (\tau^r + 1) \wedge \overline{\tau} \wedge T$, that

$$2r \geq w_j^{t+1} - (1 - Cr^2)\frac{1}{\sqrt{d}} \geq C\eta_1 a^0 \mu_D(\sigma) \sum_{s=0}^{t\wedge\tau_j^r} \frac{\chi(\boldsymbol{w}^s)}{w_j^s}.$$

Hence we deduce that

$$\eta_1 a^0 \mu_D(\sigma) \sum_{s=0}^{t\wedge\tau_j^r} \frac{\chi(\boldsymbol{w}^s)}{w_j^s} \leq Cr. \tag{49}$$

Using that $w_j^t \le Cr$ for $t \le \tau_j^r$ and $w_i^s \le 3/(2\sqrt{d})$ for $s < \overline{\tau}$ in Eq. (48), we get that, for any $t < \overline{\tau} \wedge T$,

$$|\eta_1 \overline{D}_i^{0,t\wedge(\tau^r+1)}| \vee |\eta_1 \underline{D}_i^{0,t\wedge(\tau^r+1)}| \le Ct\eta_1 |a^0|\tilde{\kappa} + \frac{C}{\sqrt{d}} r\eta_1 a^0 \mu_D(\sigma) \sum_{s=0}^{t\wedge(\tau^r+1)} \frac{\chi(\boldsymbol{w}^s)}{w_j^s}$$
$$\le \frac{Cr^2}{\sqrt{d}} .$$

We deduce that for all $P+1 \le i \le d$ and $t < \overline{\tau}$ and $t+1 < \tau^0$

$$\frac{1-Cr^2}{\sqrt{d}} \le w_i^{t+1} \le \frac{1+Cr^2}{\sqrt{d}} , \tag{50}$$

and therefore taking r sufficiently small, $\tau^+ \wedge \tau \ge \tau^0$.

Step 4: Proof of Theorem 8.(b) and (c).

Choose $\overline{T}_1 := T$ and η_1 that satisfy Eqs. (32), (34), (37) and (43). We have with probability at least $1 - Cd^{-C}$ by Lemma 12 and Lemma 15 that $\tau^0 > \tau^+ \wedge \tau \wedge T$. And Eqs. (50) and (42) imply that $\tau^+ \wedge \tau \wedge \tau^0 > \overline{T}_1$. Furthermore, by Eq. (43), we have $\tau^\Delta < \overline{T}_1$ which implies Theorem 8.(b) by Eq. (40). Theorem 8.(c) follows from Eq. (50).

Step 5: Upper bound for all neurons with early stopping.

Theorem 8.(a) follows from Eqs. (45) and (47) for neurons with initialization satisfying

$$a_j^0 \mu_D(\sigma)(w_{j,1}^0)^{k_1} \cdots (w_{j,P}^0)^{k_P} > 0 .$$

For neurons that do not satisfy this condition, the analysis in Section C.2 still holds and we get bounds on the dynamics similar to the ones in Eq. (35), with the difference that $a^0 \mu_D(\sigma) < 0$, and therefore the drift has a negative contribution to the dynamics, and τ^+, τ are now defined on all coordinates instead of only $i = P+1, \dots, d$.

We can upper bound the drift contribution using that $\beta_t = \max_{i \in [P]} w_i^t$ satisfy for $t < \overline{\tau}$

$$\beta_t \le (1 + C/\sqrt{\log(d)}) \frac{1}{\sqrt{d}} + C\eta_1 |a^0 \mu_D(\sigma)| \sum_{s=0}^{t} \beta_s^{D-1} , \tag{51}$$

and therefore, taking the same bounds (44) and (46), we get for $t \le \overline{\tau} \wedge \overline{T}_1/(C\log(d)^C)$ for C a constant sufficiently large that

$$\beta^t \le \frac{1}{\sqrt{d}} (1 + C^0/\sqrt{\log(d)}) . \tag{52}$$

Furthermore, denoting $\alpha_t = \min_{i \in [P]} w_i^t$, we have

$$\alpha_t \ge (1 - C/\sqrt{\log(d)}) \frac{1}{\sqrt{d}} - C\eta_1 |a^0 \mu_D(\sigma)| \sum_{s=0}^{t} \beta_s^{D-1} .$$

Using the analysis of Lemma 17, we get that the drift has the same upper bound as β_t in Eq. (52),

$$(1 + C/\sqrt{\log(d)}) \frac{1}{\sqrt{d}} + C\eta_1 |a^0 \mu_D(\sigma)| \sum_{s=0}^{t} \beta_s^{D-1} \le \frac{1}{\sqrt{d}} (1 + C^0/\sqrt{\log(d)})$$

for $t \le \overline{\tau} \wedge \overline{T}_1/(C\log(d)^C)$ and therefore

$$\alpha_t \ge \frac{1}{\sqrt{d}}(1 - C/\sqrt{\log(d)})\,.$$

The bound on coordinates $i > P$ follows similarly to step 3. In particular, we deduce that for $t < \overline{\tau} \wedge \overline{T}_1/(C\log(d)^C)$ and $t+1 < \tau^0$, we have for all $i \in [d]$

$$(1 - C/\sqrt{\log(d)})\frac{1}{\sqrt{d}} \le w_i^{t+1} \le (1 + C/\sqrt{\log(d)})\frac{1}{\sqrt{d}}\,,$$

and therefore $\tau^+ \wedge \tau \quad > \tau^0 \wedge \overline{T}_1/(C\log(d)^C)$, which concludes the proof of Theorem 8.(a).

C.4. Technical lemmas

Lemma 16 (Tail bounds on functions of Gaussians) *Assume that $\|\boldsymbol{w}^t\|_2 \le 1 + \sqrt{D}$. Then there exist constants c, C that only depend on D and K such that*

$$\begin{aligned}
&\mathsf{P}(|y^t| \ge z + c) \le \exp\big\{ - Cz^{2/D}\big\},\\
&\mathsf{P}_{\boldsymbol{w}^t}\left(|y^t\sigma(\langle \boldsymbol{w}^t, \boldsymbol{x}^t\rangle)| \ge z + c\right) \le \exp\big\{ - Cz^{2/D}\big\}\,,\\
&\mathsf{P}_{\boldsymbol{w}^t}\big(|g_i^t/a^0| \ge z + c\big) \le \exp\big\{ - Cz^{2/(D+1)}\big\}\,,\\
&\mathsf{P}_{\boldsymbol{w}^t}\big(\|\boldsymbol{g}^t/a^0\|_2 \ge \sqrt{d}(z + c)\big) \le \exp\big\{ - Cz^{2/(D+1)}\big\}\,,\\
&\mathsf{P}_{\boldsymbol{w}^t}\big(\|\boldsymbol{g}^t/a^0\|_2^2 \cdot |g_i^t/a^0| \ge d(z + c)\big) \le \exp\big\{ - Cz^{2/(3D+3)}\big\}\,,
\end{aligned} \tag{53}$$

where $\mathsf{P}_{\boldsymbol{w}^t}(\cdot) := \mathsf{P}(\cdot|\boldsymbol{w}^t)$ denotes the conditional probability. Furthermore, for any $q \ge 2$, there exists a constant C_q that only depends on q, D, K such that

$$\mathsf{E}_{\boldsymbol{w}^t}\left[\left\|\boldsymbol{g}^t/a^0\right\|_q^q\right]^{1/q} \le C_q\sqrt{d}\,, \tag{54}$$

where $\mathsf{E}_{\boldsymbol{w}^t}[\cdot] := \mathsf{E}[\cdot|\boldsymbol{w}^t]$ denotes the conditional expectation.

Proof [Proof of Lemma 16] Recall that for polynomials $f : \mathsf{R}^d \to \mathsf{R}$ of degree D on Gaussian variables, we have the hypercontractivity inequality $\|f\|_{L^q} \le (q-1)^{D/2}\|f\|_{L^2}$ for any $q \ge 2$. Hence, there exist constants C, C^0 that only depend on D such that

$$\mathsf{P}\big(|f(\boldsymbol{x}) - \mathsf{E}_{\boldsymbol{x}}[f(\boldsymbol{x})]| \ge z\sqrt{\mathrm{Var}_{\boldsymbol{x}}(f)}\big) \le C^0\exp\big\{ - Cz^{2/D}\big\}\,. \tag{55}$$

Recall that $y = \mathrm{He}_{\boldsymbol{k}}(\boldsymbol{x}) + \varepsilon$, where $\mathrm{He}_{\boldsymbol{k}}$ is a degree-D multivariate Hermite polynomial and ε is K-subgaussian. Further, recall that we assumed $\|\sigma\|_{\mathsf{1}}, \|\sigma^0\|_{\mathsf{1}} \le K$. Applying Eq. (55), there exist constants C, c that only depend on D and K such that for any $\boldsymbol{w} \in \mathsf{R}^d$,

$$\mathsf{P}(|y| \ge z + c) \le \exp\big\{ - Cz^{2/D}\big\}\,, \qquad \mathsf{P}(|y\sigma(\langle \boldsymbol{w}, \boldsymbol{x}\rangle)| \ge z + c) \le \exp\big\{ - Cz^{2/D}\big\}\,,$$

where we used that $\mathsf{E}[|y\sigma(\langle \boldsymbol{w}, \boldsymbol{x}\rangle)|] \le K\mathsf{E}[|y|^2]^{1/2} \le c$. Following a similar reasoning, we get for any $\|\boldsymbol{w}\|_2^2 \le 1 + \sqrt{D}$ and $i \in [d]$,

$$\begin{aligned}
&\mathsf{P}(|yx_i\sigma^0(\langle \boldsymbol{w}, \boldsymbol{x}\rangle)| \ge z + c) \le \exp\big\{ - Cz^{2/(D+1)}\big\}\,,\\
&\mathsf{P}(|yw_i\langle \boldsymbol{w}, \boldsymbol{x}\rangle\sigma^0(\langle \boldsymbol{w}, \boldsymbol{x}\rangle)| \ge z + c) \le \exp\big\{ - Cz^{2/(D+1)}\big\}\,.
\end{aligned}$$

Recall that $|g_i^t/a^0| \le |\gamma_i^t|\big(|yx_i\sigma^0(\langle \boldsymbol{w}, \boldsymbol{x}\rangle)| + |yw_i\langle \boldsymbol{w}, \boldsymbol{x}\rangle\sigma^0(\langle \boldsymbol{w}, \boldsymbol{x}\rangle)|\big)$. Conditioning on $\boldsymbol{w}^t$ and assuming that $\|\boldsymbol{w}^t\|_2 \le 1 + \sqrt{D}$, we obtain

$$\begin{aligned}
&\mathsf{P}_{\boldsymbol{w}^t}\big(|g_i^t/a^0| \ge z + c\big) \le \exp\big\{-Cz^{2/(D+1)}\big\}\,,\\
&\mathsf{P}_{\boldsymbol{w}^t}\big(\|\boldsymbol{g}^t/a^0\|_2 \ge \sqrt{d}(z+c)\big) \le \mathsf{P}_{\boldsymbol{w}^t}\big(\|\boldsymbol{g}^t/a^0\|_2^2 \ge d(z^2+c)\big)\exp\big\{-Cz^{2/(D+1)}\big\}\,,\\
&\mathsf{P}_{\boldsymbol{w}^t}\big(\|\boldsymbol{g}^t/a^0\|_2^2|g_i^t/a^0| \ge d(z+c)\big) \le d\exp\big\{-Cz^{2/(3D+3)}\big\}\,.
\end{aligned}$$

Furthermore, again assuming that $\|\boldsymbol{w}^t\|_2 \le 1 + \sqrt{D}$, we have for any $q \ge 2$,

$$\mathsf{E}_{\boldsymbol{w}^t}\big[\|\boldsymbol{g}^t/a^0\|_q^q\big]^{1/q} \le K\mathsf{E}_{\boldsymbol{w}^t}\big[\|y^t\boldsymbol{x}^t\|_q^q\big]^{1/q} + K\mathsf{E}_{\boldsymbol{w}^t}\big[\|y^t\boldsymbol{w}^t\langle \boldsymbol{w}^t, \boldsymbol{x}^t\rangle\|_q^q\big]^{1/q} \le C_q\sqrt{d}\,,$$

which concludes the proof. ■

The following lemma provides simple upper and lower bounds on sequences satisfying some geometric bound on their evolution. The upper bound can be seen as a discrete version of Bihari–LaSalle inequality. This upper bound was proven in (Arous et al., 2021, Appendix C), and we modify their proof to obtain a lower bound.

Lemma 17 (Bounds on sequences) *Let $k \ge 2$ be an integer and $a_0, a_1, b_0, b_1 > 0$ be four positive constants with $a_0 \le b_0$ and $a_1 \le b_1$. Consider a sequence $(u_t)_{t\in\mathsf{N}}$ that satisfy for all $t \in \mathsf{N}$,*

$$a_0 + a_1\sum_{s=0}^{t-1} u_s^{k-1} \le u_t \le b_0 + b_1\sum_{s=0}^{t-1} u_s^{k-1}\,.$$

If $k = 2$, then for any $t \in \mathsf{N}$,

$$a_0(1+a_1)^t \le u_t \le b_0(1+b_1)^t\,.$$

If $k > 2$, then for any $\Delta > 0$ and any $t \in \mathsf{N}$,

$$\Delta \wedge \left\{\frac{1}{\Big(a_0^{(k-2)} - \frac{k-2}{(1+a_1\Delta^{k-2})^{k-1}}a_1 t\Big)^{\frac{1}{k-2}}}\right\} \le u_t \le \frac{1}{\Big(b_0^{(k-2)} - (k-2)b_1 t\Big)^{\frac{1}{k-2}}}\,.$$

Proof [Proof of Lemma 17] Note that by induction, we have $w_t \le u_t \le v_t$ for any $t \ge 0$ where

$$v_t = b_0 + b_1\sum_{s=0}^{t-1} v_s^{k-1}\,, \qquad w_t = a_0 + a_1\sum_{s=0}^{t-1} w_s^{k-1}\,.$$

For $k = 2$, it is straightforward to get $v_t = v_{t-1}(1+b_1) = b_0(1+b_1)^t$ and $w_t = a_0(1+a_1)^t$.

For $k > 2$, we consider the upper bound on v_t. First, notice that

$$b_1 = \frac{v_t - v_{t-1}}{v_{t-1}^{k-1}} \ge \int_{v_{t-1}}^{v_t}\frac{1}{x^{k-1}}\mathrm{d}x = \frac{1}{k-2}\left[\frac{1}{v_{t-1}^{k-2}} - \frac{1}{v_t^{k-2}}\right]\,.$$

Hence, rearranging the terms, we get for any t,

$$\frac{1}{v_t^{k\ \ 2}} \geq \frac{1}{v_{t\ \ 1}^{k\ \ 2}} - (k-2)b_1 \geq \frac{1}{v_0^{k\ \ 2}} - (k-2)b_1 t\,.$$

We deduce that

$$v_t \leq \frac{1}{\left(b_0{}^{(k\ \ 2)} - (k-2)b_1 t\right)^{\frac{1}{k\ \ 2}}}\,.$$

Let us now lower bound w_t. We have

$$\begin{aligned} a_1 = \frac{w_t - w_{t\ \ 1}}{w_{t\ \ 1}^{k\ \ 1}} &= \int_{w_{t\ \ 1}}^{w_t} \frac{1}{x^{k\ \ 1}}\mathrm{d}x + \int_{w_{t\ \ 1}}^{w_t} \frac{x^{k\ \ 1} - w_{t\ \ 1}^{k\ \ 1}}{w_{t\ \ 1}^{k\ \ 1} x^{k\ \ 1}}\mathrm{d}x \\ &\leq \frac{w_t^{k\ \ 1}}{w_{t\ \ 1}^{k\ \ 1}} \int_{w_{t\ \ 1}}^{w_t} \frac{1}{x^{k\ \ 1}}\mathrm{d}x \\ &= \frac{(1 + a_1 w_{t\ \ 1}^{k\ \ 2})^{k\ \ 1}}{k-2} \left[\frac{1}{w_{t\ \ 1}^{k\ \ 2}} - \frac{1}{w_t^{k\ \ 2}}\right]. \end{aligned}$$

Hence, as long as $w_t \leq \Delta$, we get

$$\frac{1}{w_t^{k\ \ 2}} \leq \frac{1}{w_{t\ \ 1}^{k\ \ 2}} - \frac{k-2}{(1 + a_1 \Delta^{k\ \ 2})^{k\ \ 1}} a_1 \leq \frac{1}{v_0^{k\ \ 2}} - \frac{k-2}{(1 + a_1 \Delta^{k\ \ 2})^{k\ \ 1}} a_1 t\,,$$

and therefore

$$w_t \geq \frac{1}{\left(a_0{}^{(k\ \ 2)} - \frac{k\ \ 2}{(1+a_1\Delta^{k\ \ 2})^{k\ \ 1}} a_1 t\right)^{\frac{1}{k\ \ 2}}}\,.$$

■

Appendix D. Proof of Theorem 10: sequential alignment to the support

In this appendix, we consider the sequential alignment to the support in Section 3.3. The proofs will follow from a similar argument as in the single monomial case. However, the dynamics will be now split in L phases corresponding to the alignment to each of the L monomials.

