

Repetita Iuvant: Data Repetition Allows SGD to Learn High-Dimensional Multi-Index Functions

Luca Arnaboldi¹, Yatin Dandi^{1,2}, Florent Krzakala¹, Luca Pesce¹, and Ludovic Stephan¹

¹Ecole Polytechnique Fédérale de Lausanne, Information, Learning and Physics lab. CH-1015 Lausanne, Switzerland.

²Ecole Polytechnique Fédérale de Lausanne, Statistical Physics of Computation Laboratory. CH-1015 Lausanne, Switzerland.

Abstract

Neural networks can identify low-dimensional relevant structures within high-dimensional noisy data, yet our mathematical understanding of how they do so remains scarce. Here, we investigate the training dynamics of two-layer shallow neural networks trained with gradient-based algorithms, and discuss how they learn pertinent features in multi-index models, that is target functions with low-dimensional relevant directions. In the high-dimensional regime, where the input dimension d diverges, we show that a simple modification of the idealized single-pass gradient descent training scenario, where data can now be repeated or iterated upon twice, drastically improves its computational efficiency. In particular, it surpasses the limitations previously believed to be dictated by the Information and Leap exponents associated with the target function to be learned. Our results highlight the ability of networks to learn relevant structures from data alone without any pre-processing. More precisely, we show that (almost) all directions are learned with at most $O(d \log d)$ steps. Among the exceptions is a set of hard functions that includes sparse parities. In the presence of coupling between directions, however, these can be learned sequentially through a hierarchical mechanism that generalizes the notion of staircase functions. Our results are proven by a rigorous study of the evolution of the relevant statistics for high-dimensional dynamics.

1 Introduction

Gradient Descent-based algorithms such as Stochastic Gradient Descent (SGD) and its variations, are a fundamental tool in training neural networks, and are therefore the subject of intense theoretical scrutiny. Recent years have witnessed significant advancements in understanding their dynamics and the learning mechanisms of neural nets. Significant progress has been made, in particular, in the case of two-layer networks thanks, in part, to the so-called mean-field analysis [Mei et al., 2018, Chizat and Bach, 2018, Rotskoff and Vanden-Eijnden, 2022, Sirignano and Spiliopoulos, 2020]. A large part of the theoretical approach focused on *one-pass* optimization algorithms, where *each iteration involves a new fresh batch of data*. In particular, in the mathematically solvable case of high-dimensional synthetic Gaussian data, and a low dimensional a multi-index target function model, the class of functions efficiently learned by these *one-pass* methods has been thoroughly analyzed in a series of recent works, and have been shown to be limited by the so-called information exponent [Ben Arous et al., 2021] and leap exponent [Abbe et al., 2022, 2023]. These analyses have sparked many follow-up theoretical works over the last few months, see, e.g. Damian et al. [2022, 2023], Dandi et al. [2023], Bietti et al. [2023], Ba et al. [2023], Moniri et al. [2023], Mousavi-Hosseini et al. [2023], Zweig and Bruna [2023], leading to a picture where SGD dynamics is governed by the information exponent, that is, the order of the first non-zero Hermite coefficient of the target.

A common point between all these works, however, is that they considered the idealized situation where one process data one sample at a time, all of them independently identically distributed, without any repetition. This is, however, not a realistic situation in practice. Indeed, most datasets contains similar data-points, so that repetition occurs (see e.g. for repetition and duplicate in CIFAR [Barz and Denzler, 2020]) Additionally, it is common in machine learning to repeatedly go through the same mini-batch of data multiple time over epoch. Finally, many gradient algorithms, often used in machine learning in Lookahead optimizers [Zhang et al., 2019], Extragradient methods [Korpelević, 1976], or Sharpness Aware Minimization (SAM) [Foret et al., 2021], are *explicitly* few gradient steps over the exact same data-point. It is thus natural to wonder if using such extra-gradient algorithms, or simply reusing many time some data, would change the picture with respect to the current consensus.

There are good reason to believe this to be the case. Indeed, it has been observed very recently [Dandi et al., 2024] that iterating twice over large ($O(d)$) datasets was enough to alter the picture reached in Ben Arous et al. [2021], Abbe et al.

[2022, 2023]. Indeed Dandi et al. [2024] showed that their exist functions that can be learned over few (just two) batch repetition, while they require a much larger (i.e. polynomial in d) number of steps otherwise. While the set of function considered in this work was limited, the conclusion is indeed surprising, thus motivating the following question:

Can data repetition increase the efficiency of SGD when learning any multi-index functions?

We establish a positive answer to the above question by the analysis of more realistic optimization schemes with respect to the standard One-Pass SGD. Indeed, the hardness exponents developed in the seminal works Abbe et al. [2021], Ben Arous et al. [2021] are substantially bound to the idealized training scenario that considers i.i.d data. Slight modifications to this training scenario toward a more realistic setting starkly change the global picture. Such slight modifications include: a) non-vanishing correlations between different data points, that is a natural requirement for real datasets; b) processing the same data point multiple times in the optimization routine, i.e., a standard step in any algorithmic procedure when looping over different epochs.

We consider an analytically tractable a single-pass training scheme to model these changes: we process one Gaussian sample at a time, but *the gradient step is repeated twice!* This simple adjustment to the idealized SGD optimization scheme, which is actually often used in machine learning in Lookahead optimizers [Zhang et al., 2019], Extragradient methods [Korpelevič, 1976], or in the context of Sharpness Aware Minimization (SAM Foret et al. [2021]), will be seen to make single-pass algorithms drastically more efficient, and, as we will see, as efficient than our best algorithms in most cases.

We show in particular that most multi-index function are learned with total sample complexity either $O(d)$ or $O(d \log d)$ if they are related to the presence of a symmetry, without any need for preprocessing of the data that is performed in various works (See e.g., Mondelli and Montanari [2018], Luo et al. [2019], Maillard et al. [2020], Chen and Meka [2020]). Our results thus demonstrate that shallow neural networks turns out to be significantly more powerful than previously believed [Ben Arous et al., 2021, Abbe et al., 2022, 2023, Arnaboldi et al., 2023] in identifying relevant feature in the data. In fact, they are shown in many (but not all) cases to saturate the bounds predicted by statistical queries (SQ) [Damian et al., 2024] (rather than the more limited correlation statistical queries (CSQ) bounds that were characteristic of the former consensus [Ben Arous et al., 2021, Abbe et al., 2023]). Our conclusions follow from the rigorous analysis of the overlaps between the trained weight vectors with the low-dimensional relevant subspace that draws inspiration from the pivotal works of Saad and Solla [1995], Coolen and Saad [2000], Coolen et al. [2000], Ben Arous et al. [2021].

2 Setting and Main Contributions

Given a dataset of N labeled examples $\{\mathbf{z}^\nu, y^\nu\}_{\nu \in [N]}$ in d dimensions, we analyze the learning properties of fully connected two-layer networks with first layer weights $W = \{\mathbf{w}_j\}_{j \in [p]}$, second layer $\mathbf{a} \in \mathbb{R}^p$, and activation function $\sigma : \mathbb{R} \rightarrow \mathbb{R}$

$$f(\mathbf{z}; W, \mathbf{a}) = \frac{1}{p} \sum_{j=1}^p a_j \sigma(\langle \mathbf{z}, \mathbf{w}_j \rangle) \quad (1)$$

We consider target functions that are dependent only on few orthogonal relevant directions $k = O_d(1)$ encoded in the target's weight matrix $W^* = \{\mathbf{w}_r^*\}_{r \in [k]} \in \mathbb{R}^{k \times d}$:

$$y^\nu = f^*(\mathbf{z}^\nu) = h^*(W^* \mathbf{z}^\nu) \quad (2)$$

This model for structured data is usually referred to as a *multi-index* model. Our main objective will be to analyze how efficiently two-layer nets adapt to this low-dimensional structure during training.

Training algorithm – Our goal is to minimize the following objective:

$$\mathcal{R}(W, \mathbf{a}) = \mathbb{E}_{\mathbf{z}, y} [\mathcal{L}(f(\mathbf{z}, W, \mathbf{a}), y)] \quad (3)$$

where $\mathcal{L} : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$ is a loss function, and the expectation is over the data distribution.

This objective is non-convex, and we do not have direct access to the expectation over the data. A central question in Machine Learning is therefore to quantify the properties of the weights attained at convergence by different algorithms. In this manuscript we consider a specific class of one-pass training routines to minimize the empirical risk. First, we partition the dataset in disjoint mini-batches of size n_b , i.e. $\mathcal{D} = \cup_{t \in [T]} \{\mathbf{x}^\nu, y^\nu\}_{\nu \in [tn_b, (t+1)n_b]}$ where T is the total number of iterations considered. For each algorithmic step, we update the hidden layer neurons $W = \{\mathbf{w}_j\}_{j \in [p]}$ with the program $\mathcal{A} : (\mathbf{w}_{j,t}, \gamma, \rho, n_b) \in \mathbb{R}^{d+3} \rightarrow \mathbf{w}_{t+1,j} \in \mathbb{R}^d$.

Algorithm 1 (Optimizer Main Step).

$$\begin{aligned}\mathbf{w}_{t+1,j} &= \mathcal{A}(\mathbf{w}_t, \gamma, \rho, n_b) \\ &= \mathbf{w}_{t,j} - \frac{\gamma}{n_b} \sum_{\nu \in [n_b]} \nabla_{\mathbf{w}_{t,j}} \mathcal{L}(f(\mathbf{z}^\nu; \tilde{W}_t(\rho), \mathbf{a}_0), y^\nu) \\ \tilde{\mathbf{w}}_{t,j}(\rho) &= \mathbf{w}_{t,j} - \rho \nabla_{\mathbf{w}_{t,j}} \mathcal{L}(f(\mathbf{z}^\nu; W_t, \mathbf{a}_0), y^\nu)\end{aligned}\tag{4}$$

This class of algorithms defined by the optimizer step $\mathcal{A}(\cdot, \gamma, \rho, n_b)$, in which we consider the final gradient update after taking a linear combination of the current iterate and its current gradient (noted as $\tilde{W}_t(\rho)$ in the above), is broadly used in different contexts. Indeed, routines with positive ρ parameters correspond to Extragradient methods (EgD) [Korpelevič, 1976], while with negative ρ have been recently used in the context of Sharpness Aware Minimization (SAM) [Foret et al., 2021]. For clarity, we present our theoretical results with the above, easily interpretable, Algorithm 1. However, our theoretical claims are valid in a more general setting; we refer to Appendices (A, B) for additional results on more general optimizer steps.