Recall that throughout the proofs, we will denote for simplicity $C, c > 0$ generic constants that only depend on D and K (note that all the other constants $P_l, D_j, \overline{D}_l \leq D$). The values of these constants are allowed to change from line to line or within the same line.

D.1. Proof of Theorem 10: alignment to the full support

We will use notations and results from Appendix C and outline the main difference with the proof of Theorem 8. We can again reduce the problem to tracking one neuron, and we assume without loss of generality that $w_1^0 = \ldots = w_d^0 = 1/\sqrt{d}$ and $a^0 \mu_{\overline{D}_l}(\sigma) > 0$ for all $l \in [L]$.

Let us introduce the following new stopping times on the dynamics: for $l \in [L]$,

$$\tau^{r,l} = \sup_{i \in [P_l]} \tau_i^r \,, \qquad \tau^{\Delta,l} = \sup_{i \in [P_l]} \tau_i^\Delta \,.$$

The population gradients for $t \leq \tau_i^\Delta \wedge \tau_0$ are now given by:

Lemma 18 *Denote $\chi_{\cdot,l}(\boldsymbol{w}^t) = \prod_{j \in [P_l]} (w_j^t)^{k_j}$ for $j \in [L]$. For $i \in [P_l] \setminus [P_{l-1}]$ and $t < \tau_i^\Delta$, the population gradient is given by: if $t \leq \tau_i^r$ (i.e., $i \in S_t$),*

$$\begin{aligned} \overline{g}_i^t = a^0 \sum_{l^0 \geq l} \frac{\chi_{\cdot,l^0}(\boldsymbol{w}^t)}{w_i^t} \left(k_i - (w_i^t)^2 \sum_{s \in S_t \setminus [P_{l^0}]} k_s \right) \mathsf{E}_G\Big[\sigma^{(\overline{D}_{l^0})}(\|\boldsymbol{w}^t\|_2 G)\Big] \\ - a^0 w_i^t \sum_{l^0 < l} \chi_{\cdot,l^0}(\boldsymbol{w}^t) \left(\sum_{s \in S_t \setminus [P_{l^0}]} k_s \right) \mathsf{E}_G\Big[\sigma^{(\overline{D}_{l^0})}(\|\boldsymbol{w}^t\|_2 G)\Big] + O(|a^0|\tilde{\kappa}) \,, \end{aligned} \tag{56}$$

while if $t > \tau_i^r$ (i.e., $i \notin S_t$)

$$\begin{aligned} \overline{g}_i^t = a^0 \sum_{l^0 \geq l} \frac{\chi_{\cdot,l^0}(\boldsymbol{w}^t)}{w_i^t} \Big(k_i \mathsf{E}_G\Big[\sigma^{(\overline{D}_{l^0})}(\|\boldsymbol{w}^t\|_2 G)\Big] + (w_i^t)^2 \mathsf{E}_G\Big[\sigma^{(\overline{D}_{l^0}+2)}(\|\boldsymbol{w}^t\|_2 G)\Big] \Big) \\ + a^0 \sum_{l^0 < l} w_i^t \chi_{\cdot,l^0}(\boldsymbol{w}^t) \mathsf{E}_G\Big[\sigma^{(\overline{D}_{l^0}+2)}(\|\boldsymbol{w}^t\|_2 G)\Big] + O(|a^0|\tilde{\kappa}) \,. \end{aligned} \tag{57}$$

For $i > P$ and $t < \tau^+ \wedge \tau_0$,

$$\overline{g}_i^t = -a^0 \sum_{j \in [L]} w_i^t \chi_{\cdot}(\boldsymbol{w}^t) \left(\sum_{s \in S_t \setminus [P_l]} k_s \right) \mathsf{E}_G\Big[\sigma^{(\overline{D}_l)}(\|\boldsymbol{w}^t\|_2 G)\Big] + O(|a^0|\tilde{\kappa}) \,. \tag{58}$$

Proof [Proof of Lemma 18] The proof follows from Lemma 13 applied to a sum of monomials. ■

Again, by Assumption 7, we can choose Δ small enough and depending only on K and $\overline{D}$ such that Eqs (22) and (23) are satisfied (with D replaced by $\overline{D}$). We can further chose Δ small enough

(only depending on K and $\overline{D}$ such that there exists constants C, c such that for all $t < \tau_i^{\Delta} \wedge \tau^{+} \wedge \tau$, if $i \in [P_l] \setminus [P_{l-1}]$, then if $t > \tau^{r,l-1}$,

$$0 < ca^0 \frac{\chi_{\cdot,l}(\boldsymbol{w}^t)}{w_i^t} \le \overline{g}_i^t \le Ca^0 \frac{\chi_{\cdot,l}(\boldsymbol{w}^t)}{w_i^t} \,,$$

and if $\tau^{r,l^0-1} < t \le \tau^{r,l^0}$ for $l^0 \le l-1$,

$$ca^0 \frac{\chi_{\cdot,l}(\boldsymbol{w}^t)}{w_i^t} - Ca^0 w_i^t \chi_{\cdot,l^0}(\boldsymbol{w}^t) \le \overline{g}_i^t \le Ca^0 \frac{\chi_{\cdot,l}(\boldsymbol{w}^t)}{w_i^t}(\sigma) - ca^0 w_i^t \chi_{\cdot,l^0}(\boldsymbol{w}^t) \,,$$

while for $i \ge P+1$ and $\tau^{r,l-1} < t \le \tau^{r,l}$,

$$|\overline{g}_i^t| \le Ca^0 w_i^t \chi_{\cdot,l}(\boldsymbol{w}^t) \,,$$

and $\overline{g}_i^t = 0$ for $t > \tau^{r,L}$.

Proof [Proof of Theorem 10] **Step 0: Bounds on the dynamics.**

We consider the dynamics up to time $T \wedge \overline{\tau}$ where $\overline{\tau} := \tau^0 \wedge \tau^{+} \wedge \tau$. We again assume that T and η_1 satisfy conditions (32), (34) and (37), so that the dynamics of w_i^t satisfies the bounds in Eqs (35) and (36), with $M_i^{t,t^0}, \overline{M}_i^{t,t^0}, \underline{M}_i^{t,t^0}$ satisfying the bounds (38), with probability at least $1 - d^{-C}$. The following steps will follow closely the proof of Theorem 8.

Step 1: Controlling the coordinates $i \in [P_l] \setminus [P_{l-1}]$ during the first $l-1$ phases.

Note that for $i \in [P_l] \setminus [P_{l-1}]$, during the $l^0 \le l-1$ phase, we have for $\tau^{r,l^0-1}+1 < t \le (\tau^{r,l^0}+1) \wedge \tau_i^{\Delta}$,

$$\begin{aligned}
w_i^t &\ge (1 - Cr^2) w_i^{\tau^{r,l^0-1}+1} + c\eta_1 \overline{D}^{\tau^{r,l^0-1}+1,t} \\
&\ge (1 - Cr^2) w_i^{\tau^{r,l^0-1}+1} + a^0 \eta_1 \sum_{s=\tau^{r,l^0-1}+1}^{t-1} \Big[c \frac{\chi_{\cdot,l}(\boldsymbol{w}^s)}{w_i^t} - C w_i^t \chi_{\cdot,l^0}(\boldsymbol{w}^s) \Big] \,, \\
w_i^t &\le (1 + Cr^2) w_i^{\tau^{r,l^0-1}+1} + C\eta_1 \underline{D}^{\tau^{r,l^0-1}+1,t} \\
&\ge (1 + Cr^2) w_i^{\tau^{r,l^0-1}+1} + a^0 \eta_1 \sum_{s=\tau^{r,l^0-1}+1}^{t-1} \Big[C \frac{\chi_{\cdot,l}(\boldsymbol{w}^s)}{w_i^t} - c w_i^t \chi_{\cdot,l^0}(\boldsymbol{w}^s) \Big] \,.
\end{aligned} \tag{59}$$

Assume that $\min_{i \in [P_l] \setminus [P_{l-1}]} \tau_i^{\Delta} > \tau^{r,l^0-1}+1$ and

$$\max_{i \in [P_l] \setminus [P_{l-1}]} |w_i^{\tau^{r,l^0-1}+1} - 1/\sqrt{d}| \le \frac{Cr^2}{\sqrt{d}} \,.$$

Denote

$$\tau^{+,l} = \min\{ t \ge \tau^{r,l^0-1}+1 : \max_{i \in [P_j] \setminus [P_{l-1}]} |w_i^t - 1/\sqrt{d}| > 1/(2\sqrt{d}) \} \,.$$

As long as $t < \tau^{+,l}$, then

$$\frac{1 - Cr^2}{\sqrt{d}} - C \frac{a^0 \eta_1}{\sqrt{d}} \sum_{s=\tau^{r,l^0-1}+1}^{t-1} \chi_{\cdot,l^0}(\boldsymbol{w}^s) \le w_i^t \le \frac{1 + Cr^2}{\sqrt{d}} + C \frac{a^0 \eta_1}{\sqrt{d}} \sum_{s=\tau^{r,l^0-1}+1}^{t-1} \chi_{\cdot,l^0}(\boldsymbol{w}^s) \,,$$

where we used the assumption that $D_{l^0} \geq 2$. Using the same argument as in Step 3 of Section C.3, we deduce that as long as $t \leq (\tau^{r,l^0} + 1) \wedge \tau^{+,l}$, then

$$\frac{1 - Cr^2}{\sqrt{d}} \leq w_i^t \leq \frac{1 + Cr^2}{\sqrt{d}} .$$

In particular, we deduce that we must have $\tau^{+,l} > \tau^{r,l^0} + 1$ and therefore $\tau_i^\Delta > \tau^{r,l^0} + 1$ for all $i \in [P_l] \setminus [P_{l\ 1}]$.

By induction, we deduce that for all $i \in [P_l] \setminus [P_{l\ 1}]$, $\tau_i^\Delta > \tau^{r,l\ 1} + 1$ and

$$\frac{1 - Cr^2}{\sqrt{d}} \leq w_i^{\tau^{r,l\ 1}+1} \leq \frac{1 + Cr^2}{\sqrt{d}} .$$

Step 2: Bounding the growth of w_i^t for $i \in [P_l] \setminus [P_{l\ 1}]$.

The same argument as in Step 1 of Section C.3 (recalling that by the previous argument, $\tau_i^r \wedge \tau_i^\Delta > \tau^{r,l\ 1} + 1$ for all $i \in [P_l] \setminus [P_{l\ 1}]$) yields

$$\inf_{\overline{\tau} \wedge T > t > \tau_i^r} w_i^t \geq r - \frac{1}{\sqrt{d \log(d)}} \geq \frac{r}{2} , \qquad \inf_{\tau_i^\Delta < t < \overline{\tau} \wedge T} w_i^t \geq \Delta - \frac{2}{\sqrt{d \log(d)}} .$$

Denote $\alpha_t = \min\{w_i^t : i \in S_t \cap [P_l] \setminus [P_{l\ 1}]\}$ (noting that $w_i^t \geq r/2$ for $i \in [P_l] \setminus [P_{l\ 1}]$ but $i \notin S_t$). Furthermore, for $i \in [P_{l\ 1}]$ and $t > \tau^{r,l\ 1}$, we have $w_i^t \geq r/2$. Hence, for $t \leq \tau^{r,j} + 1$,

$$\alpha_t \geq \alpha^{\tau^{r,l\ 1}+1} + C\eta_1 a^0 r^{\overline{D}_{l\ 1}} \sum_{s=\tau^{r,l\ 1}+1}^{t\ 1} \alpha_s^{D_{l\ 1}} .$$

Furthermore, $\alpha^{\tau^{r,l\ 1}+1} \geq (1 - Cr^2)/\sqrt{d}$ by the previous step. We deduce by Lemma 17: for $D_l > 2$,

$$\alpha_t \geq r \wedge \left\{ \frac{1}{\left(Cd^{D_l/2\ 1} - C\eta_1 a^0 r^{\overline{D}_{l\ 1}} t\right)^{\frac{1}{k\ 2}}} \right\} ,$$

which implies

$$\tau^{r,l} \leq \frac{Cd^{D_l/2\ 1}}{\eta_1 a^0 r^{\overline{D}_{l\ 1}}} .$$

Similarly, for $D_l = 2$,

$$\tau^{r,l} \leq \frac{C \log(d)}{\eta_1 a^0 r^{\overline{D}_{l\ 1}}} .$$

Similarly, we obtain similar bounds on $\tau^{\Delta,l}$ (see Step 2 of Section C.3).

Step 3: Concluding the proof.

Theorem 10.(a) follows by Step 2 and taking η_1 and $\overline{T}_1 := T$ that satisfy (32), (34) and (37), and the growth conditions in Step 2. Theorem 10.(b) follows by the same argument as in Step 3 of Section C.3. ■

D.2. Extending the analysis: adaptive step size and non-nested monomials

D.2.1. ADAPTIVE STEP-SIZE

Let us consider

$$h\ (\boldsymbol{z}) = \sum_{l=1}^{L} \prod_{s2[P_l]} \mathrm{He}_{k_s}(z_s)\,, \tag{60}$$

with increasing[16] leaps $1 \le D_1 < D_2 < \ldots < D_L =: D$, so that neurons align with the support sequentially at increasing time scales. As mentioned below Theorem 10, the time complexity to escape each of these leaps is only tight for the biggest leap if we take a constant step size $\eta \propto d^{\ D/2}$. Indeed, for the first phases of the dynamics, SGD requires a number of steps $d^{(D_l+D)/2\ \ 1}$ much smaller than $d^{D\ \ 1}$ to align to the l-th monomial. In that case, we can take bigger step sizes and still have negligible contribution from the martingale part of the dynamics. In practice, such as in Figure 1, we can see a saddle-to-saddle dynamics[17] to occur, with a number $O(d^{D_l\ \ 1})$ of steps to escape each saddle even for constant step size.

To prove these tight scalings for each plateau with constant step size, we would need to study the joint training of the two layers, which is currently out of reach of our proof techniques. Instead, we show in the next theorem that we can use a learning rate schedule η^t, i.e.,

$$\widetilde{\boldsymbol{w}}_j^{t+1} = \boldsymbol{w}_j^t - \eta^t \cdot \mathrm{grad}_{\boldsymbol{w}_j^t} \ell\big(y^t, \hat{f}_{\mathsf{NN}}(\boldsymbol{x}^t; \boldsymbol{\Theta}^t)\big)\,,$$

to get a scaling $\widetilde{\Theta}(d^{D_l\ \ 1})$ to align to each new monomial.

Theorem 19 (First layer training adaptive step size) *Let $h\ : \mathsf{R}^P \to \mathsf{R}$ be defined as in Eq. (60) and assume σ satisfy Assumption 7. Then for $0 < r < \Delta$ sufficiently small (depending on D, K) and $\rho \le \Delta$ the following holds. For any constant $C\ > 0$, there exist C_i for $i = 0, \ldots, 5$, that only depend on D, K and $C\ $ such that, by splitting our learning rate schedule in L phases with step sizes $\eta^t = \eta_l$ for $t \in\in \{\overline{T}_{l\ \ 1}, \overline{T}_{l\ \ 1} + 1, \ldots, \overline{T}_l - 1\}$, with*

$$\overline{T}_l = C_0 d^{D_l\ \ 1} \log(d)^{C_0}\,, \qquad \eta_l = \frac{1}{C_1 \kappa d^{D_l/2} \log(d)^{C_1}}\,, \qquad \kappa \le \frac{1}{C_2 d^{C_2}}\,,$$

the following events hold with probability at least $1 - MC_3 d^{\ C}\ /r$. For any neuron $j \in [M]$,

(a) Early stopping for $l \in [L]$: $|w_{j,i}^t - w_{j,i}^0| \le C_4/\sqrt{d \log(d)}$ for all $i = P_{l\ \ 1} + 1, \ldots, d$ and $t \le \overline{T}_l/(C_5 \log(d)^{C_5})$.

For any neuron $j \in [M]$ such that $a^0 \mu_{\overline{D}_l}(\sigma)(w_{j,1}^0)^{k_1} \cdots (w_{j,P_l}^0)^{k_{P_l}} > 0$ for all $l \in [L]$,

(b) On the support: $\big|w_{j,i}^{T_l} - \mathit{sign}(w_{j,i}^0) \cdot \Delta\big| \le C_4/\sqrt{d \log(d)}$ for $i = 1, \ldots, P_l$ and $l \in [L]$.

(c) Outside the support: $|w_{j,i}^{T_l} - w_{j,i}^0| \le C_5 r^2/\sqrt{d}$ for $i = P_l + 1, \ldots, d$ and $l \in [L]$. Furthermore, $\sum_{i > P_l} (w_{j,i}^{\overline{T}_l})^2 = 1$.

16. The case $D_1 = 1$ in the first phase of the dynamics can be studied easily by modifying the proof of Theorem 10 and noting that the drift is now just a sum of constant terms.
17. Again, we expect this saddle-to-saddle dynamic to occur in the case of increasing leaps, otherwise we might have mixing of the different phases for different neurons and no plateaus, except at the biggest leap.

There are two key differences between Theorem 19 and Theorem 10. First we prove a tighter scaling $\widetilde{\Theta}(d^{D_l - 1})$ of number of steps for the first phases of the training. Second we show that the alignment is sequential for all the neurons at the same time: at the end $\overline{T}_l$ of each phase, we exactly picked up the support $[P_l]$ and nothing else. In particular, using a similar proof as in Corollary 9.(b), we can show that the neural network at time $\overline{T}_l$ cannot fit the remaining $L - l$ monomials at all using the second layer weights. This agrees with the picture obtained in the numerical simulation in Figure 1.

The $\overline{T}_l$ and η_l are chosen such that the martingale term remain negligible during the whole dynamics. Furthermore, because of the separation of time scales between the different phases of the dynamics, we can show that for $t \le \overline{T}_l$ and step size η_l, the contribution of the drift terms coming from the next monomials remains small. The proof follows almost identically to the proofs of Theorems 8 and 10.

D.2.2. NON-NESTED MONOMIALS

While we wrote Theorems 10 and 19 in the case of h a nested sum of monomials, we note that this is foremost a convenient assumption that help simplify the equations in the proofs. However, the compositionality of the monomials in the decomposition of h is not a required structure for the leap complexity to hold. Note that this compositionality might be favorable if we consider large $P = \omega_d(1)$ (such as in Abbe et al. (2021a)) or for the dependency in ε and the Hermite coefficients of h in the prefactor of $\widetilde{\Theta}(d^{(\mathrm{Leap}(h)-1)_+})$.

Below we describe how we can modify the proof of Theorem 10 for non-compositional h and leave the task of proving Conjecture 2 for general leap functions to future works.

Consider $\boldsymbol{k}_l = (k_1^{(l)}, \dots, k_{P_l}^{(l)}) \in \mathsf{N}^{P_l}$ for $l \in [L]$ such that

$$h(\boldsymbol{z}) = \sum_{l=1}^{L} \mathrm{He}_{\boldsymbol{k}_l}(\boldsymbol{z}), \qquad \mathrm{He}_{\boldsymbol{k}}(\boldsymbol{z}) = \prod_{s \in [|\boldsymbol{k}|]} \mathrm{He}_{k_s}(z_s), \tag{61}$$

and $k_s^{(l)} > 0$ for $s \in [P_l] \setminus [P_{l-1}]$ (each new coordinates appear in the next monomial) and $\overline{D}_l = \|\boldsymbol{k}_l\|_1$ with $\overline{D}_1 < \overline{D}_2 < \dots < \overline{D}_L$, and denote $D_l = \overline{D}_l - \overline{D}_{l-1}$ (with $\overline{D}_0 = 0$). Denote $D = \max_{l \in [L]} D_l$ which corresponds to the leap complexity of h.

First note that the same formulas as in Lemma 18 hold with

$$\chi_{\cdot,l}(\boldsymbol{w}^t) = \mathbb{1}\{k_i^{(l)} > 0\} \prod_{j \in [P_l]} (w_j^t)^{k_j^{(l)}},$$

however, we cannot simplify the gradient to be of order $\chi_{\cdot,l}(\boldsymbol{w}^t)/w_i^t$ during the l-th phase. Below, we outline how to modify the proof of Theorem 8 in Section D.1 to the case (61). The bounds on the martingale terms and on the dynamics from Section C.2 still hold in that case, with the difference being in the formulas of the population gradients.

By taking Δ small enough, there exists constants C, c such that we can upper and lower bound $\overline{g}_i^t$ as follows. For $i \in [P_l] \setminus [P_{l-1}]$, during the $l^\emptyset \le l - 1$ phase, we have for $\tau^{r,l^\emptyset - 1} + 1 < t \le$

$(\tau^{r,l^0}+1)\wedge\tau_i^{\Delta}$,

$$\overline{g}_i^t \le Ca^0 \sum_{q\ \ l} \frac{\chi_{\ ,q}(\boldsymbol{w}^t)}{w_i^t} - ca^0 \sum_{l^0\ \ q<l} w_i^t \chi_{\ ,q}(\boldsymbol{w}^t)\,,$$

$$\overline{g}_i^t \ge ca^0 \sum_{q\ \ l} \frac{\chi_{\ ,q}(\boldsymbol{w}^t)}{w_i^t} - Ca^0 \sum_{l^0\ \ q<l} w_i^t \chi_{\ ,q}(\boldsymbol{w}^t)\,.$$

If $t > \tau^{r,l\ \ 1}$, then

$$ca^0 \sum_{q\ \ l} \frac{\chi_{\ ,q}(\boldsymbol{w}^t)}{w_i^t} \le \overline{g}_i^t \le Ca^0 \sum_{q\ \ l} \frac{\chi_{\ ,q}(\boldsymbol{w}^t)}{w_i^t}\,.$$

We can plug these population gradients in steps 1 and 2 in Theorem 10, and control the contribution of each of these terms using $\alpha_{l,t} = \min\{w_i^t : i \in S_t \cap [P_l] \setminus [P_{l\ \ 1}]\}$ and $\beta_{l,t} = \max\{w_i^t : i \in S_t \cap [P_l] \setminus [P_{l\ \ 1}]\}$, with similar arguments as in step 2 of Section C.3.

Appendix E. Fitting the second layer weights: proof of Corollaries 9 and 11

E.1. Proof of Corollary 9: fitting one monomial

We first focus on the case $h\ (\boldsymbol{z}) = z_1 \cdots z_P$ and prove parts (a) and (b) separately in Sections E.1.1 and E.1.2. The case of $h\ (\boldsymbol{z}) = \mathrm{He}_D(z_1)$ follows from a similar argument and we outline the differences in Section E.1.3.

E.1.1. SECOND-LAYER FITTING

Recall that in this case $P = D$ and we can use both interchangeably. We consider the case of no biases in this part, i.e., fixing $b_j = 0, j \in [M]$.

Phase I: first layer weights. By Theorem 8, with probability at least $1 - Md^{-C}$, for each neuron $(a, \boldsymbol{w})$ satisfying $a^0 \mu_P(\sigma) w_1^0 \cdots w_P^0 > 0$ at initialization, we get at the end of the dynamics:

$$\text{For } i \in [P]: \qquad |w_i^{\overline{T}_1} - \mathrm{sign}(w_i^0) \cdot \Delta| \le C/\sqrt{d \log(d)}\,, \tag{62}$$

$$\text{For } i \notin [P]: \qquad |w_i^{\overline{T}_1} - w_i^0| \le Cr/\sqrt{d}\,, \qquad \sum_{i=P+1}^{d} (w_i^{\overline{T}_1})^2 = 1\,. \tag{63}$$

For the remainder of the proof, assume the above event is true.

Constructing good features. Now we show that for any sign vector $\boldsymbol{\delta} \in \{\pm 1\}^P$ we can combine multiple trained neurons to approximate the function $\mathsf{E}_G[\sigma(\Delta\langle \boldsymbol{\delta}, \boldsymbol{z}\rangle + G)]$.

Lemma 20 *There exists a constant C that depends only on D, K such that the following is true. For any R weights $\{\boldsymbol{w}_{j_1}^0, \ldots, \boldsymbol{w}_{j_R}^0\}$ which coincide on the first P coordinates $\boldsymbol{w}_{j_s,1:P}^0 = \boldsymbol{\delta}/\sqrt{d}$ where $\boldsymbol{\delta} \in \{\pm 1\}^P$, and with biases b_{j_s} such that $|b_{j_s} - b| \le 1/R$, and with sign$(a_{j_s}^0) =$ sign$(\mu_P(0) w_{j_s,1}^0 \ldots w_{j_s,P}^0)$, there exists a constant C that only depends on P, K such that*

$$\begin{aligned}
&\mathsf{E}_{\boldsymbol{x}}\Big[\Big(\frac{1}{R}\sum_{s \in [R]} \sigma(\langle \boldsymbol{w}_{j_s}^{\overline{T}_1}, \boldsymbol{x}\rangle + b_{j_s}) - \mathsf{E}_G[\sigma(\Delta\langle \boldsymbol{\delta}, \boldsymbol{z}\rangle + G + b)]\Big)^2\Big] \\
&\qquad \le \frac{C}{\sqrt{d\log(d)}} + Cr + \frac{C}{R} + \frac{C}{R^2} \sum_{s, s' \in [R]} |\langle \boldsymbol{w}_{j_s,P+1:d}^0, \boldsymbol{w}_{j_{s'},P+1:d}^0\rangle|
\end{aligned}$$

Proof First if we replace $\boldsymbol{w}_{j_s}^{\overline{T}_1}$ by $\Delta\boldsymbol{\delta}$ and b_{j_s} by b, the error is bounded by

$$|\sigma(\langle \boldsymbol{w}_{j_s}^{\overline{T}_1}, \boldsymbol{x}\rangle + b_{j_s}) - \sigma(\Delta\langle \boldsymbol{\delta}, \boldsymbol{z}\rangle + \langle \boldsymbol{w}_{j_s,P+1:d}^{\overline{T}_1}, \boldsymbol{x}_{P+1:d}\rangle + b)| \le \frac{2PK}{\sqrt{d\log(d)}} + \frac{K}{R},$$

which is accounted for by the first two terms since we can take C large enough depending on P, K. For the last two error terms,

$$\begin{aligned}
&\mathsf{E}_{\boldsymbol{x}}\Big[\Big(\frac{1}{R}\sum_{s \in [R]} \sigma(\Delta\langle \boldsymbol{\delta}, \boldsymbol{z}\rangle + \langle \boldsymbol{w}_{j_s,P+1:d}^{\overline{T}_1}, \boldsymbol{x}_{P+1:d}\rangle) - \mathsf{E}_G[\sigma(\Delta\langle \boldsymbol{\delta}, \boldsymbol{z}\rangle + G)]\Big)^2\Big] \\
&= \mathsf{E}_{\boldsymbol{x}}\Big[\Big(\frac{1}{R}\sum_{s \in [R]} h(\boldsymbol{z}, \langle \boldsymbol{x}_{P+1:d}, \boldsymbol{w}_{j_s,P+1:d}^{\overline{T}_1}\rangle)\Big)^2\Big] = (*)
\end{aligned}$$

where $h(\boldsymbol{z}, u) = \sigma(\Delta\langle\boldsymbol{\delta}, \boldsymbol{z}\rangle + u + b) - \mathsf{E}_G[\sigma(\Delta\langle\boldsymbol{\delta}, \boldsymbol{z}\rangle + G + b)]$.

If $\boldsymbol{u}$ satisfies $\|\boldsymbol{u}\| = 1$, then $\langle \boldsymbol{u}, \boldsymbol{x}_{P+1:d}\rangle$ is distributed as $\mathsf{N}(0, 1)$. So for all $\boldsymbol{z}$,

$$\mathsf{E}_{\boldsymbol{x}_{P+1:d}}[h(\boldsymbol{z}, \langle \boldsymbol{x}_{P+1:d}, \boldsymbol{u}\rangle)] = 0$$

Furthermore, for any $\boldsymbol{u}, \boldsymbol{v}$ satisfying $\|\boldsymbol{u}\| = \|\boldsymbol{v}\| = 1$, let $\tilde{\boldsymbol{u}} = \boldsymbol{u} - \boldsymbol{v}\langle \boldsymbol{u}, \boldsymbol{v}\rangle$, which satisfies $\langle \tilde{\boldsymbol{u}}, \boldsymbol{v}\rangle = 0$. Then $\langle \boldsymbol{x}_{P+1:d}, \tilde{\boldsymbol{u}}\rangle$ and $\langle \boldsymbol{x}_{P+1:d}, \boldsymbol{v}\rangle$ are independent so

$$\begin{aligned}
&|\mathsf{E}_{\boldsymbol{x}_{P+1:d}}[h(\boldsymbol{z}, \langle \boldsymbol{x}_{P+1:d}, \boldsymbol{u}\rangle)h(\boldsymbol{z}, \langle \boldsymbol{x}_{P+1:d}, \boldsymbol{v}\rangle)]| \\
&\le |\mathsf{E}_{\boldsymbol{x}_{P+1:d}}[h(\boldsymbol{z}, \langle \boldsymbol{x}_{P+1:d}, \tilde{\boldsymbol{u}}\rangle)h(\boldsymbol{z}, \langle \boldsymbol{x}_{P+1:d}, \boldsymbol{v}\rangle)]| + |\mathsf{E}_{\boldsymbol{x}_{P+1:d}}[|K\langle \boldsymbol{x}_{P+1:d}, \tilde{\boldsymbol{u}} - \tilde{\boldsymbol{v}}\rangle| h(\boldsymbol{z}, \langle \boldsymbol{x}_{P+1:d}, \boldsymbol{v}\rangle)]| \\
&= K|\mathsf{E}_{\boldsymbol{x}_{P+1:d}}[|\langle \boldsymbol{x}_{P+1:d}, \tilde{\boldsymbol{u}} - \boldsymbol{u}\rangle| h(\boldsymbol{z}, \langle \boldsymbol{x}_{P+1:d}, \boldsymbol{v}\rangle)]| \\
&\le K\sqrt{\mathsf{E}_{\boldsymbol{x}_{P+1:d}}[\langle \boldsymbol{x}_{P+1:d}, \tilde{\boldsymbol{u}} - \boldsymbol{u}\rangle^2]} \\
&= K|\langle \boldsymbol{u}, \boldsymbol{v}\rangle|\,.
\end{aligned}$$

So

$$(*) \le \frac{K}{R^2} \sum_{s,s^0 2[R]} |\langle \boldsymbol{w}^{\overline{T}_1}_{j_s, P+1:d}, \boldsymbol{w}^{\overline{T}_1}_{j_{s^0}, P+1:d}\rangle| \le Cr + \frac{K}{R^2} \sum_{s,s^0 2[R]} |\langle \boldsymbol{w}^{0}_{j_s, P+1:d}, \boldsymbol{w}^{0}_{j_{s^0}, P+1:d}\rangle|\,,$$

which concludes the proof of the lemma. ∎

Certificate. We now write $h\ (\boldsymbol{z}) = \prod_{i=1}^P z_i$ as a linear combination of functions of the form $\mathsf{E}_G[\sigma(\Delta\langle\boldsymbol{\delta}, \boldsymbol{z}\rangle + G)]$ for different $\boldsymbol{\delta} \in \{+1, -1\}^P$. By a Taylor approximation, for any $0 < s < 1$ and $x \in [-s, s]$,

$$\mathsf{E}_G[\sigma(x + G)] = \sum_{k=0}^{P} \frac{\mu_k(\sigma)}{k!} x^k + O(s^{P+1})\,,$$

with a constant in the $O(\cdot)$ that depends only on P, K. So if we define the coefficient

$$c_{\boldsymbol{\delta}} = \frac{P!}{2^P \Delta^P \mu_P(\sigma)} \prod_{i=1}^{P} \delta_i$$

then we can approximate $h\ (\boldsymbol{z}) = \prod_{i=1}^P z_i$ as follows for any $\boldsymbol{z}$ such that $\Delta|\langle\boldsymbol{\delta}, \boldsymbol{z}\rangle| < 1$,

$$\begin{aligned}
\sum_{\boldsymbol{\delta} 2 \mathsf{f}\ \ 1\mathsf{g}^P} c_{\boldsymbol{\delta}} \mathsf{E}_G[\sigma(\Delta\langle\boldsymbol{\delta}, \boldsymbol{z}\rangle + G)] &= \sum_{k=0}^{P} \frac{\mu_k(\sigma)}{k!} \sum_{\boldsymbol{\delta} 2 \mathsf{f}\ \ 1\mathsf{g}^P} \Delta^k \langle\boldsymbol{\delta}, \boldsymbol{z}\rangle^k c_{\boldsymbol{\delta}} + O(\Delta|\langle\boldsymbol{\delta}, \boldsymbol{z}\rangle|^k) \\
&= \prod_{i=1}^{P} z_i + O(\Delta|\langle\boldsymbol{\delta}, \boldsymbol{z}\rangle|^k)\,,
\end{aligned}$$

where we use that for any $S \subseteq [P]$, we have $\frac{1}{2^P} \sum_{\boldsymbol{\delta}} (\prod_{i=1}^P \delta_i)(\prod_{i 2 S} \delta_i) = \begin{cases} 0, & S \neq [P] \\ 1, & S = [P] \end{cases}$.