As previously stated, the central object of our analysis is the efficiency of the network to adapt to the low-dimensional structure identified by W^* . Therefore, we focus on the learned representations by the first layer weights W_t while keeping the second layer weights $\mathbf{a}_t = \mathbf{a}_0$ fixed during training. This assumption is favorable to performing the theoretical analysis and is largely used in the theoretical community (e.g., Damian et al. [2022], Ba et al. [2022], Bietti et al. [2022]).

Numerous theoretical efforts have been devoted to understanding the sample complexity needed to learn multi-index targets. However, an important aspect of many algorithmic routines considered is to exploit a clever *warm start* to initialize the iterates [Mondelli and Montanari, 2018, Luo et al., 2019, Maillard et al., 2020, Chen and Meka, 2020]. We show in this work that such preprocessing of the data is not needed to learn all polynomial multi-index functions in $T = O(d \log d)$ with the routine in Alg. 1. As a way to show this, we consider the extreme case in which the initial iterates W_0 lives in the orthogonal subspace of the target's weights W^* , i.e., the ‘coldest’ possible start.

Weak recovery in high dimensions – The essential object in our analysis is the evolution of the correlation between the hidden neurons $W = \{\mathbf{w}_j\}_{j \in [p]}$ and the target’s weights W^* , as a function of the algorithmic time steps. More precisely, we are interested in the number of algorithmic steps needed to achieve an order one correlation with the target’s weights W^* , known as *weak recovery*.

Definition 1 (Weak recovery). *The target subspace V^* is defined as the span of the rows of the target weights W^* :*

$$V^* = \text{span}(\mathbf{w}_1^*, \dots, \mathbf{w}_k^*)\tag{5}$$

We define the following weak recovery stopping time for a parameter $\eta \in (0, 1)$ independent from d :

$$t_\eta^+ = \min\{t \geq 0 : \|WW^{*\top}\|_F \geq \eta\}\tag{6}$$

From Information Exponents to Generative Exponents – Recent works have sharply theoretically characterized the weak recovery stopping time for a variety of multi-index targets learned with one-pass Stochastic Gradient Descent. First Ben Arous et al. [2021] for the single-index case, then Abbe et al. [2022] for the multi-index one have unveiled the presence of an Information Exponent ℓ characterizing the time complexity needed to weakly recover the target subspace V^* , defined in the single-index case as

$$\ell = \min\{k \in \mathbb{N} : \mathbb{E}_{x \sim \mathcal{N}(0,1)} [h^*(x)H_k(x)] \neq 0\}.\tag{7}$$

where H_k is the k -th Hermite polynomial [O'Donnell, 2014]. Under this framework, the sample complexity required to achieve weak recovery when the link function has Information Exponent ℓ is [Ben Arous et al., 2021]

$$T(\ell) = \begin{cases} O(d^{\ell-1}) & \text{if } \ell > 2 \\ O(d \log d) & \text{if } \ell = 2 \\ O(d) & \text{if } \ell = 1. \end{cases}\tag{8}$$

These bounds have been improved by Damian et al. [2023] up to order $d^{\ell/2}$; the latter matches the so-called *Correlated Statistical Query* lower bound, which considers queries of the form $\mathbb{E}[y\phi(\mathbf{z})]$.

However, this Information Exponent is not the right measure of complexity for low-dimensional weak recovery. Indeed, [Damian et al. \[2024\]](#) introduce a new Generative Exponent ℓ^* (generative exponent) governing the weak recovery time scaling for algorithms in the Statistical Query (SQ) or Low Degree Polynomial (LDP) family. This exponent is defined as

$$\ell^* = \min\{k \in \mathbb{N} : \mathbb{E}_{x \sim \mathcal{N}(0,1)} [f(h^*(x))H_k(x)] \neq 0 \text{ for some } f : \mathbb{R} \rightarrow \mathbb{R}\}, \quad (9)$$

which allows for the application of arbitrary transformations of the labels before performing the queries. Under this definition, the “hard” problems with $\ell^* > 2$ are also impossible to learn for the best first-order algorithm, *Approximate Message Passing* [[Barbier et al., 2019](#), [Celentano et al., 2021](#), [Mondelli and Montanari, 2018](#), [Luo et al., 2019](#), [Maillard et al., 2020](#)].

We discuss in this work the extension of the generative exponent to the multi-index setting and investigate the dynamics of specific gradient-based programs (Algorithm 1) as a function of the freshly defined hardness exponent ℓ^* . These theoretical findings match the recent one of [Kou et al. \[2024\]](#) on the computational efficiency of SGD in learning sparse k -parity (associated with $\ell^* = k$).

When the optimal transformation f is known, a simple SGD algorithm on $(z, f(y))$ achieves the Generative Exponent lower bound. However, one problem remains: does there exist a class of SGD-like algorithms that can achieve this lower bound agnostically?

Achieving lower bounds with data repetition – Although the performance of single-pass SGD is limited by the Information Exponent of h^* , the situation drastically changes when multiple-pass algorithms are considered. Recently, [Dandi et al. \[2024\]](#) proved that non-even single-index targets are weakly recovered in $T = O(d)$ when considering extensive batch sizes with multiple pass. This begs the question of how good can a “reusing” algorithm be. We answer this question for the class of polynomial link functions:

Theorem 1 (Informal). *There is a choice of hyperparameters such that if h^* is a polynomial function, Algorithm 1 achieves weak recovery in $O(d \log(d)^2)$ samples.*

This shows that a simple gradient-based algorithm, which only requires seeing the data twice, is optimal for learning any polynomials.

Learning representations – The ability of learning relevant features/representations in the data is arguably one of the key properties of neural networks. However, a considerable attention in the theoretical community has been devoted to understanding the lazy training of two-layer networks [[Jacot et al., 2018](#)] where features are not effectively learned. Indeed, in this regime, shallow networks are equivalent to kernel machines, a class of fixed feature methods which are not able to adapt efficiently to low-dimensional relevant structure in the data (see e.g. [Ghorbani et al. \[2020\]](#)). Thus, understanding how representations are learned and providing corrections to the kernel regime is a central question in machine learning theory and has been explored in a variety of works (see e.g., [Dudeja and Hsu \[2018\]](#), [Atanasov et al. \[2022\]](#), [Bietti et al. \[2022\]](#), [Damian et al. \[2022\]](#), [Petrini et al. \[2022\]](#)). In this manuscript we provably characterize the number of iterations needed from Algorithm 1 to learn the relevant features of the data, i.e., attain weak recovery of the target subspace. Our results provide the scaling of the relevant hyperparameters (γ, ρ) with the diverging input dimension d . The minibatch size will be fixed to $n_b = 1$ in the main body, and we refer to Appendix B for the extension to larger batch sizes.

Summary of Main Results

- We prove for single-index targets that one-pass gradient-based algorithms are able to surpass the Correlation Statistical Query (CSQ) limitations established by the Information Exponent [[Ben Arous et al., 2021](#)].
- We unveil that the dynamics of the family of Algorithm 1 are governed by the generative exponent introduced in [Damian et al. \[2024\]](#) with an additional polynomial restriction on the transformation of the output.
- We generalize the notion of generative exponent to multi-index targets. Similarly to the single-index case, one-pass gradient-based algorithms can overcome CSQ performance dictated by the Leap Coefficient [[Abbe et al., 2022](#)].
- We prove that all polynomial multi-index functions are learned by Algorithm 1 either with total sample complexity $O(d)$ or $O(d \log(d)^2)$ if associated with the presence of a symmetry.
- The implementation of Algorithm 1 does not require pre-processing to initially correlate the estimation with the ground truth, and learns the meaningful representations from data alone. This is in contrast numerous findings

(e.g., [Mondelli and Montanari \[2018\]](#), [Luo et al. \[2019\]](#), [Maillard et al. \[2020\]](#) for single index targets or [Chen and Meka \[2020\]](#) for multi-index ones).

- We characterize the class of hard functions not learned in (almost) linear sample complexity. We show, however, that such functions can be learned through an hierarchical mechanism that extends the CSQ staircase first developed in [Abbe et al. \[2021\]](#) to a different, larger set, of functions.
- We validate the formal theoretical claims with detailed numerical illustrations. The code to reproduce representative figures is available in the Github repository <https://github.com/IdePHICS/Repetita-Iuvant>.

3 Single-Index Model

3.1 Main results

We first consider, for simplicity, the class of single-index models, in which $k=p=1$. This corresponds to a mismatched setting of Generalized Linear Models, in which we want to learn

$$y = h^*(\langle \mathbf{w}^*, \mathbf{z} \rangle), \quad \text{with } f(\mathbf{z}, \mathbf{w}, a) = a\sigma(\langle \mathbf{w}, \mathbf{z} \rangle).$$

In this section we rigorously characterize the number of iterations needed for the class of Algorithms 1 to perform weak recovery (as in Definition 1) in the context of single-index targets. We establish a clear separation between the learning efficiency of the algorithmic family 1 and One-Pass SGD, limited by the Information Exponent of the target to be learned [[Ben Arous et al., 2021](#)].

We start by introducing a restriction of the generative information exponent in [Damian et al. \[2024\]](#) to polynomial transformations.

Definition 2 (Polynomial Generative Information exponent). *We define ℓ_p^* as the smallest integer k such that there exists a polynomial $p : \mathbb{R} \rightarrow \mathbb{R}$ with:*

$$\mathbb{E}_{x \sim \mathcal{N}(0,1)} [p(h^*(x)) H_k(x)] \neq 0, \tag{10}$$

Our assumptions on the SGD implementation and the target function are as follows:

Assumption 1. *Algorithm 1 is run with single-sample steps ($n_b = 1$), and uses the correlation loss*

$$\mathcal{L}(y, \hat{y}) = 1 - y\hat{y}.$$

We also consider a spherical version of Alg. 1, with

$$\mathbf{w}_{t+1} = \frac{\mathbf{w}_t - \gamma \nabla_{\mathbf{w}}^\perp \mathcal{L}(f(\mathbf{z}^\nu; \tilde{\mathbf{w}}_t(\rho), a_0), y^\nu)}{\|\mathbf{w}_t - \gamma \nabla_{\mathbf{w}}^\perp \mathcal{L}(f(\mathbf{z}^\nu; \tilde{\mathbf{w}}_t(\rho), a_0), y^\nu)\|} \tag{11}$$

where $\nabla^\perp f = \nabla f - \langle \nabla f, \mathbf{w} \rangle \mathbf{w}$ is the spherical gradient.

Assumption 2. *The activation σ is analytic, with bounded first and second derivative. Further, for all $n \geq 1$, we have*

$$\mathbb{E} [\sigma^{(n)}(x)\sigma'(x)^{n-1}] \neq 0 \quad \text{and} \quad \mathbb{E} [x\sigma^{(n)}(x)\sigma'(x)^{n-1}] \neq 0$$

Assumption 3. *The initial value \mathbf{w}_0 is drawn according to the uniform measure on*

$$\mathbb{S}_\epsilon = \{\mathbf{w} \in \mathbb{S}^{d-1} : \text{sign}(\langle \mathbf{w}, \mathbf{w}^* \rangle) = \epsilon\}$$

where ϵ depends on h^* and σ .

We are now ready to state our main result.

Theorem 2. *Suppose that Assumptions 1 and 2 hold, and let $\rho = \rho_0 d^{-1}$. Then, for any $\delta > 0$, there exists constants $\gamma_0(\delta)$ and $C(\delta)$ such that the following holds:*

- If $\ell_p^* = 1$, choosing $\gamma = \gamma_0(\delta)d^{-1}$, then $\mathbb{P}(t_\eta^+ \leq C(\delta) \cdot d) \geq 1 - \delta$,
- If $\ell_p^* = 2$, choosing $\gamma = \gamma_0(\delta)[d \log(d)]^{-1}$, then $\mathbb{P}(t_\eta^+ \leq C(\delta) \cdot d \log(d)^2) \geq 1 - \delta$.

The above holds for almost every choice of ρ_0 under the Lebesgue measure.

Proof Sketch – We provide here the proof sketch for Theorem 2. For any vector \mathbf{w} , under the correlation loss, we have

$$\nabla_{\mathbf{w}} \mathcal{L}(f(\mathbf{z}; \mathbf{w}, a_0), y) = a_0 y \sigma'(\langle \mathbf{w}, \mathbf{z} \rangle) \cdot \mathbf{z}, \quad (12)$$

so the gradient is aligned with \mathbf{z} . As a result, we have

$$\nabla_{\mathbf{w}} \mathcal{L}(f(\mathbf{z}; \tilde{\mathbf{w}}(\rho), a_0), y) = a_0 y \sigma'(\langle \mathbf{w}, \mathbf{z} \rangle + a_0 \rho y \sigma'(\langle \mathbf{w}, \mathbf{z} \rangle) \cdot \underbrace{\|\mathbf{z}\|^2}_{\approx d}) \cdot \mathbf{z} \quad (13)$$

This expression for the gradient exhibits two important properties:

- Even though $\rho \asymp d^{-1}$, since the gradient lies along \mathbf{z} , the additional dot product with \mathbf{z} amplifies the signal by a factor of d ,
- The resulting gradient (13) is a *non-linear* function of y , as opposed to the linear function of (12).

The latter property enables us to show that the dynamics driven by equation (13) can implement all polynomial transformations of the output y , for almost all choices of ρ_0 . Subsequently, the SGD algorithm on the transformed input $p(y)$ can be studied with the same techniques as Ben Arous et al. [2021], but with an information exponent equal to ℓ_p^* .

3.2 Discussion

Significance and necessity of the assumptions – Assumption 1 simplifies Alg. 1 to a more tractable version. In particular, the normalization step allows us to keep track of only one parameter, the overlap between \mathbf{w}_t and \mathbf{w}^* . We expect however Thm.2 to hold in the general setting of Alg.1.

Assumption 2 is a regularity assumption, ensuring that some quantities of interest in the proof are non-zero. The first part is satisfied by virtually every activation function except for ReLU; the second is more restrictive, but we show that it is satisfied for biased activations:

Lemma 1. *Let σ be a non-polynomial analytic function. Then the second part of Assumption 2 is true for the function $x \mapsto \sigma(x + b)$ for almost every choice of b (according to the Lebesgue measure).*

Assumption 3 is similar to the one in Ben Arous et al. [2021], and is necessary in case the initialization is a strict saddle. It is equivalent to conditioning the usual uniform initialization on an event of probability 1/2.

Finally, although we state our result as an almost sure event over ρ_0 , it can be reformulated into an event on the second layer weight a_0 , in which the randomness comes from the initialization step.

Learning polynomial functions – Although our definition of the Polynomial Generative Information exponent is more restrictive than the one in Damian et al. [2024], we show that it is enough to allow learning of a large class of functions, namely polynomials:

Theorem 3. *Assume that h^* is a polynomial function. Then the Polynomial Generative Information exponent of h^* is always at most 2, and is equal to 1 whenever h^* is not even.*

This theorem implies that the Extragradient algorithm class allows us to match the sample complexity of the algorithm of Chen and Meka [2020], without the need for *ad hoc* preprocessing.

Beyond Algorithm 1 – A succinct summary of the proof goes as follows: the first time we see a sample \mathbf{z} , we store information about y along the direction of \mathbf{z} . The second time we see the same sample, the component along \mathbf{z} in \mathbf{w} interact *non-linearly* with y , which bypasses the CSQ framework. Importantly, we expect that this behavior also appears in SGD algorithms where we do not see the data twice *in a row*, but *across multiple epochs*! Indeed, upon small enough correlation between the \mathbf{z}' , seeing multiple examples before the repetition of \mathbf{z} should not interfere with the data stored along this direction. We are therefore not claiming that Algorithm 1 is superior to other classically-used methods; rather, it provides a tractable and realistic setting to study data repetition in SGD, which was ignored in previous theoretical works on the topic.

3.3 Numerical illustrations

We illustrate in Fig. 1 the stark difference between the weak recovery dynamics of One-Pass SGD and Algo. 1 for single-index targets. The Cosine Similarity between the learned weight \mathbf{w}_t and the ground truth informative direction

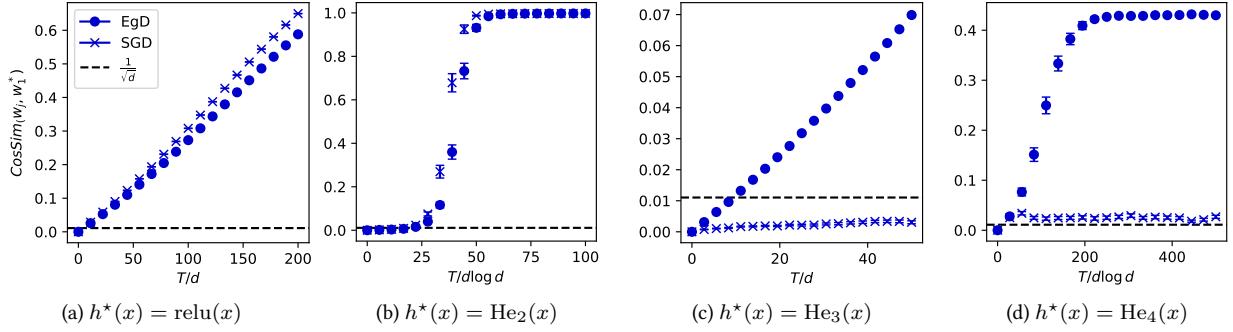


Figure 1: **Learning single-index targets** – Evolution of the Cosine Similarity attained by EgD (crosses) and SGD (dots) as a function of normalized iteration time. The dashed horizontal line $\frac{1}{\sqrt{d}}$ is a visual guide to place random performance. **(a)** $(\ell, \ell^*) = (1, 1)$: both the algorithm learn in linear time. **(b)** $(\ell, \ell^*) = (2, 2)$: both the algorithm learn in $O(d \log d)$ time. **(c)** $(\ell, \ell^*) = (3, 1)$: EgD learns in linear time, while SGD would learn the same function in quadratic time. **(d)** $(\ell, \ell^*) = (4, 2)$: EgD learns in $O(d \log d)$ time, while SGD would learn in $O(d^3)$ time. See all the details in App. C.

w_* is shown as a function of the time steps t for different target functions. Two optimization routines are considered to exemplify the learning behaviour: a) Extragradient Descent (EgD), corresponding to the family of Algorithms 1 with positive ρ parameters; b) One-Pass SGD, corresponding to vanilla SGD, or equivalently Algorithm 1 associated with $\rho = 0$ hyperparameter. The scaling of (γ, ρ) as a function of the input dimension d and exponent ℓ^* are given in Thm. 2, while the mini-batch size is fixed to $n_b = 1$ (See App. B for extension to larger n_b). For all plots, we take σ to be the relu activation and the implementation details can be found in Appendix C.