Putting this together with Lemma 20 and the guarantees on the first layer weights after training (62) and (63), we obtain the following lemma.

Lemma 21 *There exists a constant $C > 0$ depending only on P, K such that with probability at least $1 - d^{-C} - C\varepsilon$ there exists a set of weights $\boldsymbol{\Theta}^{cert} = (\boldsymbol{W}^{cert}, \boldsymbol{a}^{cert})$ satisfying*

- *(First layer weights are the trained weights) For all $j \in [M]$, we have $\boldsymbol{w}_j^{\overline{T}_1} = \boldsymbol{w}_j^{cert}$.*
- *(Second-layer weights are small) We have $\|\boldsymbol{a}^{cert}\| \le C/(\Delta^P \sqrt{M})$.*
- *(Squared error is small) We have $R^{sq}(\boldsymbol{\Theta}^{cert}) \le \varepsilon/4$.*

Proof Consider the event that for each $\boldsymbol{\delta} \in \{\pm 1\}^P$ the set $S_{\boldsymbol{\delta}} = \{j : a_j^0 \mu_D(\sigma) w_{j,1}^0 \cdots w_{j,P}^0 > 0\}$ is of size $|S_{\boldsymbol{\delta}}| \ge R := M/2^{P+2}$. This holds with probability at least $1 - O(\varepsilon)$ by a union bound and a Hoeffding bound, so we condition on it from now on. Consider the event that for all $\boldsymbol{\delta}$ we have

$$\frac{1}{|S_{\boldsymbol{\delta}}|^2} \sum_{j,j' \in S_{\boldsymbol{\delta}}} |\langle \boldsymbol{w}_{j,P+1:d}^0, \boldsymbol{w}_{j',P+1:d}^0 \rangle| \le \frac{1}{R} + C_{11}\sqrt{\frac{\log(d)}{d}}$$

and note that this holds with probability at least $1 - d^{-C}$ by a Hoeffding bound for a constant C_{11} depending on P, K, C, so we also condition on it.

Let $\boldsymbol{a}^{cert}$ be given by $a_j^{cert} = c_{\boldsymbol{\delta}}/|S_{\boldsymbol{\delta}}|$ if $j \in S_{\boldsymbol{\delta}}$, and 0 otherwise. From this it follows that

$$\|\boldsymbol{a}^{cert}\| \le (\sqrt{M}/R) \max_{\boldsymbol{\delta}} |c_{\boldsymbol{\delta}}| \le C/(\Delta^P \sqrt{M}),$$

for a constant C depending only on P, K. By Lemma 20,

$$\begin{aligned}
R^{sq}(\boldsymbol{\Theta}^{cert}) &= \mathbf{E}_{\boldsymbol{x}}\Big[\Big(\prod_{i=1}^P x_i - \sum_{\boldsymbol{\delta}} c_{\boldsymbol{\delta}} \frac{1}{|S_{\boldsymbol{\delta}}|} \sum_{j \in S_{\boldsymbol{\delta}}} \sigma(\langle \boldsymbol{w}_j^{\overline{T}_1}, \boldsymbol{x}\rangle)\Big)^2\Big] \\
&\le C\mathbf{E}_{\boldsymbol{x}}\Big[\Big(\prod_{i=1}^P x_i - \sum_{\boldsymbol{\delta}} c_{\boldsymbol{\delta}} \mathsf{E}_G[\sigma(\Delta\langle \boldsymbol{\delta}, \boldsymbol{x}_{1:P}\rangle + G)]\Big)^2\Big] \\
&\qquad + C\Big(\sqrt{\frac{C_{11}\log(d)}{d}} + \frac{1}{R} + r\Big)^2 \\
&\le \min_{s \in (0,1/\Delta)} C\Big\{K^2 \mathsf{P}_{\boldsymbol{x}}[|\langle \boldsymbol{\delta}, \boldsymbol{x}_{1:P}\rangle| > s] + \Delta^2 s^{2k} + \frac{C_{11}\log(d)}{d} + \frac{1}{R^2} + r^2\Big\} \\
&\le \min_{s \in (0,1/\Delta)} C\Big\{K^2 \exp(-(s/P)^2) + \Delta^2 s^{2k} + \frac{C_{11}\log(d)}{d} + \frac{1}{R^2} + r^2\Big\} \\
&\le \varepsilon/4,
\end{aligned}$$

by taking a small enough choice of parameters Δ, r and large enough d, M. ∎

Concluding fitting of the monomial: Now that we have constructed the certificate $\boldsymbol{\Theta}^{cert}$, we show that SGD on the second layer converges quickly to a solution with low population loss by a bias-variance analysis of SGD for ridge-regularized least-squares linear regression in Lemma 25. We train the second-layer while keeping the weights of the first layer fixed, which corresponds to linear regression with input embedding

$$\phi(\boldsymbol{x}) = [\sigma(\langle \boldsymbol{w}_j^{\overline{T}_1}, \boldsymbol{x}\rangle)]_{j \in [m]} \in \mathsf{R}^m.$$

Because of the boundedness of σ, we have $\|\phi(\boldsymbol{x})\| \le K\sqrt{M}$ almost surely over $\boldsymbol{x}$. Also, the initialization of the second layer implies $\|\boldsymbol{a}^0\| \le 1/\sqrt{M}$. Finally, the labels $y^{\overline{T}_1}, y^{\overline{T}_1+1}, \dots, y^{\overline{T}_2}$ satisfy $\mathsf{E}[(y^t)^2] \le K$ and $|y^t| \le C_0 \log(1/\varepsilon)^{C_0}$ for all $\overline{T}^1 < t \le \overline{T}^2 - 1$ with probability at least $1 - d^{-C}$ by (53) and a union bound. So applying Lemma 25 from Section E.3, there is a constant C_{12} depending on D, K such that if $\lambda_a \le M$,

$$\mathsf{P}\Big[R^{sq}(\boldsymbol{\Theta}^{\overline{T}_1+\overline{T}_2}) \ge R^{sq}(\boldsymbol{\Theta}^{cert}) + \frac{\lambda_a}{2}\|\boldsymbol{a}^{cert}\|^2 + C_{12}\log(1/\varepsilon)^{C_{12}} M\left((1-\lambda_a\eta)^{2\overline{T}^2}(\frac{1}{M}+\frac{1}{\lambda_a}) + \log(\overline{T}_2/\delta)\frac{\eta M^2}{\lambda_a^2}\right)\Big] \le \delta\,.$$

So, plugging in Lemma 21 and taking $\lambda_a = \varepsilon\Delta^{2P}M/(4C)$, $\delta = \varepsilon$

$$\mathsf{P}\Big[R^{sq}(\boldsymbol{\Theta}^{\overline{T}_1+\overline{T}_2}) \ge \varepsilon/2 + C_{12}\log(1/\varepsilon)^{C_{12}} M\Big((1-\frac{\varepsilon\Delta^{2P}M}{4C}\eta)^{2\overline{T}^2}(\frac{1}{M}+\frac{4C}{\varepsilon M\Delta^{2P}}) + \log(\frac{\overline{T}_2}{\varepsilon})\frac{16C^2\eta}{\varepsilon^2\Delta^{4P}}\Big)\Big] \le C\varepsilon + d^{-C}\,.$$

By taking $\eta = \frac{\varepsilon^4\Delta^{4P}}{16MC^2}$ and $\overline{T}^2 = \frac{64C^3}{\varepsilon^6\Delta^{6P}}$, for small enough ε,

$$\mathsf{P}\Big[R^{sq}(\boldsymbol{\Theta}^{\overline{T}_1+\overline{T}_2}) \ge \varepsilon\Big] \le C\varepsilon + d^{-C}\,.$$

This proves part (a) of Corollary 9.

E.1.2. CONVERSE IF EARLY STOPPING

We now prove the converse. The proof will follow very similarly to the proof of (Ghorbani et al., 2021, Theorem 1). By Theorem 8, if we train the first layer for time $\overline{T}_1^0 \le \overline{T}_1/(C_8\log(d)^{C_8})$ steps for a large enough $C_8 > 0$, then with probability at least $1 - Md^{-C}$ for each neuron $j \in [M]$,

$$|w_{j,i}^{\overline{T}_1^0} - w_{j,i}^0| \le C_4/\sqrt{d\log(d)} \text{ for all } i \in [d]\,, \tag{64}$$

and some constant C_4. In particular, this implies that for large enough d,

$$\left|w_{j,i}^{\overline{T}_1^0}\right| \le 2/\sqrt{d}\,.$$

For ease of notations, denote $\boldsymbol{w}_j := \boldsymbol{w}_j^{\overline{T}_1^0}$. Let us introduce $\phi(\boldsymbol{x}) = [\sigma(\langle \boldsymbol{w}_1, \boldsymbol{x}\rangle), \dots, \sigma(\langle \boldsymbol{w}_M, \boldsymbol{x}\rangle)]$ and $\phi_0(\boldsymbol{x}) = [\sigma(\langle \boldsymbol{w}_1^0, \boldsymbol{x}\rangle), \dots, \sigma(\langle \boldsymbol{w}_M^0, \boldsymbol{x}\rangle)]$.

By a simple calculation, we have

$$\min_{\boldsymbol{a}\in\mathsf{R}^M} \mathsf{E}_{\boldsymbol{x}}\Big[\big(f(\boldsymbol{x}) - \boldsymbol{a}^{\mathsf{T}}\phi(\boldsymbol{x})\big)^2\Big] = \|f(\boldsymbol{x})\|_{L^2}^2 - \boldsymbol{V}^{\mathsf{T}}\boldsymbol{U}^{-1}\boldsymbol{V}\,,$$

where we denoted

$$\boldsymbol{V} = \mathsf{E}_{\boldsymbol{x}}[\phi(\boldsymbol{x})f(\boldsymbol{x})] \in \mathsf{R}^M\,, \qquad \boldsymbol{U} = \mathsf{E}_{\boldsymbol{x}}[\phi(\boldsymbol{x})\phi(\boldsymbol{x})^{\mathsf{T}}] \in \mathsf{R}^{M\times M}\,.$$

Corollary 9 will follow by showing that there exist constants c, C that only depend on K, P such that with high probability, we have

$$\lambda_{\min}(\boldsymbol{U}) \geq c, \qquad \|\boldsymbol{V}\|_2^2 \leq CMd^{-P}.$$

These are proved in the following two lemmas.

Lemma 22 *Under the same setting as in Corollary 9, there exist constants $c, C > 0$ such that with probability at least $1 - CMd^{-C}$,*

$$\lambda_{\min}(\boldsymbol{U}) \geq c\,.$$

Proof [Proof of Lemma 22] Consider the event described in Eq. (64). By rotational invariance of the distribution of $\boldsymbol{x}$, the entries $\boldsymbol{U} = (U_{ij})_{i,j\in[n]}$ are given by

$$U_{ij} = \mathsf{E}_{\boldsymbol{x}}[\sigma(\langle \boldsymbol{w}_i, \boldsymbol{x}\rangle)\sigma(\langle \boldsymbol{w}_j, \boldsymbol{x}\rangle)] = \mathsf{E}_{G_1,G_2}\Big[\sigma(\alpha_i G_1)\sigma\Big(\alpha_j\beta_{ij}G_1 + \alpha_j\sqrt{1-\beta_{ij}^2}G_2\Big)\Big]\,,$$

where $(G_1, G_2) \sim \mathsf{N}(0, \mathbf{I}_2)$, $\alpha_i = \|\boldsymbol{w}_i\|_2$ and $\beta_{ij} = \langle \boldsymbol{w}_i, \boldsymbol{w}_j\rangle/(\alpha_i\alpha_j)$.

For $i = j$, we have

$$U_{ii} = \mathsf{E}_G[\sigma(\alpha_i G)^2] = \mathsf{E}_G[\sigma(G)^2] + \mathsf{E}_G[\sigma(\alpha_i G)^2 - \sigma(G)^2]\,.$$

We can do a Taylor expansion and bound the second term

$$\begin{aligned}
\big|\mathsf{E}_G[\sigma(\alpha_i G)^2 - \sigma(G)^2]\big| &\leq \Big|\mathsf{E}_G\Big[\Big(\sigma(G) + (\alpha_i - 1)G\sigma'(c(G))\Big)^2 - \sigma(G)^2\Big]\Big| \\
&\leq 2K^2|\alpha_i - 1|\mathsf{E}[|G|] + K^2|\alpha_i - 1|^2\mathsf{E}[|G|^2] \\
&\leq \frac{C}{\sqrt{\log(d)}}\,,
\end{aligned}$$

where we used that $|\|\boldsymbol{w}_i\|_2 - 1| \leq C/\sqrt{\log(d)}$ by Eq. (64).

Consider now $i \neq j$. Note that

$$h(t) = \mathsf{E}_{G_1,G_2}\Big[\sigma(G_1)\sigma\Big(tG_1 + \sqrt{1-t^2}G_2\Big)\Big]\,,$$

has derivative

$$h'(t) = \mathsf{E}_{G_1,G_2}\Big[\sigma(G_1)\sigma'\Big(tG_1 + \sqrt{1-t^2}G_2\Big)(G_1 + t/\sqrt{1-t^2}G_2)\Big]\,.$$

Hence, for $|t| \leq 1/2$, we have $|h'(t)| \leq C|t|$. Note that $|\beta_{ij} - \langle \boldsymbol{w}_i^0, \boldsymbol{w}_j^0\rangle| \leq C/\sqrt{\log(d)}$. By standard concentration, using that $\boldsymbol{w}_i^0 \sim \mathrm{Unif}(\{\pm 1/\sqrt{d}\}^d)$, there exists constants c, C such that with probability at least $1 - e^{-cd}$, we have

$$\max_{i\neq j\in[M]} |\langle \boldsymbol{w}_i^0, \boldsymbol{w}_j^0\rangle| \leq C\log(M)/\sqrt{d}\,.$$

Using the same computation as above, we can replace α_i and α_j by 1 while only incurring an error $C/\sqrt{\log(d)}$, and show that

$$|U_{ij} - h(0)| \leq C/\sqrt{\log(d)} + C\log(M)/\sqrt{d}\,.$$

From the above bounds, we deduce (using $\|\boldsymbol{M}\|_{\rm op} \le \|\boldsymbol{M}\|_F$) that with high probability

$$\|\boldsymbol{U} - h(0)^2 \mathbf{1}\mathbf{1}^{\mathsf{T}} - (h(1)-h(0))\mathbf{I}\|_{\rm op} \le CM/\sqrt{\log(d)}\,.$$

For σ not constant, $h(1) > h(0)$ and using that $M = O_d(1)$, we deduce that

$$\lambda_{\min}(\boldsymbol{U}) \ge \frac{h(1)-h(0)}{2}\,,$$

which concludes the proof. ■

Lemma 23 *Under the same setting as in Corollary 9, there exists constants $C > 0$ such that with probability at least $1 - CMd^{\ C}$,*

$$\|\boldsymbol{V}\|_2^2 \le CMd^{\ P}\,.$$

Proof [Proof of Lemma 23] Note that we have

$$\|\boldsymbol{V}\|_2^2 = \sum_{j \in [M]} \mathsf{E}_{\boldsymbol{x}}[f^{\ }(\boldsymbol{x})\sigma(\langle \boldsymbol{w}_j, \boldsymbol{x}\rangle)]^2$$

First note that for any $\boldsymbol{w}$, the correlation of $\sigma(\langle \boldsymbol{w}, \boldsymbol{x}\rangle)$ with $f^{\ }(\boldsymbol{x}) = \prod_{i=1}^P x_i$ is bounded by

$$|\mathsf{E}_{\boldsymbol{x}}[\sigma(\langle \boldsymbol{w}, \boldsymbol{x}\rangle) f^{\ }(\boldsymbol{x})]| \le K \prod_{i \in [P]} |w_i|\,.$$

Indeed, as in the proof of Lemma 13, we use the formula from integration by parts:

$$\mathsf{E}_{\boldsymbol{x}}\Big[\sigma(\langle \boldsymbol{w}, \boldsymbol{x}\rangle) \prod_{i \in [P]} x_i\Big] = \Big(\prod_{i \in [P]} w_i\Big) \cdot \mathsf{E}_G[\sigma^{(P)}(\|\boldsymbol{w}\|_2 G)]\,,$$

using $\|\sigma^{(P)}\|_{1} \le K$.