SGD-easy non-symmetric targets ($\ell = \ell^* = 1$) – The Left section of Fig. 1 shows $h^*(x) = \text{relu}(x)$. Here both SGD and Algorithm 1 learn in $T = O(d)$.

SGD-easy symmetric targets ($\ell = \ell^* = 2$) – The Center-Left section of Fig. 1 shows $h^*(x) = \text{He}_2(x)$. Here both SGD and Algorithm 1 learn in $T = O(d \log d)$.

SGD-hard non-symmetric targets ($\ell > 2, \ell^* = 1$) – The Center-Right section of Fig. 1 shows $h^*(x) = \text{He}_3(x)$. Here Algorithm 1 learns in $T = O(d)$ while One Pass SGD suffers from the limitations detailed in eq. (8), i.e. $T = O(d^{\ell-1})$ with $\ell = 3$ for this case.

SGD-hard symmetric targets ($\ell > 2, \ell^* = 2$) – The Right section of Fig. 1 shows $h^*(x) = \text{He}_4(x)$. Here Algorithm 1 learns in $T = O(d \log d)$ while One Pass SGD suffers from the time scaling law detailed in eq. (8) $T = O(d^{\ell-1})$.

4 Multi-Index Model

We now investigate the superiority of Algorithm 1 over the idealized One-Pass SGD training scheme when learning multi-index targets. In this setting, the networks weights can align with only a few select directions from V^* . To quantify this behavior, we define directional versions of the Information and Generative Exponents:

Definition 3. Let $\mathbf{v} \in V^*$. The Information Exponent $\ell_{\mathbf{v}}$ (resp. Generative exponent $\ell_{\mathbf{v}}^*$) of y in the direction \mathbf{v} is the smallest k such that

$$\mathbb{E}[y H_k(\langle \mathbf{v}, z \rangle)] \neq 0 \quad (\text{resp. } \mathbb{E}[f_{\mathbf{v}}(y) H_k(\langle \mathbf{v}, z \rangle)] \neq 0)$$

for a function $f_{\mathbf{v}}$. We also let $\ell = \min_{\mathbf{v}} \ell_{\mathbf{v}}$ and $\ell^* = \min_{\mathbf{v}} \ell_{\mathbf{v}}^*$.

The identification of the class of hard functions has been subject of intense theoretical scrutiny [Abbe et al., 2021, 2022, 2023, Dandi et al., 2023, Bietti et al., 2023]; and for the problem of initial alignment it was shown to depend on the Information exponent defined above. Thus, similarly to the single-index scenario, the time complexity needed for Algorithm 1 to weakly recover the target subspace follows the law in eq. (8) for the multi-index information exponent ℓ . A natural question is thus to ask whether the class of Algorithm 1 also allows us to bypass this requirement.

Our simulations enable us to affirmatively answer this question. We illustrate this numerically by comparing the performance of One-Pass SGD with Algorithm 1 in Figure 2; we show that the dynamics of the latter algorithm is governed by the multi-index Generative Exponent (Def. 3).

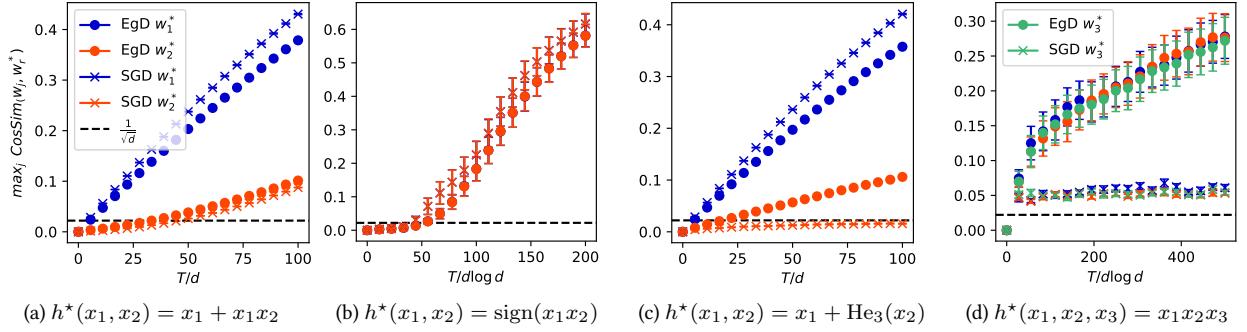


Figure 2: **Learning multi-index targets** – Evolution of the maximum Cosine Similarities attained by EgD (crosses) and SGD (dots) as a function of the normalized iteration time. Different target directions $\{\mathbf{w}_r^*\}_{r \in [k]}$ are identified by different colors: \mathbf{w}_1^* (blue), \mathbf{w}_2^* (orange), \mathbf{w}_3^* (green). The dashed horizontal line $\frac{1}{\sqrt{d}}$ is a visual guide to place random performance. See details in App. C
(a) $(\ell, \ell^*) = (1, 1)$: both the algorithm learn the first direction in linear time, and they also learn the second in another $O(d)$ steps using the staircase mechanism. **(b)** $(\ell, \ell^*) = (2, 2)$: both the algorithm learn the two direction simultaneously in $O(d \log d)$ steps.
(c) $(\ell_{\mathbf{w}_2^*}, \ell_{\mathbf{w}_2^*}^*) = (3, 1)$: both the algorithms learn \mathbf{w}_1^* in linear time, but only EgD can also learn \mathbf{w}_2^* in $O(d)$; SGD requires $O(d^2)$ steps to learn \mathbf{w}_2^* . **(d)** $(\ell, \ell^*) = (3, 2)$: EgD learns all the 3 directions simultaneously in $O(d \log d)$ steps, while SGD needs quadratic time to achieve the same recovery.

4.1 Numerical investigation

We measure the maximal cosine similarity between the hidden layer neurons and the different rows of the target’s weight matrix W^* as a function of the iteration time. We set without loss of generality the target’s directions to the standard basis of \mathbb{R}^d to ease the notation. The algorithms considered are again EgD and vanilla One-Pass SGD. We again refer to Appendix C for the implementation details.

SGD-easy non-symmetric targets ($\ell = \ell^* = 1$) – Fig. 2, left, shows $h^*(x_1, x_2) = x_1 + x_1x_2$. Here both SGD and Alg. 1 learn in $T = O(d)$ the directions $(\mathbf{w}_1^*, \mathbf{w}_2^*)$.

SGD-easy symmetric targets ($\ell = \ell^* = 2$) – Fig. 2, center left, shows $h^*(x_1, x_2) = x_1x_2$. Here both SGD and Alg. 1 learn in $T = O(d \log d)$ the directions $(\mathbf{w}_1^*, \mathbf{w}_2^*)$.

Non-symmetric targets with SGD-easy and hard directions ($\ell_v > 2, \ell_v^* = 1$) – Fig. 2, center-right, shows $h^*(x_1, x_2) = x_1 + \text{He}_3(x_2)$. Here both SGD and Alg. 1 learn in $T = O(d)$ the direction \mathbf{w}_1^* , that satisfies $\ell_{\mathbf{w}_1^*} = \ell_{\mathbf{w}_1^*}^* = 1$. However, since $\ell_{\mathbf{w}_2^*} = 3$ while $\ell_{\mathbf{w}_2^*}^* = 1$, One-Pass SGD suffers from the limitations detailed in eq. (8) and requires $\Omega(d^2)$ samples to learn the second direction \mathbf{w}_2^* . This contrasts with the behaviour of Alg. 1 that learns also \mathbf{w}_2^* in $T = O(d)$ steps.

SGD-hard symmetric targets ($\ell > 2, \ell^* = 2$) – The right section Fig. 2 shows $h^*(x_1, x_2, x_3) = x_1x_2x_3$. The Information Exponent is $\ell = 3$ and hence vanilla SGD is not able to learn any direction in the target subspace in $T = O(d \log d)$ iterations. However, Algorithms 1 learns in $T = O(d \log d)$ steps all the three directions $\{\mathbf{w}_1^*, \mathbf{w}_2^*, \mathbf{w}_3^*\}$ since the Generative Exponents $\ell_{\mathbf{w}_r^*}^*$ are all equal to two.

4.2 Learning hard functions through a hierarchical mechanism

The investigation of the hierarchical nature of SGD learning has attracted noticeable attention [Abbe et al., 2021, 2022, 2023, Dandi et al., 2023]. In this paper, we portray a completely different picture in terms of computational efficiencies when data repetition is considered in the algorithmic SGD routine. One may wonder if there is a generalization of such a hierarchical learning mechanism to the present novel setting; we show that coupling between directions can hierarchically guide the learning process.

The Left panel in Fig. 3 shows an example of the so-called staircase functions [Abbe et al., 2021], i.e. $f^*(\mathbf{z}) = z_1 + z_1 \text{He}_3(z_2)$. While one-pass SGD needs to learn hierarchically first the direction \mathbf{w}_1^* in $T = O(d)$ and then \mathbf{w}_2^* in $T = O(d^2)$, Algorithm 1 easily learns both in linear time. This observation could lead to infer that hierarchical learning mechanisms are not present when more realistic training scenarios are considered.

To refute this, we illustrate in the right panel of Fig. 3 a scenario that precisely exemplifies the presence of hierarchical mechanisms within Algorithm 1. We run this algorithm on two different target functions, $f_{\text{sign}}^*(\mathbf{z}) = \text{sign}(z_1 z_2 z_3)$ and $f_{\text{stair}}^*(\mathbf{z}) = \text{He}_2(z_1) + \text{sign}(z_1 z_2 z_3)$. We show in App. A that 3–sparse parities of the form $\text{sign}(z_1 z_2 z_3)$ are hard

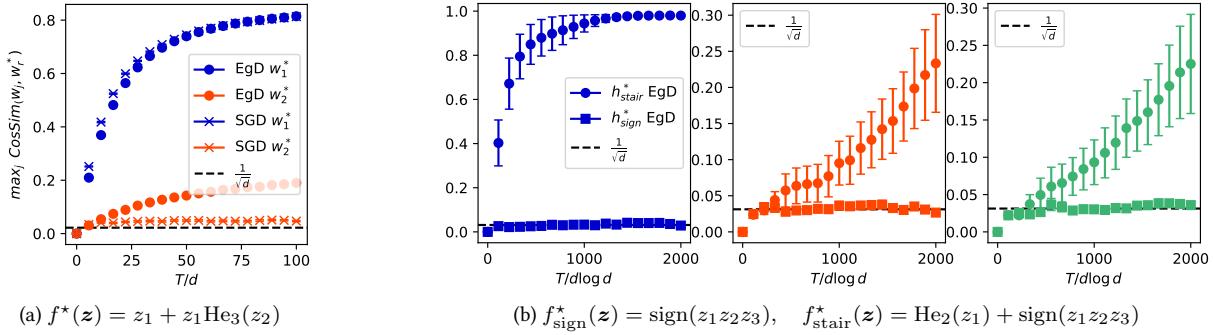


Figure 3: Hierarchical picture of learning – Evolution of the maximum Cosine Similarities with different target directions $\{\mathbf{w}_r^*\}_{r \in [k]}$ are identified by different colors: \mathbf{w}_1^* (blue), \mathbf{w}_2^* (orange), \mathbf{w}_3^* (green) as a function of the normalized iteration time. **(a):** both the algorithms learn \mathbf{w}_1^* in linear time, but only EgD can also learn \mathbf{w}_2^* in $O(d)$ using staircase mechanism; SGD requires another $O(d^2)$ steps to take advantage of the staircase and learn \mathbf{w}_2^* . **(b):** EgD performance for the target function f_{stair}^* (dots) and f_{sign}^* (squares); we plot in separate subplots the overlap with different directions to highlight the presence of hierarchical learning mechanism. Learning the first target direction \mathbf{w}_1^* triggers the hierarchical mechanism and EgD is able to weakly recover the full target subspace $\text{Span}(\mathbf{w}_1^*, \mathbf{w}_2^*, \mathbf{w}_3^*)$ in $O(d \log d)$, while this does not happen when removing $\text{He}_2(x_1)$ from the target. See App. C for additional details.

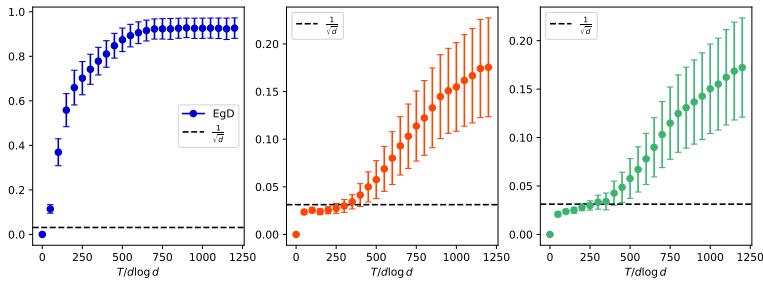


Figure 4: Novel hierarchical picture of learning – Evolution of the maximum Cosine Similarities attained by EgD with different target directions $\{\mathbf{w}_r^*\}_{r \in [k]}$ are identified by different colors: \mathbf{w}_1^* (blue), \mathbf{w}_2^* (orange), \mathbf{w}_3^* (green) as a function of the normalized iteration time. The target function is $f_{\text{sq}-\text{stair}}^*(z) = \text{He}_4(z_1) + \text{sign}(z_1 z_2 z_3)$, that is an SQ-staircase, but not a CSQ-staircase. EgD learn \mathbf{w}_1^* in $O(d \log d)$ steps and use the information to learn the other two directions in another $O(d \log d)$ steps. SGD (not showed in the plot) cannot take advantage of the staircase mechanism since $\text{He}_4(z_1)$ and $\text{sign}(z_1 z_2 z_3)$ have information exponent $\ell = 4$ and $\ell = 3$ respectively. See App. C for additional details.

functions even for SGD with data repetition (Algorithm 1), and indeed they are not learned in (almost) linear time. On the other hand, for the function f_{stair}^* , our simulations predict that the first direction \mathbf{w}_1^* is learned in $T = O(d \log d)$ iterations (see the rightmost panel of Fig. 2). However, once the direction \mathbf{w}_1^* is learned, it can be used to obtain order-one correlation with $\{\mathbf{w}_2^*, \mathbf{w}_3^*\}$ again in $T = O(d \log d)$ steps. While the latter example belongs to the class of “CSQ” staircase function depicted in Abbe et al. [2021], novel “SQ” hierarchical mechanisms arise in our framework, such as for instance the function $f_{\text{sq}-\text{stair}}^*(z) = \text{He}_4(z_1) + \text{sign}(z_1 z_2 z_3)$ (see Fig. 4).

5 Conclusions

We have pushed in this paper the boundaries of learning multi-index models in high-dimensional settings using gradient-type algorithms, and shown that, contrary to what was believed, they are very efficient in doing so. We argue in this manuscript with a simple analytically solvable model how to bypass the limitations of the CSQ framework: we process the same sample twice. Crucially, similar phenomenon appears also in standard SGD when training over multiple epochs, with the only difference of not seeing same the data consecutively. We believe that the model proposed in this manuscript paves the way for the analysis of more realistic training scenario where *correlation* in the data is taken into account, surpassing the limiting i.i.d. scenario.

A natural avenue of future research would be to follow Damian et al. [2023] and use smoothing to make them even closer to SQ bounds in the more difficult setting where the generative exponent is higher than two (associated with a restricted class of non-polynomial functions).

Acknowledgements

The authors would like to thank Nicolas Flammarion, Bruno Loureiro, Nati Srebro, Emanuele Troiani, and Lenka Zdeborová for interesting discussions. This work was supported by the Swiss National Science Foundation under grant SNSF OperaGOST (grant number 200390).

References

- 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.
- Lenaic 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, 2018.
- Grant Rotskoff and Eric Vanden-Eijnden. Trainability and accuracy of artificial neural networks: An interacting particle system approach. *Communications on Pure and Applied Mathematics*, 75(9):1889–1935, 2022. doi: <https://doi.org/10.1002/cpa.22074>.
- Justin Sirignano and Konstantinos Spiliopoulos. Mean field analysis of neural networks: A central limit theorem. *Stochastic Processes and their Applications*, 130(3):1820–1852, 2020.
- Gerard Ben Arous, Reza Gheissari, and Aukosh Jagannath. Online stochastic gradient descent on non-convex losses from high-dimensional inference. *Journal of Machine Learning Research*, 22(106):1–51, 2021.
- Emmanuel Abbe, Enric Boix-Adserà, 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, 2022.
- Emmanuel Abbe, Enric Boix Adserà, and Theodor Misiakiewicz. Sgd learning on neural networks: leap complexity and saddle-to-saddle dynamics. In Gergely Neu and Lorenzo Rosasco, editors, *Proceedings of Thirty Sixth Conference on Learning Theory*, volume 195 of *Proceedings of Machine Learning Research*, pages 2552–2623. PMLR, 12–15 Jul 2023. URL <https://proceedings.mlr.press/v195/abbe23a.html>.
- Alexandru Damian, Jason Lee, and Mahdi Soltanolkotabi. Neural networks can learn representations with gradient descent. In Po-Ling Loh and Maxim Raginsky, editors, *Proceedings of Thirty Fifth Conference on Learning Theory*, volume 178 of *Proceedings of Machine Learning Research*, pages 5413–5452. PMLR, 02–05 Jul 2022.
- Alex Damian, Eshaan Nichani, Rong Ge, and Jason D. Lee. Smoothing the landscape boosts the signal for SGD: optimal sample complexity for learning single index models. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/02763667a5761ff92bb15d8751bcd223-Abstract-Conference.html.
- Yatin Dandi, Florent Krzakala, Bruno Loureiro, Luca Pesce, and Ludovic Stephan. How two-layer neural networks learn, one (giant) step at a time, 2023.
- Alberto Bietti, Joan Bruna, and Loucas Pillaud-Vivien. On learning gaussian multi-index models with gradient flow. *arXiv preprint arXiv:2310.19793*, 2023.
- Jimmy Ba, Murat A Erdogdu, Taiji Suzuki, Zhichao Wang, and Denny Wu. Learning in the presence of low-dimensional structure: A spiked random matrix perspective. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 17420–17449. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/38a1671ab0747b6ffe4d1c6ef117a3a9-Paper-Conference.pdf.
- Behrad Moniri, Donghwan Lee, Hamed Hassani, and Edgar Dobriban. A theory of non-linear feature learning with one gradient step in two-layer neural networks, 2023.

Alireza Mousavi-Hosseini, Denny Wu, Taiji Suzuki, and Murat A Erdogdu. Gradient-based feature learning under structured data. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 71449–71485. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/e21955c93dede886af1d0d362c756757-Paper-Conference.pdf.

Aaron Zweig and Joan Bruna. Symmetric single index learning, 2023.

Björn Barz and Joachim Denzler. Do we train on test data? purging cifar of near-duplicates. *Journal of Imaging*, 6(6):41, 2020.

Michael Zhang, James Lucas, Jimmy Ba, and Geoffrey E Hinton. Lookahead optimizer: k steps forward, 1 step back. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/90fd4f88f588ae64038134f1eeaa023f-Paper.pdf.

G. M. Korpelevič. An extragradient method for finding saddle points and for other problems. *Èkonom. i Mat. Metody*, 12(4):747–756, 1976. ISSN 0424-7388.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations*, 2021.

Yatin Dandi, Emanuele Troiani, Luca Arnaboldi, Luca Pesce, Lenka Zdeborová, and Florent Krzakala. The benefits of reusing batches for gradient descent in two-layer networks: Breaking the curse of information and leap exponents. *arXiv preprint arXiv:2402.03220*, 2024.