We conclude by noting that on the high probability event (64), we have $|w_{j,i}| \le 2/\sqrt{d}$. ■

E.1.3. PROOF FOR A SINGLE-INDEX HERMITE MONOMIAL

Let's now consider $h^{\ }(\boldsymbol{z}) = \mathrm{He}_D(z_1)$. In this case, we consider the biases $b_j \sim \mathrm{Unif}([-\Delta, \Delta])$, where Δ is chosen sufficiently small as discussed in Theorem 8. We can use the same proof strategy as in Section E.1.1 and construct good features

$$\mathsf{E}_G[\sigma(\Delta z_1 + b + G)]$$

for any $b \in [-\Delta, \Delta]$, by considering neurons with initializations $\{(\boldsymbol{w}_{j_1}^0, b_{j_1}^0), \dots, (\boldsymbol{w}_{j_R}^0, b_R^0)\}$ with $w_{j_s}^1 = 1/\sqrt{d}$ and $\mathrm{sign}(a_{j_s}^0) = \mathrm{sign}(\mu_D(0) w_{j_s,1}^D)$, and $|b_{j_s}^0 - b| \le r$ (by an easy modification of Lemma 20). We will take sufficiently many neurons (but still independent of d) so that we have a sufficiently large R for any intervals of size r for $b \in [-\Delta, \Delta]$ with high probability.

Let us now construct a certificate for $\mathrm{He}_D(z_1)$ based on these good features. By a Taylor approximation, for any $0 < s < 1$ and $x \in [-s, +s]$,

$$\begin{aligned}
\mathsf{E}_G[\sigma(x+b+G)] &= \sum_{k=0}^{D} \frac{\mu_k(\sigma)}{k!}(x+b)^k + O(s^{D+1} + \Delta^{D+1}) \\
&= \sum_{k=0}^{D} b^k \left[\sum_{s=0}^{D} \frac{\mu_{k+s}(\sigma)}{(k+s)!} \binom{k+s}{s} x^s \right] + O(s^{D+1} + \Delta^{D+1}) \\
&=: \sum_{k=0}^{D} b^k Q_{D\ \ k}(x) + O(s^{D+1} + \Delta^{D+1})\,.
\end{aligned}$$

We can consider measures with density $\nu_\ell(b)$ with respect to $b \sim \mathrm{Unif}([-\Delta, \Delta])$ such that

$$\begin{aligned}
\int_{\Delta}^{\Delta} \mathsf{E}_G[\sigma(x+b+G)]\nu_\ell(b)\mathrm{d}b &= \sum_{k=0}^{P} Q_{P\ \ k}(x) \int_{\Delta}^{\Delta} b^k \nu_\ell(b)\mathrm{d}b + O(s^{D+1} + \Delta^{D+1}) \\
&= Q_{D\ \ \ell}(x) + O(s^{D+1} + \Delta^{D+1})\,.
\end{aligned}$$

Note that the polynomials $\{Q_k\}_{k=0,\dots,D}$ are linearly independent (distinct degrees) with coefficients that only depend on D, K. Hence we can take a linear combination of $\nu_\ell(b)$ with coefficients that only depend on D and K such that we have $\tilde{\nu}_\ell$ with

$$\int_{\Delta}^{\Delta} \mathsf{E}_G[\sigma(x+b+G)]\tilde{\nu}_\ell(b)\mathrm{d}b = x^\ell + O(s^{D+1} + \Delta^{D+1})\,.$$

In particular, we can rescale and sum these coefficients such that for some $\nu\ (b)$ that has second moment bounded by $1/\Delta^{CD}$,

$$\int_{\Delta}^{\Delta} \mathsf{E}_G[\sigma(\Delta z+b+G)]\nu\ (b)\mathrm{d}b = \mathrm{He}_D(z_1) + O(s(s/\Delta)^D + \Delta)\,.$$

We can now construct a certificate by sampling b_s from the signed measure $\nu\ (b)$, and for each b_s constructing an approximate good feature, as described in Lemma 20. The proof for the low test error then follows from applying the bound on the least squares linear regression of Lemma 25.

For the lower bound with early stopping, we use that

$$|\mathsf{E}_{\boldsymbol{x}}[\sigma(\langle \boldsymbol{w}, \boldsymbol{x}\rangle + b)\mathrm{He}_D(x_1)]| = |\mathsf{E}_{\boldsymbol{x}}[w_1^D \sigma^{(D)}(\langle \boldsymbol{w}, \boldsymbol{x}\rangle + b)]| \le K|w_1|^D\,,$$

and we can conclude using the same argument as in Section E.1.2.

E.2. Proof of Corollary 11: sequential learning of monomials

Let us formally state Corollary 11 and prove it.

Corollary 24 (Second layer training, sum of monomials; formal statement) *Let*

$$h\ (\boldsymbol{z}) = \sum_{l2[L]} \prod_{i=1}^{P_l} z_l$$

for some $P_1 < P_2 < \cdots < P_L = P$. Then there exists an activation function $\sigma : \mathsf{R} \to \mathsf{R}$ such that the following is true. For any constants $C > 0$ and $\varepsilon > 0$, there exist C_i for $i = 0, \dots, 10$, that only depend on D, K and C such that taking width $M = C_0 \varepsilon^{C_0}$, bias initialization scale $\rho = C_1$, second-layer initialization scale $\kappa = \frac{1}{C_2 M d^{C_2}}$, second-layer regularization $\lambda_a = M\varepsilon/C_3$, and $\Delta = \varepsilon^{C_4}/C_4$, and $r = \varepsilon^{C_5}/C_5$, and

$$\overline{T}_1 = C_6 d^{D^{\,1}} \log(d)^{C_6}\,, \qquad \eta_1 = \frac{1}{C_7 \kappa d^{D/2} \log(d)^{C_7}}\,,$$
$$\overline{T}_2 = C_8 \varepsilon^{\,C_8}\,, \qquad \eta_2 = \varepsilon^{C_9}/(C_9 M)\,,$$

we have for large enough $d \geq C_{10}\varepsilon^{\,C_{10}}$, that with probability at least $1 - d^{\,C} - \varepsilon$ at the end of the dynamics,

$$R(\boldsymbol{\Theta}^{\overline{T}_1 + \overline{T}_2}) \leq \varepsilon\,.$$

In contrast to the proof of Corollary 9, we only prove this result for "diverse" enough activation functions. For the proof, we will construct a specific activation function that have this "diversity" property. This activation depends on P (or upper bound on P), but otherwise is independent of h . The idea is that we will use biases of different magnitudes, which will change the signs of the Hermite coefficients of the activation, in order to ensure enough neurodiversity to learn the sum of increasing monomials. This is required due to the specific choice of training of the first layer weights considered in this paper. However, we show in simulations that standard ReLus activations are enough to learn these functions.

Construction of activation function For any bias $b \in \mathsf{R}$, define $\sigma_b(x) = \sigma(x + b)$. We construct the activation function such that for all $\boldsymbol{s} \in \{+1, -1\}^P$ there is a bias $b(\boldsymbol{s}) \in [-C, C]$ satisfying

$$\mu_k(\sigma_{b(\boldsymbol{s})}) = s_k \text{ for all } k \in [P], \tag{65}$$

for all $i \in [P]$. This can be achieved as follows. Let $\tau > 0$ be a constant that we will take large enough. Then for any k, define the "truncated Hermite function"

$$p_{k,\tau}(x) = \mathrm{He}_k(x) m_\tau(x)\,,$$

where $m_\tau : \mathsf{R} \to [-1, 1]$ is a compactly-supported smooth function such that

$$m_\tau(x) = \begin{cases} 0, & x \notin [-\tau, \tau] \\ 1, & x \in [-\tau/2, \tau/2] \\ \in [-1, 1], & \text{otherwise} \end{cases}.$$

We order the sign vectors $\boldsymbol{s}^{(1)}, \dots, \boldsymbol{s}^{(2^P)} \in \{+1, -1\}^P$ arbitrarily. The bias $b(\boldsymbol{s}^{(i)})$ is given by $b(\boldsymbol{s}^{(i)}) = -4i\tau$. The activation function $\sigma : \mathsf{R} \to \mathsf{R}$ is given by

$$\sigma(x) = \sum_{\boldsymbol{s} 2 f+1,\ 1 g^P} \sum_{k 2 [P]} \gamma_{\boldsymbol{s},k} p_{k,\tau}(x - b(\boldsymbol{s}))\,,$$

for some choice of coefficients $\gamma_{\boldsymbol{s},k} \in \mathsf{R}$ depending only on P. This satisfies Assumption 7 because $p_{k,\tau}$ is uniformly-bounded and has uniformly-bounded first $P + 3$ derivatives. It remains to show

that we can choose the coefficients $\gamma_{\boldsymbol{s},k}$ so that (65) holds. This is because for any $\boldsymbol{s}$ we have

$$\mu_k(\sigma_{b(\boldsymbol{s})}) = \mathsf{E}_G[\mathrm{He}_k(G)\sigma(G+b(\boldsymbol{s}))] = \sum_{\boldsymbol{s}^0 \in \{+1,-1\}^P} \sum_{k^0 \in [P]} \gamma_{\boldsymbol{s}^0,k^0} A_{(\boldsymbol{s}^0,k^0),(\boldsymbol{s},k)},$$

where

$$A_{(\boldsymbol{s}^0,k^0),(\boldsymbol{s},k)} = \mathsf{E}_G[p_{k^0,\tau}(G - b(\boldsymbol{s}^0) + b(\boldsymbol{s}))\mathrm{He}_k(G)]\,.$$

And we show that $A_{(\boldsymbol{s}^0,k^0),(\boldsymbol{s},k)}$ is invertible when viewed as a $P2^P \times P2^P$ matrix. For large enough τ depending on k, the diagonal elements are lower-bounded by a constant:

$$A_{(\boldsymbol{s},k),(\boldsymbol{s},k)} = \mathsf{E}_G[\mathrm{He}_k(G)\mathrm{He}_k(G)m_\tau(G)] > 1/2\,,$$

And the off-diagonal elements are small. When $\boldsymbol{s} \neq \boldsymbol{s}^0$, for large enough τ we have

$$\begin{aligned}
|A_{(\boldsymbol{s}^0,k^0),(\boldsymbol{s},k)}| &= |\mathsf{E}_G[\mathrm{He}_{k^0}(G - b(\boldsymbol{s}^0) + b(\boldsymbol{s}))m_\tau(G - b(\boldsymbol{s}^0) + b(\boldsymbol{s}))\mathrm{He}_k(G)]| \\
&\le C \int_{\tau + b(\boldsymbol{s}^0) \;\; b(\boldsymbol{s})}^{\tau + b(\boldsymbol{s}^0) \;\; b(\boldsymbol{s})} \exp(-x^2/2)|x - b(\boldsymbol{s}^0) + b(\boldsymbol{s})|^{k^0}|x|^k dx \\
&\le C\tau \min_{s \;\; 3\tau} \exp(-s^2/2)|\tau|^{k^0}|s + 2\tau|^k dx \\
&< 1/\tau\,.
\end{aligned}$$

And similarly when $\boldsymbol{s} = \boldsymbol{s}^0$ but $k \neq k^0$, for large enough τ we have

$$\begin{aligned}
|A_{(\boldsymbol{s},k^0),(\boldsymbol{s},k)}| &= |\mathsf{E}_G[\mathrm{He}_{k^0}(G)\mathrm{He}_k(G)m_\tau(G)]| = |\mathsf{E}_G[\mathrm{He}_{k^0}(G)\mathrm{He}_k(G)(1 - m_\tau(G))]| \\
&\le |\mathsf{E}_G[|\mathrm{He}_{k^0}(G)||\mathrm{He}_k(G)|1(|G| > \tau/2)]| \le C|\mathsf{E}_G[|G|^{k^0+k}1(|G| > \tau/2)]| \\
&< 1/\tau\,.
\end{aligned}$$

So if we take large enough τ the system of equations defined by $A_{(\boldsymbol{s}^0,k^0),(\boldsymbol{s},k)}$ is invertible, so coefficients $\gamma_{\boldsymbol{s},k}$ exist such that σ satisfies (65).

Certificate. Now we provide a certificate for learning $h(\boldsymbol{z}) = z_1 \cdots z_{P_1} + z_1 \cdots z_{P_2} + \cdots + z_1 \cdots z_{P_L}$, which is a linear combination of functions of the form $\mathsf{E}_G[\sigma(\Delta\langle\boldsymbol{\delta}, \boldsymbol{z}\rangle + G + b)]$ for different $\boldsymbol{\delta} \in \{+1,-1\}^P$ and biases $b \in [-C, C]$.

The main difficulty is that we no longer have access to $\mathsf{E}_G[\sigma(\Delta\langle\boldsymbol{\delta}, \boldsymbol{z}\rangle + G)]$ for each $\boldsymbol{\delta} \in \{+1,-1\}^d$, so we have to compensate by using the biases. For each $\boldsymbol{\delta} \in \{+1,-1\}^P$, let $\boldsymbol{s} \in \{+1,-1\}^P$ be a sign vector such that $s_l \prod_{i=1}^{P_l} \delta_i > 0$ for all $l \in [L]$. Then, by Theorem 10 and the guarantee from (65) a constant fraction of neurons j after training the first layer have $\boldsymbol{w}_{j,1:P}^{\overline{T}_1} \approx \Delta\boldsymbol{\delta}$ and bias $b_j \approx b(\boldsymbol{s}) + \zeta$ for any $\zeta \in [-\Delta, \Delta]$. (Note that we can apply Theorem 10 despite its restriction that $\rho \in [-\Delta, \Delta]$, since we only care about the result holding for neurons whose bias is in $b(\boldsymbol{s}) + [-\Delta, \Delta]$ for different $\boldsymbol{s}$). So by Lemma 20, we can combine them first layer to approximate $\mathsf{E}_G[\sigma(\Delta\langle\boldsymbol{\delta}, \boldsymbol{z}\rangle + G + b(\boldsymbol{s}) + \zeta)]$ for any $\zeta \in [-\Delta, \Delta]$. By an analogous argument to Section E.1.3, we can find a measure with density ν_k with respect to $\zeta \sim \mathrm{Unif}([-\Delta, \Delta])$ that allows us to approximate

$$\int_{\Delta}^{\Delta} \mathsf{E}_G[\sigma(\Delta\langle\boldsymbol{\delta}, \boldsymbol{z}\rangle + G + b(\boldsymbol{s}) + \zeta)]\nu_k(\zeta)d\zeta = \langle\boldsymbol{\delta}, \boldsymbol{z}\rangle^k + O(\Delta)\,,$$

and where $\nu_k(\zeta)$ has second moment bounded by $1/\Delta^{Ck}$. Since we can estimate $\langle \boldsymbol{\delta}, \boldsymbol{z}\rangle^k$ to $O(\Delta)$ error for each $\boldsymbol{\delta} \in \{+1,-1\}^P$, we can approximate h via a linear combination

$$\sum_{\boldsymbol{\delta} 2 f+1,\ 1 g^P} \sum_{l=1}^{L} \left(\prod_{i=1}^{P_l} \delta_i\right) \left(\langle \boldsymbol{\delta}, \boldsymbol{z}\rangle^{P_l} + O(\Delta)\right) = h\ (\boldsymbol{z}) + O(\Delta)\,.$$

We conclude analogously to the proof of Corollary 9, using the bounded-norm certificate to obtain a generalization guarantee.