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, 2021.

Marco Mondelli and Andrea Montanari. Fundamental limits of weak recovery with applications to phase retrieval. In *Conference On Learning Theory*, pages 1445–1450. PMLR, 2018.

Wangyu Luo, Wael Alghamdi, and Yue M Lu. Optimal spectral initialization for signal recovery with applications to phase retrieval. *IEEE Transactions on Signal Processing*, 67(9):2347–2356, 2019.

Antoine Maillard, Bruno Loureiro, Florent Krzakala, and Lenka Zdeborová. Phase retrieval in high dimensions: Statistical and computational phase transitions. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 11071–11082. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/7ec0dbeee45813422897e04ad8424a5e-Paper.pdf.

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

Luca Arnaboldi, Ludovic Stephan, Florent Krzakala, and Bruno Loureiro. From high-dimensional & mean-field dynamics to dimensionless odes: A unifying approach to sgd in two-layers networks. In Gergely Neu and Lorenzo Rosasco, editors, *Proceedings of Thirty Sixth Conference on Learning Theory*, volume 195 of *Proceedings of Machine Learning Research*, pages 1199–1227. PMLR, 12–15 Jul 2023. URL <https://proceedings.mlr.press/v195/arnaboldi23a.html>.

Alex Damian, Loucas Pillaud-Vivien, Jason D Lee, and Joan Bruna. The computational complexity of learning gaussian single-index models. *arXiv preprint arXiv:2403.05529*, 2024.

David Saad and Sara A. Solla. On-line learning in soft committee machines. *Physical Review E*, 52(4):4225–4243, October 1995. doi: 10.1103/PhysRevE.52.4225.

A. C. C. Coolen and D. Saad. Dynamics of learning with restricted training sets. *Phys. Rev. E*, 62:5444–5487, Oct 2000. doi: 10.1103/PhysRevE.62.5444. URL <https://link.aps.org/doi/10.1103/PhysRevE.62.5444>.

A. C. C. Coolen, D. Saad, and Yuan-Sheng Xiong. On-line learning from restricted training sets in multilayer neural networks. *Europhysics Letters*, 51(6):691, sep 2000. doi: 10.1209/epl/i2000-00394-5. URL <https://dx.doi.org/10.1209/epl/i2000-00394-5>.

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. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 37932–37946. Curran Associates, Inc., 2022.

Alberto Bietti, Joan Bruna, Clayton Sanford, and Min Jae Song. Learning single-index models with shallow neural networks. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 9768–9783. Curran Associates, Inc., 2022.

Ryan O’Donnell. *Analysis of Boolean Functions*. Cambridge University Press, Cambridge, 2014. ISBN 9781107038325. doi: 10.1017/CBO9781139814782.

Jean Barbier, Florent Krzakala, Nicolas Macris, Léo Miolane, and Lenka Zdeborová. Optimal errors and phase transitions in high-dimensional generalized linear models. *Proceedings of the National Academy of Sciences*, 116(12):5451–5460, 2019.

Michael Celentano, Chen Cheng, and Andrea Montanari. The high-dimensional asymptotics of first order methods with random data. *arXiv:2112.07572*, 2021.

Yiwen Kou, Zixiang Chen, Quanquan Gu, and Sham M. Kakade. Matching the statistical query lower bound for k-sparse parity problems with stochastic gradient descent, 2024.

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

Behrooz Ghorbani, Song Mei, Theodor Misiakiewicz, and Andrea Montanari. When do neural networks outperform kernel methods? In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 14820–14830. Curran Associates, Inc., 2020.

Rishabh Dudeja and Daniel Hsu. Learning single-index models in gaussian space. In Sébastien Bubeck, Vianney Perchet, and Philippe Rigollet, editors, *Proceedings of the 31st Conference On Learning Theory*, volume 75 of *Proceedings of Machine Learning Research*, pages 1887–1930. PMLR, 06–09 Jul 2018. URL <https://proceedings.mlr.press/v75/dudeja18a.html>.

Alexander Atanasov, Blake Bordelon, and Cengiz Pehlevan. Neural networks as kernel learners: The silent alignment effect. In *International Conference on Learning Representations*, 2022.

Leonardo Petrini, Francesco Cagnetta, Eric Vanden-Eijnden, and Matthieu Wyart. Learning sparse features can lead to overfitting in neural networks. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*. URL http://papers.nips.cc/paper_files/paper/2022/hash/3d3a9e085540c65dd3e5731361f9320e-Abstract-Conference.html.

Luca Arnaboldi, Yatin Dandi, Florent Krzakala, Bruno Loureiro, Luca Pesce, and Ludovic Stephan. Online learning and information exponents: The importance of batch size and time/complexity tradeoffs. In *International Conference on Machine Learning*, 2024.

A Proof of the Main Results

A.1 Proof of Theorem 2

We formalize here the proof sketch of Theorem 2. For simplicity, we assume that $a_0 = 1$. We track the dynamics of \mathbf{w}_t through the following sufficient statistic:

$$m_t := \langle \mathbf{w}_t, \mathbf{w}^* \rangle. \quad (14)$$

We assume in the following that $m_0 > 0$, so that $\epsilon = 1$ in Assumption 3. We will discuss the actual choice of ϵ in the course of the proof.

We define the following quantities:

$$\Phi(\mathbf{z}) = \nabla_{\mathbf{w}} \mathcal{L}(f(\mathbf{z}; \tilde{\mathbf{w}}(\rho), a_0), y) \quad (15)$$

$$\Psi(x, x^*) = h^*(x^*) \sigma'(x + \rho_0 \sigma'(x) h^*(x^*)) x^* \quad (16)$$

$$\phi(m) = \mathbb{E}[\Psi(x, x^*)] \quad \text{for} \quad \begin{pmatrix} x \\ x^* \end{pmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{pmatrix} 1 & m \\ m & 1 \end{pmatrix}\right). \quad (17)$$

Using these definitions, we have for the update equation (11)

$$\mathbf{w}_{t+1} = \frac{\mathbf{w}_t - \gamma \Phi(\mathbf{z}_t)}{\|\mathbf{w}_t - \gamma \Phi(\mathbf{z}_t)\|}$$

This places us under the framework of Ben Arous et al. [2021], replacing $\nabla \mathcal{L}_N$ by Φ (and noting that all conditions in their theorems relate to $\nabla \mathcal{L}_N$ instead of \mathcal{L}_N). We also define the constant $\kappa(\delta)$ such that

$$\mathbb{P}\left(m_0 \geq \frac{\kappa(\delta)}{\sqrt{d}}\right) \geq 1 - \frac{\delta}{2};$$

standard considerations on spherical vectors imply that $\kappa(\delta) = O(1)$ for any choice of δ .

The Lipschitz condition on σ easily implies that Assumption B in Ben Arous et al. [2021] is satisfied, and hence the bounds in Proposition 4.4, as well as the martingale control in Proposition 4.5 thereof, imply the following:

Lemma 2. *There exists a constant $c(\delta)$ such that if $\gamma_0(\delta) \leq c(\delta)$, $m_0 > \kappa(\delta)/\sqrt{d}$ and*

$$T \leq c(\delta) \gamma^{-2} d,$$

we have

$$\mathbb{P}\left(m_t \geq \frac{m_0}{2} + \frac{1}{2} \sum_{s=0}^t \mathbb{E}[\langle \Phi(\mathbf{z}_s), \mathbf{w}^* \rangle] \quad \forall t \leq T\right) \geq 1 - \frac{\delta}{2}$$

It remains to study the above expectation. By the computations done in the proof sketch, we have

$$\langle \Phi(\mathbf{z}_t), \mathbf{w}^* \rangle = h^*(\langle \mathbf{w}^*, \mathbf{z}_t \rangle) \sigma' \left(\langle \mathbf{w}_t, \mathbf{z}_t \rangle + \rho h^*(\langle \mathbf{w}^*, \mathbf{z}_t \rangle) \sigma'(\langle \mathbf{w}_t, \mathbf{z}_t \rangle) \cdot \|\mathbf{z}_t\|^2 \right) \langle \mathbf{w}^*, \mathbf{z}_t \rangle.$$

We show that this is well approximated by $\phi(m_t)$:

Lemma 3. *Deterministically, the following bound holds:*

$$\left| \mathbb{E}[\langle \Phi(\mathbf{z}_t), \mathbf{w}^* \rangle] - \phi(m) \right| \leq \frac{C}{\sqrt{d}} \quad (18)$$

Proof. The distribution of the pair $(\langle \mathbf{w}_t, \mathbf{z}_t \rangle, \langle \mathbf{w}^*, \mathbf{z}_t \rangle)$ is the same as the one of (x, x^*) , so the only difference between both expressions is the replacement of $\|\mathbf{z}_t\|^2$ by d . Since σ'' is bounded, we can write

$$\begin{aligned} \mathbb{E}\left[|\langle \Phi(\mathbf{z}_t), \mathbf{w}^* \rangle - \phi(m)|\right] &\leq C\rho \mathbb{E}\left[h^*(\langle \mathbf{w}^*, \mathbf{z}_t \rangle)^2 \sigma'(\langle \mathbf{w}_t, \mathbf{z}_t \rangle) (\|\mathbf{z}_t\|^2 - d) \langle \mathbf{w}^*, \mathbf{z}_t \rangle\right] \\ &\leq C'\rho \sqrt{\mathbb{E}\left[(\|\mathbf{z}_t\|^2 - d)^2\right]} \\ &\leq \frac{C''}{\sqrt{d}}, \end{aligned}$$

where at the last line we used that $\rho = O(d^{-1})$. \square

Finally, we can obtain the desired difference equation:

Proposition 1. *There exists a constant $c(\delta)$ such that if $\gamma_0(\delta) \leq c(\delta)$, $m_0 > \kappa(\delta)/\sqrt{d}$ and*

$$T \leq c(\delta)\gamma^{-2}d^{-1} =: \bar{T},$$

we have for large enough d

$$\mathbb{P}\left(m_t \geq \frac{m_0}{4} + \frac{1}{2} \sum_{s=0}^t \phi(m) \quad \forall t \leq T\right) \geq 1 - \frac{\delta}{2}$$

Proof. From Lemmas 2 and 3,

$$\mathbb{P}\left(m_t \geq \frac{m_0}{2} + \frac{\gamma}{2} \sum_{s=0}^t \phi(m) - \frac{C\gamma T}{\sqrt{d}} \quad \forall t \leq T\right) \geq 1 - \frac{\delta}{2}.$$

The lemma is therefore proven as soon as

$$\frac{C\gamma T}{\sqrt{d}} \leq \frac{m_0}{4}, \quad \text{or} \quad T \leq C'\gamma^{-1}\sqrt{d}m_0.$$

But since $\sqrt{d}m_0 \geq \kappa(\delta)$, and $\gamma \ll 1$, this condition follows from $T \leq c(\delta)\gamma^{-2}d$ when d is large enough. \square

Proving Theorem 2 will therefore ensue from the study of the function ϕ . In particular, the results of Ben Arous et al. [2021] imply the following theorem:

Theorem 4. *There exists a choice of $\eta > 0$ and ϵ in Assumption 3, depending on ϕ , such that the following holds:*

- If $\phi(0) \neq 0$, then letting $\gamma = \gamma_0(\delta)d^{-1}$, with probability $1 - \delta$,

$$t_\eta^+ \leq C(\delta)d,$$

- If $\phi(0) = 0$ and $\phi'(0) \neq 0$, then letting $\gamma = \gamma_0(\delta)(d\log(d))^{-1}$, with probability $1 - \delta$,

$$t_\eta^+ \leq C(\delta)d\log(d)^2$$

Proof. We first treat the case where $\phi(0) \neq 0$. Upon changing the choice of ϵ , we can assume that $\phi(0)$ has the same sign as m_0 , and is therefore positive without loss of generality. Let $c = \phi(0)/2$; we define η to be a small constant such that $\phi(m) > c$ on $[0, \eta]$. Then, by Proposition 1, we have

$$\mathbb{P}\left(m_0 \geq \frac{\kappa(\delta)}{\sqrt{d}}, \quad m_t \geq \frac{m_0}{4} + \frac{c\gamma}{2}t \quad \forall t \leq \bar{T} \wedge t_\eta^+\right) \geq 1 - \delta \tag{19}$$

As a result, $t_\eta^+ \leq 2\eta/c\gamma$ as long as the latter bound is lower than \bar{T} . The above condition is equivalent to

$$\gamma_0(\delta) \leq \frac{c(\delta) \cdot c\eta}{2},$$

which can be ensured by decreasing $\gamma_0(\delta)$.

The case where $\phi(0)$ is treated similarly: by the discrete Grönwall inequality,

$$\mathbb{P}\left(m_0 \geq \frac{\kappa(\delta)}{\sqrt{d}}, \quad m_t \geq \frac{m_0}{4}e^{c\gamma t} \quad \forall t \leq \bar{T} \wedge t_\eta^+\right) \geq 1 - \delta, \tag{20}$$

and thus $t_\eta^+ \leq C(\delta)\gamma^{-1}\log(d)$, again under the condition that this bound is lower than \bar{T} . This time, the bound is equivalent to

$$\gamma_0(\delta) \leq \frac{c(\delta)}{C(\delta)},$$

which can again be easily ensured. \square

Analysis of ϕ To prove Theorem 2, it remains to show the following proposition:

Proposition 2. *If $\ell_p^* = 1$, then for almost every choice of ρ_0 , we have $\phi(0) \neq 0$. On the other hand, if $\ell_p^* = 2$, then $\phi'(0) \neq 0$ for almost every choice of ρ_0 .*

The main ingredient here is to write $\phi(m) = \phi(m; \rho_0)$ and differentiate w.r.t ρ_0 . We show the following:

Lemma 4. *Under Assumption 3, the function $\phi(m; \rho_0)$ is analytic in ρ_0 . Further, we have*

$$\frac{\partial^k \phi(m; \rho_0)}{\partial \rho_0^k} \Big|_{\rho_0=0} = \mathbb{E} \left[h^*(x^*)^{(k+1)} x^* \sigma^{k+1}(x) \sigma'(x)^k \right] \quad (21)$$

where x, x^* follow the same distribution as in eq. (17).

Proof. The first statement stems from the analyticity of σ , and the fact that it is a property conserved through integration. For the second, we differentiate inside the expectation, to find

$$\frac{\partial \phi}{\partial \rho_0} = \mathbb{E} \left[\sigma'(x) h^*(x^*)^2 \sigma''(x + \rho_0 \sigma'(x) h^*(x^*)) x^* \right];$$

the result follows by induction and taking $\rho_0 = 0$ at the end. \square

We define the following quantities related to Assumption 3:

$$\begin{aligned} u_0^{(k)} &= \mathbb{E} \left[\sigma^{(k)}(x) \sigma'(x)^{k-1} \right] & u_1^{(k)} &= \mathbb{E} \left[x \sigma^{(k)}(x) \sigma'(x)^{k-1} \right] \\ v_0^{(k)} &= \mathbb{E} \left[x^* h^*(x^*)^k \right] & v_1^{(k)} &= \mathbb{E} \left[[(x^*)^2 - 1] h^*(x^*)^k \right] \end{aligned}$$

Then, by Proposition 11.37 from O'Donnell [2014], we have

$$\frac{\partial^k \phi(m; \rho_0)}{\partial \rho_0^k} \Big|_{\rho_0=0} = u_0^{(k)} v_0^{(k)} + u_1^{(k)} v_1^{(k)} \cdot m + o(m)$$

This allows us to show Proposition 2:

Proof. Assume first that $\ell_p^* = 1$. Then $\phi(0; \rho_0)$ is an analytic function of ρ_0 with

$$\frac{\partial^k \phi(0; \rho_0)}{\partial \rho_0^k} \Big|_{\rho_0=0} = u_0^{(k)} v_0^{(k)}.$$

Since $\ell_p^* = 1$, there exists a $k \in \mathbb{N}$ such that $v_0^{(k)} \neq 0$, and by Assumption 2 the coefficient $u_0^{(k)}$ is nonzero for every k . As a result, $\phi(0; \rho_0)$ is an analytic and non-identically zero function of ρ_0 , so it is non-zero for almost every choice of ρ_0 , as requested. The case $k_p = 2$ is done in a similar way, noting that this time

$$\frac{\partial^k \phi'(0; \rho_0)}{\partial \rho_0^k} \Big|_{\rho_0=0} = u_1^{(k)} v_1^{(k)}.$$

\square

A.2 Proof of Lemma 1

Let σ be an analytic function which is not a polynomial. Then for any $k \geq 0$, the function $\sigma^{(k)}$ is also analytic and non-identically zero, and hence so is the function $f_n = \sigma^{(k)} \cdot (\sigma')^k$. Lemma 1 thus ensues from the following result:

Lemma 5. *Let f be an analytic and non-identically zero function. Then*

$$\mathbb{E} [f(x+b)] \neq 0 \quad \text{and} \quad \mathbb{E} [xf(x+b)] \neq 0$$

for almost every b under the Lebesgue measure.

Proof. The function $\psi(b) = \mathbb{E}[f(x+b)]$ is analytic in b , and we have

$$\psi^{(k)}(0) = \mathbb{E}[f^{(k)}(x)] = c_k u_k(f),$$

where $u_k(f)$ is the k -th Hermite coefficient of f and c_k is a nonzero absolute constant. Since f is nonzero, at least one of its Hermite coefficients is nonzero, and ψ is a non-identically zero analytic function. As a result, $\psi(b) \neq 0$ for almost every choice of b .

The other case is handled as the first one, noting that by Stein's lemma

$$\mathbb{E}[xf(x+b)] = \mathbb{E}[f'(x+b)].$$

□

The above shows that for fixed $n \in \mathbb{N}$, the set

$$\mathcal{B}_n = \left\{ b \in \mathbb{R} : \mathbb{E}\left[\sigma^{(n)}(x+b)\sigma'(x+b)^{n-1}\right] = 0 \quad \text{or} \quad \mathbb{E}\left[x\sigma^{(n)}(x+b)\sigma'(x+b)^{n-1}\right] = 0 \right\}$$

has measure 0. Since there is a countable number of such subsets, the set

$$\mathcal{B} = \bigcup_{n \in \mathbb{N}} \mathcal{B}_n$$

also has measure zero, which is equivalent to the statement of Lemma 1.

A.3 Proof of Theorem 3

We decompose the polynomial h^* as an even and odd part $e(x)$ and $o(x)$, with degrees d_o and d_e . We first assume that o is non-zero, and that the leading coefficient of o is positive. Then, for odd $m \geq 0$, we have

$$\mathbb{E}[xh^*(x)^m] = \sum_{k=0}^m \binom{m}{k} \mathbb{E}[xe(x)^k o(x)^{m-k}]$$

Each term in the above sum is zero if k is odd, we focus on the terms where k is even. There exist constants $A, \varepsilon > 0$ such that if $|x| \geq A$,

$$|e(x)| \geq \varepsilon|x|^{d_e} \quad \text{and} \quad |o(x)| \geq \varepsilon|x|^{d_o},$$

and we let

$$B = \sup_{x \in [-A, A]} |e(x)| \vee \sup_{x \in [-A, A]} |o(x)|$$

As a result, when k is even, both $e(x)^k$ and $xo(x)^{m-k}$ are even polynomials with positive leading coefficients, so

$$\begin{aligned} \mathbb{E}[xe(x)^k o(x)^{m-k}] &= \mathbb{E}\left[xe(x)^k o(x)^{m-k} \mathbf{1}_{x \in [-A, A]}\right] + \mathbb{E}\left[xe(x)^k o(x)^{m-k} \mathbf{1}_{x \notin [-A, A]}\right] \\ &\geq \varepsilon^2 \mathbb{E}\left[x^{kd_e + (m-k)d_o + 1} \mathbf{1}_{x \notin [-A, A]}\right] - AB^m \end{aligned}$$

Let $d(m, k) = kd_e + (m-k)d_o + 1$. Then,

$$\mathbb{E}[xe(x)^k o(x)^{m-k}] \geq \varepsilon^2 \mathbb{E}\left[x^{d(m, k)}\right] - A^{d(m, k)} - AB^m.$$

Going back to the sum, and with the crude bound $\binom{m}{k} \leq 2^m$,

$$\begin{aligned} \mathbb{E}[xh^*(x)^m] &\geq \varepsilon^2 \mathbb{E}\left[x^{d(m, 0)}\right] - \sum_{k=1}^m 2^m (A^{d(m, k)} + AB^m) \\ &\geq \mathbb{E}\left[x^{m+1}\right] - C^m \end{aligned}$$

Since the Gaussian moments grow faster than any power of m , this last expression is non-zero for a large enough choice of m .