E.3. Technical result: last iterate convergence of SGD on linear models

We analyze of the last iterate for online-SGD on a linear model with ridge-regularized least-squares loss by using the well-known bias-variance decomposition Jain et al. (2017). A very similar analysis also appears in the appendix of Abbe et al. (2022b); the key difference is that we analyze online gradient descent with one sample per iteration (as opposed to online minibatch gradient descent) with a small learning rate in order to match the setting of the theorem. Compare also to Zhang (2004) which gives final-iterate bounds for the final risk, but these hold in expectation instead of with exponentially high probability.

Given an embedding of data $\phi(\boldsymbol{x}) \in \mathsf{R}^N$, consider training a linear model $\langle \boldsymbol{a}, \phi(\boldsymbol{x})\rangle$ with online-SGD. In this section, write the square loss as

$$\mathcal{L}(\boldsymbol{a}) = \frac{1}{2}\mathsf{E}_{\boldsymbol{x},y}[(y - \langle \boldsymbol{a}, \phi(\boldsymbol{x})\rangle)^2]\,.$$

For a parameter $\lambda_a > 0$, the ridge-regularized square loss is

$$\mathcal{L}_{\lambda_a}(\boldsymbol{a}) = \mathcal{L}(\boldsymbol{a}) + \frac{\lambda_a}{2}\|\boldsymbol{a}\|^2$$

Each iteration of the dynamics of online-SGD on the ridge-regulariezd square loss is is given by

$$\boldsymbol{a}^{t+1} = (1-\lambda_a)\boldsymbol{a}^t + \eta(y^t - \langle \boldsymbol{a}^t, \phi(\boldsymbol{x}^t)\rangle)\phi(\boldsymbol{x}^t)\,.$$

Lemma 25 (Analysis of online-SGD on linear model with ridge-regularized square loss) *There is a universal constant $C > 0$ such that following holds. Suppose there is $B_1 \geq 1$ such that $\|\phi(\boldsymbol{x})\| \leq B_1\sqrt{N}$ almost surely, and $|y^s| \leq B_1$ for all $0 \leq s \leq t$, and $\mathsf{E}[y^2] \leq B_1^2$, and $\lambda_a \leq N$. Then for any $\boldsymbol{a}^{cert} \in \mathsf{R}^N$*

$$\mathsf{P}\left[\mathcal{L}(\boldsymbol{a}^t) \geq \mathcal{L}_{\lambda_a}(\boldsymbol{a}^{cert}) + CB_1^2 N\left((1-\lambda_a\eta)^{2t}(\|\boldsymbol{a}^0\|^2 + \frac{B_1^2}{\lambda_a}) + \log(t/\delta)\frac{\eta B_1^6 N^2}{\lambda_a^2}\right)\right] \leq \delta\,.$$

Proof Let $\boldsymbol{a}$ be the minimizer of $\mathcal{L}_{\lambda_a}$, which is unique by strict convexity when $\lambda_a > 0$. We prove the following convergence to the optimum. For any iteration t, define the gap to optimality

$$\boldsymbol{\alpha}^t = \boldsymbol{a}^t - \boldsymbol{a}\ .$$

Defining $\boldsymbol{H} = \mathsf{E}_{\boldsymbol{x}}[\phi(\boldsymbol{x}) \otimes \phi(\boldsymbol{x})] + \lambda_a \boldsymbol{I}$ and $\boldsymbol{v} = \mathsf{E}[\phi(\boldsymbol{x})y]$, the excess loss at iteration t equals

$$\begin{aligned}\mathcal{L}_{\lambda_a}(\boldsymbol{a}^t) - \mathcal{L}_{\lambda_a}(\boldsymbol{a}\ ) &= \frac{1}{2}\langle \boldsymbol{a}^t \otimes \boldsymbol{a}^t, \boldsymbol{H}\rangle - \frac{1}{2}\langle \boldsymbol{a}\ \otimes \boldsymbol{a}\ , \boldsymbol{H}\rangle - \langle \boldsymbol{v}, \boldsymbol{a}^t - \boldsymbol{a}\ \rangle \\ &= \frac{1}{2}\langle \boldsymbol{\alpha}^t \otimes \boldsymbol{\alpha}^t, \boldsymbol{H}\rangle,\end{aligned}$$

by the first-order optimality condition $\boldsymbol{H}\boldsymbol{a}^* = \boldsymbol{v}$. So

$$\mathcal{L}_{\lambda_a}(\boldsymbol{a}^t) - \mathcal{L}_{\lambda_a}(\boldsymbol{a}^*) \le \frac{1}{2}\|\boldsymbol{\alpha}^t\|^2(\mathsf{E}_{\boldsymbol{x}}\|\phi(\boldsymbol{x})\|^2 + \lambda_a)\,. \tag{66}$$

It remains to bound $\|\boldsymbol{\alpha}^t\|$. We write the evolution of $\boldsymbol{\alpha}^t$ as:

$$\boldsymbol{\alpha}^{t+1} = \boldsymbol{P}^t\boldsymbol{\alpha}^t + \eta\boldsymbol{\zeta}^t$$

where

$$\begin{aligned}\boldsymbol{P}^t &= \boldsymbol{I} - \eta(\phi(\boldsymbol{x}^t)\otimes\phi(\boldsymbol{x}^t) + \lambda_a\boldsymbol{I})\\ \boldsymbol{\zeta}^t &= y^t\phi(\boldsymbol{x}^t) - (\phi(\boldsymbol{x}^t)\otimes\phi(\boldsymbol{x}^t) + \lambda_a\boldsymbol{I})\boldsymbol{a}^t\,.\end{aligned}$$

Inductively, one obtains the well-known "bias-variance" decomposition

$$\boldsymbol{\alpha}^t = \underbrace{\Big(\prod_{l=t-1}^{0}\boldsymbol{P}^l\Big)\boldsymbol{\alpha}^0}_{\text{(Bias term)}} + \underbrace{\eta\sum_{j=0}^{t-1}\Big(\boldsymbol{P}^{t-1}\cdots\boldsymbol{P}^{j+1}\Big)\boldsymbol{\zeta}^j}_{\text{(Variance term)}}\,.$$

Notice that (a) $\mathbf{0} \preceq \boldsymbol{P}^l \preceq (1-\eta\lambda_a)\boldsymbol{I}$, and (b) $\mathcal{L}_{\lambda_a}(\boldsymbol{a}) \ge \frac{\lambda_a}{2}\|\boldsymbol{a}\|^2$ and $\mathcal{L}_{\lambda_a}(\mathbf{0}) = \frac{1}{2}\mathsf{E}[y^2]$, so

$$\|\text{(Bias term)}\| \overset{(a)}{\le} (1-\eta\lambda_a)^t\|\boldsymbol{\alpha}^0\| \overset{(b)}{\le} (1-\eta\lambda_a)^t(\|\boldsymbol{a}^0\| + \sqrt{\mathsf{E}[y^2]/\lambda_a})\,. \tag{67}$$

To bound the variance term, define the norm squared of the variance term:

$$m_t = \eta^2\Big\|\sum_{j=0}^{t-1}\big(\boldsymbol{P}^{t-1}\cdots\boldsymbol{P}^{j+1}\big)\,\boldsymbol{\zeta}^j\Big\|^2.$$

Also, for any time t, define $\tilde{m}_0 = 0$ and

$$\tilde{m}_{t+1} = \eta^2\|\boldsymbol{\zeta}^t\|^2 + 2\eta^2\Big\langle\boldsymbol{\zeta}^t, \boldsymbol{P}^t\sum_{j=0}^{t-1}(\boldsymbol{P}^{t-1}\cdots\boldsymbol{P}^{j+1})\boldsymbol{\zeta}^j\Big\rangle + (1-\eta\lambda_a)^2\tilde{m}_t.$$

By induction on t, we can show that $\tilde{m}_t \ge m_t$ at all times t. The base case is clear since $m_0 = \tilde{m}_0 = 0$. The inductive step is:

$$\begin{aligned}m_{t+1} &= \eta^2\Big\|\boldsymbol{\zeta}^t + \boldsymbol{P}^t\sum_{j=0}^{t-1}(\boldsymbol{P}^{t-1}\cdots\boldsymbol{P}^{j+1})\boldsymbol{\zeta}^j\Big\|^2\\ &\le \eta^2\|\boldsymbol{\zeta}^t\|^2 + 2\eta^2\Big\langle\boldsymbol{\zeta}^t, \boldsymbol{P}^t\sum_{j=0}^{t-1}(\boldsymbol{P}^{t-1}\cdots\boldsymbol{P}^{j+1})\boldsymbol{\zeta}^j\Big\rangle + (1-\eta\lambda_a)^2m_t^2\\ &\le \tilde{m}_{t+1}\,,\end{aligned}$$

where we use the inductive hypothesis $\tilde{m}_t \ge m_t$.

The reason we study $\tilde{m}_t$ instead of m_t is because it satisfies these bounded differences:

$$|\tilde{m}_{t+1} - (1-\eta\lambda_a)^2\tilde{m}_t| \le \eta^2\|\boldsymbol{\zeta}^t\|^2 + 2\eta\|\boldsymbol{\zeta}^t\|(1-\eta\lambda_a)\sqrt{\tilde{m}_t}\,. \tag{68}$$

Furthermore, let $\mathcal{F}_t = \sigma\big(\{\boldsymbol{x}^s, y^s\}_{s \le t}\big)$ be the history until time t. Since $\mathsf{E}[\boldsymbol{\zeta}^t \mid \mathcal{F}_{t-1}] = \mathsf{E}[\boldsymbol{\zeta}^t \mid \mathcal{F}_{t-1}] = \boldsymbol{v} - \boldsymbol{H}\boldsymbol{a}_{t-1} = \boldsymbol{0}$,

$$\mathsf{E}[\tilde{m}_{t+1} \mid \mathcal{F}_t] = \eta^2\mathsf{E}[\|\boldsymbol{\zeta}^t\|^2] + (1-\eta\lambda_a)^2\tilde{m}_t\,. \tag{69}$$

So the martingale concentration bound in Lemma 26 applied to $\tilde{m}_t$ and using (68) and (69) with $c = (1-\eta\lambda_a)^2$, $a = \eta^2\mathsf{E}[\|\boldsymbol{\zeta}^0\|^2]$ and $M = \max_{0 \le t^0 \le t-1}\eta^2\|\boldsymbol{\zeta}^{t^0}\|^2$, yields, for any $\varepsilon \ge \max(M/c, a^2/Mc)$, and some large enough universal constant $C > 0$,

$$\mathsf{P}\Big[\tilde{m}_t \ge \frac{a}{1-c} + \varepsilon\Big] \le t\exp\Big(-\frac{\varepsilon(1-c^2)}{CM}\Big)\,.$$

By applying $\|\boldsymbol{a}_{t-1}\| \le \sqrt{\mathsf{E}[y^2]/\lambda_a}$ and triangle inequalities, we have

$$\begin{gathered} M \lesssim \eta^2(B_1^4 N + ((B_1^3 N/\sqrt{\lambda_a} + \sqrt{\lambda_a}B_1)^2)) \lesssim \eta^2 B_1^6 N^2/\lambda_a\,, \\ a \lesssim \eta^2 B_1^6 N^2/\lambda_a\,, \\ \eta\lambda_a \le 1-c \le 1-c^2 \le 4\eta\lambda_a \le 4\eta N\,. \end{gathered}$$

Plug this in and simplify,

$$\mathsf{P}\Big[\tilde{m}_t \ge C\frac{\eta B_1^6 N^2}{\lambda_a^2} + \varepsilon\Big] \le t\exp\Big(-\frac{\varepsilon\lambda_a^2}{C\eta B_1^6 N^2}\Big)\,,$$

for all $\varepsilon > 0$. So using $\tilde{m}_t \ge m_t = \|(\text{Variance term})\|^2$, there is a universal constant C such that for any $0 < \delta < 1/2$,

$$\mathsf{P}\Big[\|(\text{Variance term})\|^2 \ge C\log(t/\delta)\frac{\eta B_1^6 N^2}{\lambda_a^2}\Big] \le \delta\,. \tag{70}$$

So combining (67) and (70) with (66),

$$\mathsf{P}\left[\mathcal{L}_{\lambda_a}(\boldsymbol{a}^t) - \mathcal{L}_{\lambda_a}(\boldsymbol{a}_{\text{cert}}) \ge CB_1^2 N\left((1-\lambda_a\eta)^{2t}(\|\boldsymbol{a}^0\|^2 + \frac{B_1^2}{\lambda_a}) + \log(t/\delta)\frac{\eta B_1^6 N^2}{\lambda_a^2}\right)\right] \le \delta\,.$$

The lemma follows by plugging in the expression for $\mathcal{L}_{\lambda_a}(\boldsymbol{a}^{cert})$ and using that $\boldsymbol{a}_{\text{cert}}$ is optimal, so $\mathcal{L}_{\lambda_a}(\boldsymbol{a}_{\text{cert}}) \le \mathcal{L}_{\lambda_a}(\boldsymbol{a})$. ∎

Lemma 26 (Martingale high-probability bound) *There is constant $C > 0$ such the the following holds. Suppose that $X_0, \dots, X_t, \dots$ are nonnegative random variables and are such that $X_0 = 0$, and $\mathsf{E}[X_{t+1} \mid \mathcal{F}_t] \le a + cX_t$ and almost surely $|X_{t+1} - cX_t| \le M + 2\sqrt{cMX_t}$ for constants $M, a \ge 0$ and $0 < c < 1$. Then for any t and $\varepsilon \ge \max(M/c, a^2/Mc)$,*

$$\mathsf{P}\Big[X_t \ge \frac{a}{1-c} + \varepsilon\Big] \le t\exp\Big(-\frac{\varepsilon(1-c^2)}{CM}\Big)\,.$$

Proof Construct $Z_t = c^{\ t}(X_t - \frac{a}{1-c})$. Then Z_t is a super-martingale:

$$\mathsf{E}[Z_{t+1} \mid \mathcal{F}_t] \le c^{\ t-1}a + c^{\ t}X_t - \frac{c^{\ t-1}a}{1-c} = Z_t + ac^{\ t-1}\Big(1 - \frac{1}{1-c} + \frac{c}{1-c}\Big) \le Z_t\,.$$

Let $\tau = \inf\{t \ge 0 : X_t \ge \iota\}$ be a stopping time for some $\iota > a/(1-c)$. Then $\tilde{Z}_t = Z_{\min(t,\tau)}$ is also a super-martingale. Furthermore, we have the bounded differences:

$$\begin{aligned}|\tilde{Z}_{t+1} - \tilde{Z}_t| &\le |c^{\ t-1}(X_{t+1} - cX_t - a)| \\ &\le c^{\ t-1}|M + 2\sqrt{cMX_t} + a| \\ &\le c^{\ t-1}(M + a + 2\sqrt{cM\iota}) =: c^{\ t-1}\tilde{M},\end{aligned}$$

if $t < \tau$ and $|\tilde{Z}_{t+1} - \tilde{Z}_t| = 0$ if $t \ge \tau$.

So by the Azuma-Hoeffding inequality, since $Z_0 \le 0$,

$$\mathsf{P}[\tilde{Z}_t \ge \varepsilon] \le \exp\Big(-\varepsilon^2/(2\sum_{j=1}^{t} c^{\ 2j}\tilde{M}^2)\Big) \le \exp\Big(-\frac{1}{2}(\varepsilon/\tilde{M})^2 c^{2t}(1-c^2)\Big)\,.$$

Let E be the event that $\tilde{Z}_{t^0} < c^{\ t^0}(\iota - (a/(1-c))$ for all $t^0 \in \{0,\dots,t\}$. By a union bound,

$$\mathsf{P}[E] \ge 1 - t\exp\Big(-\frac{(\iota - (a/(1-c)))^2(1-c^2)}{2\tilde{M}^2}\Big) \ge 1 - t\exp\Big(-\frac{\iota^2(1-c^2)}{2\tilde{M}^2}\Big)\,.$$

Finally, note that under event E we have $\tilde{Z}_t = Z_t$, and $X_t < \frac{1}{1-c} + \iota$. And for $\iota \ge \max(M/c, a^2/Mc)$ we have $\tilde{M}^2 \le 16cM\iota$. ∎

Appendix F. Lower bounds for linear methods and CSQ methods

F.1. Linear methods

We define our general linear methods as follows (see for example (Abbe et al., 2022b, Appendix H) for additional discussion). Fix a Hilbert space $(\mathcal{H}, \langle\cdot,\cdot\rangle_{\mathcal{H}})$ and a feature map $\psi : \mathsf{R}^d \to \mathcal{H}$. Given data points $(y_i, \boldsymbol{x}_i)_{i\in[n]}$, the linear method constructs weights $\hat{\boldsymbol{a}} \in \mathcal{H}$ by minimizing the regularized empirical risk for some loss function $L : \mathsf{R}^{2n} \to \mathsf{R} \cup \{\infty\}$ and some regularization parameter $\lambda > 0$,

$$\hat{\boldsymbol{a}} = \arg\min_{\boldsymbol{a}\in\mathcal{H}} \{L((y_i, \langle \boldsymbol{a}, \psi(\boldsymbol{x}_i)\rangle)_{i\in[n]}) + \lambda \|\boldsymbol{a}\|_{\mathcal{H}}^2\},$$

and estimates the target function using the linear prediction model

$$\hat{f}(\boldsymbol{x}) = \langle \hat{\boldsymbol{a}}, \psi(\boldsymbol{x})\rangle\,.$$

The takeaway of this section is that to learn any degree-D functions with small support on isotropic data, linear methods must pay at least $\Omega(d^D)$ samples (and "width" $\dim(\mathcal{H}) \geq d^D$) when the support is not known. This is proved by Abbe et al. (2022b) in the case of the binary hypercube:

Proposition 27 (Limitations for linear methods on hypercube, cf. Proposition 11 of Abbe et al. (2022b)) *Let $h : \{+1,-1\}^P \to \mathsf{R}$ be a function given by*

$$h(\boldsymbol{z}) = \sum_{S\subseteq[P]} \hat{h}(S) \prod_{i\in S} z_i\,.$$

Let $D = \max\{|S| : \hat{h}(S) \neq 0\}$ be the degree of h. Consider the class of functions which depend as h on some subset of coordinates

$$\mathcal{F} = \cup_{\sigma\in S_d}\{f_{\sigma} : \{+1,-1\}^d \to \mathsf{R},\ \text{where } f_{\sigma}(\boldsymbol{x}) = h(x_{\sigma(1)},\dots,x_{\sigma(P)})\}.$$

For any linear method, let $\hat{f}_\sigma$ be the function estimated by the linear method on (possibly noisy) samples $(\boldsymbol{x}_i, f_{\sigma}(\boldsymbol{x}_i) + \epsilon_i)_{i\in[n]}$. Then there are constants $C_h, c_h > 0$ such that

$$\frac{1}{|S_d|}\sum_{\sigma\in S_d} \mathsf{E}_{\boldsymbol{x}\sim \mathsf{Unif}(\{+1,-1\}^d)}[(f_{\sigma}(\boldsymbol{x}) - \hat{f}_\sigma(\boldsymbol{x}))^2] \geq c_h - C_h \min(n, \dim(\mathcal{H}))d^{-D}\,.$$

Proof Apply Proposition 11 of Abbe et al. (2022b), letting Ω be the subspace of $f \in L^2(\{+1,-1\}^d)$ that are degree-D homogeneous. Then $\max_\sigma \frac{1}{|S_d|}\sum_{\sigma'\in S_d} |\mathsf{E}[f_{\sigma}(\boldsymbol{x})\mathsf{P}_\Omega f_{\sigma'}(\boldsymbol{x})]| \leq O(d^{-D})$. ∎

We now give an analogous result for the Gaussian data distribution, where the degree also drives the complexity for linear methods. This bound is new and was not derived in Abbe et al. (2022b).

Proposition 28 (Limitations for linear methods on Gaussian data) *Let $h : \mathsf{R}^P \to \mathsf{R}$ be a function given by*

$$h(\boldsymbol{z}) = \sum_{S=(k_1,\dots,k_P)\in\mathsf{N}^P} \hat{h}(S) \prod_{i\in[P]} \mathrm{He}_{k_i}(z_i)\,.$$

Let $D = \max\{\sum_i k_i : \hat{h}\ (S) \neq 0\}$ *be the degree of* h *. Consider the class of functions which depend as* h *on some subspace of coordinates*

$$\mathcal{F} = \bigcup_{\substack{\boldsymbol{M} 2 \mathsf{R}^{P\ \ d} \\ \boldsymbol{M}\boldsymbol{M}^{>} = \boldsymbol{I}}} \{f_{\ ,\boldsymbol{M}} : \mathsf{R}^d \to \mathsf{R},\ \textit{where}\ f_{\ ,\boldsymbol{M}}(\boldsymbol{x}) = h\ (\boldsymbol{M}\boldsymbol{x})\}.$$

For any linear method, let $\hat{f}_{\boldsymbol{M}}$ *be the function estimated by the linear method on (possibly noisy) samples* $(\boldsymbol{x}_i, f_{\ ,\boldsymbol{M}}(\boldsymbol{x}_i) + \epsilon_i)_{i 2 [n]}$*. Then there are constants* $C_h\ , c_h\ > 0$ *such that with respect to a uniformly random* $\boldsymbol{M} \sim \mathsf{R}^{P\ \ d}$*, satisfying* $\boldsymbol{M}\boldsymbol{M}^{>} = \boldsymbol{I}$*, we have*

$$\mathsf{E}_{\boldsymbol{M}}[\mathsf{E}_{\boldsymbol{x}\ \ \mathsf{N}(0,I_d)}[(f_{\ ,\boldsymbol{M}}(\boldsymbol{x}) - \hat{f}_{\boldsymbol{M}}(\boldsymbol{x}))^2]] \geq c_h\ \ - C_h\ \ \min(n, \dim(\mathcal{H}))d^{\ \ D}.$$

Proof First, we we can write a degree-D monomial as a linear combination of functions in $\mathcal{F}$.

Claim 29 *There are semiorthogonal matrices* $\boldsymbol{M}^1, \ldots, \boldsymbol{M}^{2^D}$ *and coefficients* $b_1, \ldots, b_{2^D}$ *such that*

$$\prod_{i 2 [D]} x_i = \sum_{j=1}^{2^D} b_j h\ (\boldsymbol{M}^j \boldsymbol{x})\,.$$

Furthermore, for all j *we have* $|a_j| \leq C_h$ *, which is a constant depending only on* h *.*

Proof [Proof of claim]

Let $S = (k_1, \ldots, k_P)$ such that $\hat{h}\ (S) \neq 0$ and $\sum_i k_i = D$. Define the prefix sums $s_i = \sum_{i^0 < i} k_i$. Then for each $\boldsymbol{\delta} \in \{+1, -1\}^D$, let $\boldsymbol{R}^{\delta} \in \mathsf{R}^{P\ \ d}$ be the matrix which for any $i \in [P]$ satisfies

$$\boldsymbol{R}^{\boldsymbol{\delta}}_{i,:} = \frac{1}{\sqrt{k_i}} \sum_{j=1}^{k_i} \delta_{s_i + j} \boldsymbol{e}_{s_i + j}.$$

Notice that $\boldsymbol{R}^{\boldsymbol{\delta}}(\boldsymbol{R}^{\boldsymbol{\delta}})^{>} = \boldsymbol{I}$, so this is a valid semi-orthogonal matrix, and so $h\ (\boldsymbol{R}^{\boldsymbol{\delta}}\boldsymbol{x}) \in \mathcal{F}$. Now let us show that we can write the monomial as a linear combination of functions of the form $h\ (\boldsymbol{R}^{\boldsymbol{\delta}}\boldsymbol{x})$. Specifically, for any $S^0 = (k_1^0, \ldots, k_P^0)$ with $\sum_i k_i^0 \leq D$ we have

$$\begin{aligned}
\mathsf{E}_{\boldsymbol{\delta}\ \ f\ \ 1g^D} \left[\left(\prod_{j 2 [D]} \delta_j \right) \left(\prod_{i 2 [P]} \mathrm{He}_{k_i^0}((\boldsymbol{R}^{\boldsymbol{\delta}}\boldsymbol{x})_i) \right) \right] &= \prod_{i 2 [P]} \mathsf{E}_{\boldsymbol{\delta}\ \ f\ \ 1g^{k_i}} \left[\left(\prod_{j 2 [k_i]} \delta_j \right) \mathrm{He}_{k_i^0}(\frac{1}{\sqrt{k_i}} \sum_{j=1}^{k_i} \delta_j x_{j+s_i}) \right] \\
&\propto \prod_{i 2 [P]} \begin{cases} 0, & k_i^0 < k_i \\ \prod_{j=1}^{k_i} x_{j+s_i}, & k_i^0 = k_i \\ \text{something else}, & k_i^0 > k_i \end{cases} \\
&\propto 1(S = S^0) \prod_{j 2 [D]} x_j\,, \qquad (71)
\end{aligned}$$

with a nonzero proportionality constant that only depends on S. Therefore,

$$\sum_{\boldsymbol{\delta} 2 f\ \ 1g} (\prod_{i=1}^{D} \delta_i) h\ (\boldsymbol{R}^{\boldsymbol{\delta}}\boldsymbol{x}) \propto \prod_{i 2 [D]} x_i\,,$$

with a nonzero proportionality constant that only depends on h . This proves the claim. ■

We will use this claim to lower-bound the error of the linear method on $\mathcal{F}$. Notice that the linear method must predict $\langle \hat{\boldsymbol{a}}, \psi(\boldsymbol{x})\rangle$, where $\hat{\boldsymbol{a}} \in \mathsf{span}\{\psi(\boldsymbol{x}_i)\}_{i \in [n]}$. So the error is lower-bounded by the norm of the orthogonal projection to this subspace. For $\boldsymbol{x} \sim \mathsf{N}(0, I_d)$ throughout,

$$\mathsf{E}_{\boldsymbol{M}}[\mathsf{E}_{\boldsymbol{x}}[(h\ (\boldsymbol{M}\boldsymbol{x}) - \hat{f}_{\boldsymbol{M}}(\boldsymbol{x}))^2]] \geq \mathsf{E}_{\boldsymbol{M}}[\min_{\boldsymbol{a} \in \mathsf{span}\{\psi(\boldsymbol{x}_i)\}_{i \in [n]}} \mathsf{E}_{\boldsymbol{x}}[(h\ (\boldsymbol{M}\boldsymbol{x}) - \langle \boldsymbol{a}, \psi(\boldsymbol{x})\rangle)^2]] = (*)\,.$$

Now let $\boldsymbol{M}^1, \dots, \boldsymbol{M}^{2^D}$ and $b_1, \dots, b_{2^D}$ be the matrices and coefficients from the claim. Let $\boldsymbol{R} \in \mathsf{R}^{d\ \ d}$ be a uniformly random rotation and let σ be a uniformly random permutation. Since $\boldsymbol{M}^i \sigma \boldsymbol{R}$ has the same distribution as $\boldsymbol{M}$,

$$\begin{aligned}
(*) &= \mathsf{E}_{\boldsymbol{M}}[\min_{\boldsymbol{a} \in \mathsf{span}\{\psi(\boldsymbol{x}_i)\}_{i \in [n]}} \mathsf{E}_{\boldsymbol{x}}[(h\ (\boldsymbol{M}\boldsymbol{x}) - \langle \boldsymbol{a}, \psi(\boldsymbol{x})\rangle)^2]] \\
&\geq \frac{1}{2^D(\sum_{i=1}^{2^D} b_i^2)} \mathsf{E}_{\boldsymbol{R},\sigma}[\min_{\boldsymbol{a} \in \mathsf{span}\{\psi(\boldsymbol{x}_i)\}_{i \in [n]}} \mathsf{E}_{\boldsymbol{x}}[(\sum_{i=1}^{2^D} b_i h\ (\boldsymbol{M}^i \sigma \boldsymbol{R}) - \langle \boldsymbol{a}, \psi(\boldsymbol{x})\rangle)^2]] \\
&= \frac{1}{2^D(\sum_{i=1}^{2^D} b_i^2)} \mathsf{E}_{\boldsymbol{R},\sigma}[\min_{\boldsymbol{a} \in \mathsf{span}\{\psi(\boldsymbol{x}_i)\}_{i \in [n]}} \mathsf{E}_{\boldsymbol{x}}[(\prod_{i \in [D]} (\boldsymbol{R}\boldsymbol{x})_{\sigma(i)} - \langle \boldsymbol{a}, \psi(\boldsymbol{x})\rangle)^2]] = (**)\,.
\end{aligned}$$

However, Proposition 11 of Abbe et al. (2022b) provides a lower-bound on the error for learning the class $\cup_{\sigma \in S_d}\{\prod_{i \in [D]} (\boldsymbol{R}\boldsymbol{x})_{\sigma(i)}\}$ with a linear method. Specifically, for two permutations σ, σ^0 such that $\sigma([D]) \neq \sigma^0([D])$, we have $\mathsf{E}_{\boldsymbol{x}}[\prod_{i \in [D]} (\boldsymbol{R}\boldsymbol{x})_{\sigma(i)} (\boldsymbol{R}\boldsymbol{x})_{\sigma^0(i)}] = \mathsf{E}_{\boldsymbol{x}}[\prod_{i \in [D]} x_{\sigma(i)} x_{\sigma^0(i)}] = 0$. So Proposition 11 of Abbe et al. (2022b) implies that there is some constant C depending only on D, such that

$$(**) \geq c_h\ (1 - C \min(n, \dim(\mathcal{H}))d^{\ \ D})\,.$$

Putting together the equations proves the lemma.

■

F.2. Correlational Statistical Query (CSQ) methods

A Correlational Statistical Query (CSQ) algorithm (Ben-David et al., 1995; Bshouty and Feldman, 2002; Reyzin, 2020) accesses the data via queries $\phi : \mathsf{R}^d \to [-1, 1]$ and returns $\mathsf{E}_{\boldsymbol{x},y}[\phi(\boldsymbol{x})y]$ up to some error tolerance τ. In our case, since $y = f\ (\boldsymbol{x}) + \varepsilon$, where ε is independent zero-mean noise, the query returns a value in $\mathsf{E}_{\boldsymbol{x}}[\phi(\boldsymbol{x}) f\ (\boldsymbol{x})] + [-\tau, +\tau]$. The CSQ algorithm outputs a guess $\hat{f}$ of the true function f . An example of a CSQ algorithm is gradient descent on the population square loss if we inject noise in the gradients (see, e.g., Abbe and Boix-Adsera (2022)).

First, we give a lower bound on the CSQ complexity of learning a function with leaps when $\boldsymbol{x}$ is drawn uniformly from the hypercube. The below lower bound is qualitatively similar to the argument in Abbe and Boix-Adsera (2022) based on the "alignment" quantity. The bounds of Abbe and Boix-Adsera (2022) have tighter constants in the exponents of the bound, but they have the disadvantage that they apply only to noisy population gradient descent instead of to general CSQ algorithms.

Proposition 30 (Limitations for CSQ algorithms on hypercube) *Let* $h\ : \{+1,-1\}^P \to \mathsf{R}$, *and let* $\mathrm{Leap}(h\ )$ *be its leap. Consider the class of functions given by applying* $h\ $ *on some subset of coordinates*

$$\mathcal{F} = \cup_{\sigma 2 S_d}\{f\ _{,\sigma} : \{+1,-1\}^d \to \mathsf{R},\ \textit{where}\ f\ _{,\sigma}(\boldsymbol{x}) = h\ (x_{\sigma(1)},\dots,x_{\sigma(P)})\}\,.$$

Then a CSQ algorithm with n *queries of error tolerance* τ *outputs* $\hat{f}$ *such that with probability* $\geq 1 - C_h\ nd^{\ \mathrm{Leap}(h\ )}/\tau^2$ *over the random choice of* $f\ \sim \mathcal{F}$,

$$\mathsf{E}_{\boldsymbol{x}\ \ f+1,\ \ 1g^d}[(f\ (\boldsymbol{x}) - \hat{f})^2] \geq c_h\ \ > 0\,.$$

Proof For any subset T, define $\mathcal{S}^{6\,T} = \{S \subseteq [P] : S \not\subseteq T, \hat{h}\ (S) \neq 0\}$. By definition of the leap, there is a subset $T \subseteq [P]$ such that $\mathcal{S}^{6\,T} \neq \emptyset$ and for all $S \in \mathcal{S}^{6\,T}$ we have $|S \setminus T| \geq \mathrm{Leap}(h\ )$. Without loss of generality, assume $T = \{1,\dots,k\} \subseteq [P]$. Write

$$h\ (\boldsymbol{z}) = h\ _T(\boldsymbol{z}) + h_{6\,T}(\boldsymbol{z})\,,$$

where

$$h\ _T(\boldsymbol{z}) = \sum_{S\ \ T} \hat{h}\ (S)\prod_{i2S} z_i,\ \text{and}\ h_{6\,T}(\boldsymbol{z}) = \sum_{S6\,T} \hat{h}\ (S)\prod_{i2S} z_i\,.$$

Suppose that the CSQ algorithm knows $\sigma(1),\dots,\sigma(k)$, which can only help it. Then the problem of learning $f\ _{,\sigma}$ from CSQ queries is equivalent to the problem of learning $f_{6\,T,\sigma}(\boldsymbol{x}) = h_{6\,T}(x_{\sigma(1)},\dots,x_{\sigma(P)})$ from CSQ queries. However, for random permutations $\sigma^{\emptyset}$ conditioned on $\sigma^{\emptyset}(1) = \sigma(1),\dots,\sigma^{\emptyset}(k) = \sigma(k)$ we have

$$\begin{aligned}\mathcal{C} &= \sup_{\phi 2 L^2(f+1,\ \ 1g^d),k\phi k^2=1} \mathsf{E}_{\sigma^{\emptyset}}[\langle f_{6\,T,\sigma^{\emptyset}},\phi\rangle^2] \leq \mathsf{E}_{\sigma^{\emptyset}}\left[\left(\sum_{S2S^{6\,T}} \hat{\phi}(\sigma^{\emptyset}(S))\hat{h}\ (S)\right)^2\right]\\ &\leq C_h\ \max_{S2S^{6\,T}} \mathsf{E}_{\sigma^{\emptyset}}[\hat{\phi}(\sigma^{\emptyset}(S))^2] \leq C_h\ \binom{d-k}{\mathrm{Leap}(h\ )} \leq C_h\ d^{\ \mathrm{Leap}(h\ )}.\end{aligned}$$

So by a union bound, with probability $\geq 1 - \mathcal{C}n/\tau^2$ all n first CSQ queries can return 0. The final output $\hat{f}$ of the algorithm can also be viewed as a statistical query. So with probability at least $1 - n\mathcal{C}/\tau^2 - \mathcal{C}/\varepsilon^2$,

$$\begin{aligned}\mathsf{E}_{\boldsymbol{x}}[(f\ _{,\sigma}(\boldsymbol{x}) - \hat{f}(\boldsymbol{x}))^2] &= \|f\ _{,\sigma} - f\ _{T,\sigma}\|^2 + \|\hat{f} - f\ _{T,\sigma}\|^2 - 2\langle f\ _{,\sigma} - f\ _{T,\sigma}, \hat{f} - f\ _{T,\sigma}\rangle\\ &\geq \|f_{6\,T,\sigma}\|^2 - (\|\mathsf{E}_{\sigma^{\emptyset}}[f_{6\,T,\sigma^{\emptyset}}]\| + \varepsilon)^2\,.\end{aligned}$$

The proposition follows by letting ε be a small enough positive constant depending on $h\ $. ■

Proposition 31 (Limitations for CSQ algorithms on Gaussian data) *Let* $h\ : \mathsf{R}^P \to \mathsf{R}$ *be a polynomial of finite degree* D. *Let*

$$\mathrm{isoLeap}(h\ ) = \max_{R2O_P} \mathrm{Leap}(h\ ,R)$$

be its isotropic leap (as defined in Appendix B.2). Consider the class of functions which given by applying h on some subspace of coordinates

$$\mathcal{F} = \bigcup_{\substack{\boldsymbol{M} \in \mathbb{R}^{P \times d} \\ \boldsymbol{M}\boldsymbol{M}^\top = \boldsymbol{I}}} \{f_{\boldsymbol{M}} : \mathbb{R}^d \to \mathbb{R}, \text{ where } f_\sigma(\boldsymbol{x}) = h(\boldsymbol{M}\boldsymbol{x})\}.$$

Then, for any CSQ algorithm with n queries of tolerance $\pm\tau$, with probability $1 - \frac{C_h n}{\tau^2} d^{\mathrm{isoLeap}(h)/2}$ over the random choice of $\boldsymbol{M}$, the estimator $\hat{f}$ it returns for $f \in \mathcal{F}$ satisfies

$$\mathbb{E}_{\boldsymbol{x} \sim \mathsf{N}(0, I_d)}[(f(\boldsymbol{x}) - \hat{f}(\boldsymbol{x}))^2] \geq c_h > 0\,.$$

Proof The proof follows a similar strategy to that of the previous proposition. Without loss of generality, suppose that $\mathrm{Leap}(h) = \mathrm{isoLeap}(h)$ (in other words, that we are already in a basis that maximizes the leap without having to apply a rotation). Then there must be $T \subseteq [P]$ which we can take to be $T = \{1, \dots, r\}$ without loss of generality such that if we define

$$h_T(\boldsymbol{z}) = \sum_{S = (k_1, \dots, k_r) \in \mathbb{N}^r} \hat{h}(S) \prod_{i \in [P]} \mathrm{He}_{k_i}(z_i) \text{ and } h_{\setminus T}(\boldsymbol{z}) = h(\boldsymbol{z}) - h_T(\boldsymbol{z})\,,$$

we must have $h_{\setminus T} \not\equiv 0$, and for each nonzero Fourier coefficient $S = (k_1, \dots, k_P)$ such that $\hat{h}_{\setminus T}(S) \neq 0$ we must have $\sum_{i \in [P] \setminus T} k_i \geq \mathrm{Leap}(h)$. Knowing the first r rows $\boldsymbol{M}_{1,:}, \dots, \boldsymbol{M}_{r,:}$ can only help the CSQ algorithm, so we just have to show that over the choice of random $\boldsymbol{M}^\emptyset \in \mathbb{R}^{P \times d}$ conditioned on $\boldsymbol{M}^\emptyset_{i,:} = \boldsymbol{M}_{i,:}$ for $i \in [r]$ we have

$$\mathcal{C} = \sup_{\phi \in L^2(\mathsf{N}(0, I_d)), \|\phi\|_2 = 1} \mathbb{E}_{\boldsymbol{M}^\emptyset}[\mathbb{E}_{\boldsymbol{x} \sim \mathsf{N}(0, I_d)}[h_{\setminus T}(\boldsymbol{M}^\emptyset \boldsymbol{x})\phi(\boldsymbol{x})]^2] \leq C_h\, d^{\mathrm{Leap}(h)/2}\,.$$

For ease of notation, we consider the case when $T = \emptyset$ since the general case is analogous. This follows from the following claim, where for any $\beta \in \mathbb{N}^d$ be define $\mathrm{He}_\beta(\boldsymbol{x}) = \prod_{i=1}^d \mathrm{He}_{\beta_i}(x_i)$.

Claim 32 *Let $\alpha \in \mathbb{N}^d$ be such that with $\alpha_i = 0$ for all $i > P$. Let $\boldsymbol{R} \in \mathbb{R}^{d \times d}$ be a random rotation drawn according to the Haar measure. Then*

$$\sup_{\phi \in L^2(\mathsf{N}(0, \mathbf{I}_d)), \|\phi\|_2 = 1} \mathbb{E}_{\boldsymbol{R}}[\mathbb{E}_{\boldsymbol{x} \sim \mathsf{N}(0, I_d)}[\mathrm{He}_\alpha(\boldsymbol{R}\boldsymbol{x})\phi(\boldsymbol{x})]^2] = O(d^{\|\alpha\|_1/2})\,.$$

Proof Write $\phi(\boldsymbol{x}) = \sum_{\beta \in \mathbb{N}^d} \hat{\phi}(\beta)\mathrm{He}_\beta(\boldsymbol{x})$. By integration by parts,

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{x} \sim \mathsf{N}(0, I_d)}[\mathrm{He}_\alpha(\boldsymbol{R}\boldsymbol{x})\phi(\boldsymbol{x})] \\
&= \mathbb{E}_{\boldsymbol{x}_{\setminus 1}}\left[\prod_{i \neq 1} \mathrm{He}_{\alpha_i}(x_i)\mathbb{E}_{x_1}[\mathrm{He}_{\alpha_1}(x_1)\phi(\boldsymbol{R}^\top \boldsymbol{x})]\right] \\
&= \mathbb{E}_{\boldsymbol{x}_{\setminus 1}}\left[\prod_{i \neq 1} \mathrm{He}_{\alpha_i}(x_i)\mathbb{E}_{x_1}\left[\frac{\partial^{\alpha_1}}{\partial x_1^{\alpha_1}}\phi(\boldsymbol{R}^\top \boldsymbol{x})\right]\right]
\end{aligned}$$

$$= \mathsf{E}_{\boldsymbol{x}} \left[\prod_{i=1}^{d} \mathrm{He}_{\alpha_i}(x_i) \mathsf{E}_{x_1} \left[\sum_{j_1,\ldots,j_{\alpha_1} \in [d]} \left(\prod_{k=1}^{\alpha_1} R_{1,j_k} \frac{\partial}{\partial z_{j_k}} \right) \phi(\boldsymbol{z}) \mid_{\boldsymbol{z}=\boldsymbol{R}^\top \boldsymbol{x}} \right] \right]$$

$$= \mathsf{E}_{\boldsymbol{x}} \left[\sum_{j_{1,1},\ldots,j_{1,\alpha_1} \in [d]} \cdots \sum_{j_{P,1},\ldots,j_{P,\alpha_P} \in [d]} \left(\prod_{l=1}^{\alpha_1} R_{1,j_{1,l}} \frac{\partial}{\partial z_{j_{1,l}}} \right) \cdots \left(\prod_{l=1}^{\alpha_P} R_{P,j_{P,l}} \frac{\partial}{\partial z_{j_{P,l}}} \right) \phi(\boldsymbol{z}) \mid_{\boldsymbol{z}=\boldsymbol{R}^\top \boldsymbol{x}} \right]$$

$$= \sum_{\substack{\Upsilon \in \mathbb{Z}^{P}_{0}{}^{d} \\ \Upsilon \mathbf{1} = \alpha_{1:P}}} \mathsf{E}_{\boldsymbol{x}} \left[\left(\prod_{i \in [P], j \in [d]} R_{i,j}^{\Upsilon_{i,j}} (\Upsilon_{i,j}!) \frac{\partial^{\Upsilon_{i,j}}}{\partial z_j^{\Upsilon_{i,j}}} \right) \phi(\boldsymbol{z}) \mid_{\boldsymbol{z}=\boldsymbol{R}^\top \boldsymbol{x}} \right]$$

$$= \sum_{\substack{\Upsilon \in \mathbb{Z}^{P}_{0}{}^{d} \\ \Upsilon \mathbf{1} = \alpha_{1:P}}} \mathsf{E}_{\boldsymbol{x}} \left[\left(\prod_{i \in [P], j \in [d]} R_{i,j}^{\Upsilon_{i,j}} (\Upsilon_{i,j}!) \frac{\partial^{\Upsilon_{i,j}}}{\partial x_j^{\Upsilon_{i,j}}} \right) \phi(\boldsymbol{x}) \right]$$

$$= \sum_{\substack{\Upsilon \in \mathbb{Z}^{P}_{0}{}^{d} \\ \Upsilon \mathbf{1} = \alpha_{1:P}}} C_\Upsilon \left(\prod_{i \in [P], j \in [d]} R_{i,j}^{\Upsilon_{i,j}} \right) \hat{\phi}(\mathbf{1}^\top \Upsilon) \,.$$

So

$$\begin{aligned}
&\mathsf{E}_{\boldsymbol{R}}[\mathsf{E}_{\boldsymbol{x}}[\mathrm{He}_\alpha(\boldsymbol{R}\boldsymbol{x})\phi(\boldsymbol{x})]^2] \\
&= \sum_{\substack{\Upsilon, \Upsilon^0 \in \mathbb{Z}^{P}_{0}{}^{d} \\ \Upsilon \mathbf{1} = \Upsilon^0 \mathbf{1} = \alpha_{1:P}}} C_\Upsilon C_{\Upsilon^0} \hat{\phi}(\mathbf{1}^\top \Upsilon) \hat{\phi}(\mathbf{1}^\top \Upsilon^0) \mathsf{E}_{\boldsymbol{R}} \left[\prod_{i \in [P], j \in [d]} R_{i,j}^{\Upsilon_{i,j} + \Upsilon^0_{i,j}} \right] \\
&\cdot \frac{1}{d^{k\alpha k_1}} \sum_{\substack{\Upsilon, \Upsilon^0 \in \mathbb{Z}^{P}_{0}{}^{d} \\ \Upsilon \mathbf{1} = \Upsilon^0 \mathbf{1} = \alpha_{1:P}}} C_\Upsilon C_{\Upsilon^0} \hat{\phi}(\mathbf{1}^\top \Upsilon) \hat{\phi}(\mathbf{1}^\top \Upsilon^0) 1(\Upsilon_{i,j} + \Upsilon^0_{i,j} \in 2\mathbb{N} \text{ for all } i,j) \\
&\cdot \frac{1}{d^{k\alpha k_1}} \sum_{\substack{\Upsilon, \Upsilon^0 \in \mathbb{Z}^{P}_{0}{}^{d} \\ \Upsilon \mathbf{1} = \Upsilon^0 \mathbf{1} = \alpha_{1:P}}} C_\Upsilon^2 \hat{\phi}(\mathbf{1}^\top \Upsilon)^2 1(\Upsilon_{i,j} + \Upsilon^0_{i,j} \in 2\mathbb{N} \text{ for all } i,j) = (*) \,.
\end{aligned}$$

We argue that for any matrix $\Upsilon \in \mathbb{Z}^{P}_{0}{}^{d}$ with $\Upsilon \mathbf{1} = \alpha$, there are at most $d^{bk\alpha k_1/2c}$ matrices Υ^0 such that $\Upsilon^0 \mathbf{1} = \boldsymbol{\alpha}$ and $\Upsilon + \Upsilon^0$ has all even entries. Indeed, let $S_{even}(\alpha) \subseteq [P] \times [d]$ denote the coordinates (i,j) where $\Upsilon_{i,j} > 0$ is even. Let $S_{odd}(\Upsilon) \subseteq [P] \times [d]$ denote the coordinates where $\Upsilon_{i,j} > 0$ is odd. Then if $\Upsilon + \Upsilon^0$ has all even entries, we must have $S_{odd}(\Upsilon) = S_{odd}(\Upsilon^0)$. So there are at most $\binom{Pd}{|S_{even}(\Upsilon^0)|} \leq \binom{Pd}{bk\alpha k_1/2c} \cdot d^{bk\alpha k_1/2c}$ choices for $S_{even}(\Upsilon^0)$, where we use $S_{even}(\Upsilon^0) \leq \lfloor \|\alpha\|_1/2 \rfloor$. So

$$(*) \cdot \frac{1}{d^{k\alpha k_1}} \sum_{\Upsilon \in \mathbb{Z}^{P}_{0}{}^{d}, \Upsilon \mathbf{1} = \alpha_{1:P}} \hat{\phi}(\mathbf{1}^\top \Upsilon)^2 d^{bk\alpha k_1/2c} \cdot d^{\,dk\alpha k_1/2e} \,,$$

where we use the normalization $\|\phi\|^2 \leq 1$ which implies $\sum_\beta \hat{\phi}(\beta)^2 \cdot 1$. We also use that for each $\beta \in \mathbb{N}^d$, there are at most C_P matrices $\Upsilon \in \mathbb{Z}^{P}_{0}{}^{d}$ such that $\mathbf{1}^\top \Upsilon = \beta$. ■

■