Finally, if o is the zero polynomial, we can do the same reasoning with $e(x)^m$ to get

$$\mathbb{E}\left[(x^2 - 1)e(x)^m\right] = \varepsilon^2(\mathbb{E}\left[x^{md_e + 2}\right] - \mathbb{E}\left[x^{md_e}\right]) - C^m = \varepsilon^2 md_e \mathbb{E}\left[x^{md_e}\right] - C^m,$$

and we conclude as before.

A.4 Hardness of sign functions

We show in the appendix the following statement:

Proposition 3. Let $m \in \mathbb{N}$, and

$$h^*(\mathbf{x}) = \text{sign}(x_1 \dots x_m)$$

Then the function h^* has Information and Generative Exponents $\ell = \ell^* = m$.

We first show the lower bound. For any vector $\mathbf{v} \in \mathbb{R}^m$, and $k < m$, $H_k(\langle \mathbf{v}, \mathbf{x} \rangle)$ is a polynomial in \mathbf{x} of degree k , and therefore each monomial term is missing at least one variable. We show that this implies the lower bound of Proposition 3 through the following lemma:

Lemma 6. Let $f : \{-1, 1\} \rightarrow \mathbb{R}$ be an arbitrary function, and $g : \mathbb{R}^m \rightarrow \mathbb{R}$ a function which is independent from x_m . Then

$$\mathbb{E}[f(h^*(\mathbf{x}))g(\mathbf{x})] = \frac{f(1) + f(-1)}{2} \mathbb{E}[g(\mathbf{x})]$$

Proof. We simply write

$$\begin{aligned} \mathbb{E}[f(h^*(\mathbf{x}))g(\mathbf{x})] &= \mathbb{E}\left[\mathbb{E}[f(h^*(\mathbf{x}))g(\mathbf{x}) | x_1, \dots, x_{m-1}]\right] \\ &= \mathbb{E}\left[g(\mathbf{x})\mathbb{E}[f(h^*(\mathbf{x})) | x_1, \dots, x_{m-1}]\right] \\ &= \mathbb{E}\left[g(\mathbf{x})\frac{f(1) + f(-1)}{2}\right] \\ &= \frac{f(1) + f(-1)}{2} \mathbb{E}[g(\mathbf{x})], \end{aligned}$$

where at the second line we used that g only depends on x_1, \dots, x_{m-1} and at the third line the addition of x_m randomly flips the sign of h^* . \square

Using this property on every monomial in $H_k(\langle \mathbf{v}, \mathbf{x} \rangle)$ and summing back, we have for any function f

$$\mathbb{E}[f(h^*(\mathbf{x}))H_k(\langle \mathbf{v}, \mathbf{x} \rangle)] = \frac{f(1) + f(-1)}{2} \mathbb{E}[H_k(\langle \mathbf{v}, \mathbf{x} \rangle)] \quad (22)$$

But the latter is zero when $k \geq 1$ by orthogonality of the Hermite polynomials. This shows that $\ell_v^* \geq m$ for all $\mathbf{v} \in \mathbb{R}^k$, and therefore $m \leq \ell^* \leq \ell$.

For the upper bound, let $\mathbf{v} = \mathbf{e}_1 + \dots + \mathbf{e}_m$. Then

$$H_k(\langle \mathbf{v}, \mathbf{x} \rangle) = m! x_1 \dots x_m + Q(\mathbf{x}),$$

where Q is a polynomial where each monomial has a missing variable. As a result,

$$\begin{aligned} \mathbb{E}[h^*(\mathbf{x})H_k(\langle \mathbf{v}, \mathbf{x} \rangle)] &= m! \mathbb{E}[h^*(\mathbf{x})x_1 \dots x_m] \\ &= m! \mathbb{E}[|x_1 \dots x_m|] \\ &= m! \left(\sqrt{\frac{2}{\pi}}\right)^m \end{aligned}$$

which is a non-zero value, hence $\ell^* \leq \ell \leq m$.

B Additional Numerical Investigation

In this section, we present further numerical investigation to support our theory and extend it beyond the mathematical hypotheses used in the formal proof.

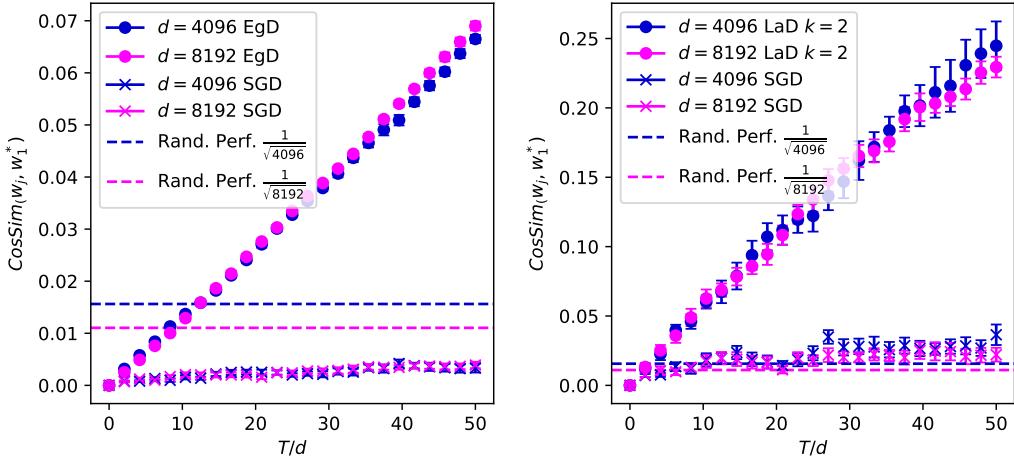


Figure 5: example of two different algorithm with data repetitions learning an hard single-index target $h^*(z) = \text{He}_3(z_1)$, $\ell = 3$, $\ell^* = 1$. **Left:** SAM, with $\rho_0 = 0.1, \gamma_0 = 0.01$. **Right:** 2-Lookahed with $\gamma_0 = 0.1$. See details in App. C.

B.1 Testing wider range of Algorithm 1

In the main we presented all the numerical simulation using $\rho > 0$, i.e. *ExtraGradient method/descent*. In Figure 5 Left, we repeat one of the experiment of Figure 1 in the case $\rho < 0$, the *Sharpe Aware Minimization*. The two algorithms are based on very different principle, but in our context they behave in the same way: the reusing of the data makes high information exponent function easily learnable. The global picture for $\rho \neq 0$ is the same regardless of its sign.

Altough Algorithm 1 includes a broad variety of algorithm, it is not the most general domain of validity of our theory. Simple algorithms such as repeating the gradient step twice on the same data before discarding it (also known as *2-lookahead optimizer* [Zhang et al., 2019]) are not part of Algorithm 1, yet our considerations are still valid. In fact, the data repetition is sufficient to introduce correlations across two consecutive steps, whose effect builds up along the dynamics causing the network to learn hard targets. A simple illustration of this claim is available in Figure 5 Right.

B.2 Extensive batch size

In this section, we want to present some evidence that our result does not change when the batch size is $1 < n_b \leq O(d^{\frac{\ell^*}{2}})$. The formal proof we presented in the paper is valid in the case where the batch size is $n_b = 1$, but we strongly believe it is the case when training with larger batches.

In the context of the information exponent, it has been shown in Arnaboldi et al. [2024] that the total sample complexity does not change when using batch sizes $n_b \leq O(d^{\frac{\ell}{2}})$. The heuristic argument for this behavior is that using a larger batch size increases the learning rate since the noise of the gradient is “more” averaged out, ultimately reducing the number of steps. The same argument should pass from information exponent to generative exponent, allowing the claim (when $n_b \leq O(d^{\frac{\ell^*}{2}})$)

$$T \cdot n_b \sim \begin{cases} O(d^{\ell-1}) & \text{if } \ell^* > 2 \\ O(d \log d) & \text{if } \ell^* = 2 \\ O(d) & \text{if } \ell^* = 1 \end{cases} \quad (23)$$

Note that this drastically reduces the time steps needed to learn the teacher’s directions, although the sample complexity does not change. Dandi et al. [2024] first shows that some high-information exponent functions, such as He_3 , can be learned in just a few steps with full-batch gradient descent $n_b = O(d)$. In the same work, they introduce a larger class of hard functions that cannot be recovered in $T = O(1)$, not providing any information on possible bounds on the number of time steps, nor any claim suggesting that not all the function in the class have the same sample complexity. Our theory can provide a full understanding of the phenomena:

- He_3 has generative exponent $\ell^* = 1$, that implies we need $O(d)$ samples to weakly recover the target; when training with batch size $n_b = O(d)$, we have $T = O(1)$ as in Dandi et al. [2024]; Figure 6 (Left) is a numerical

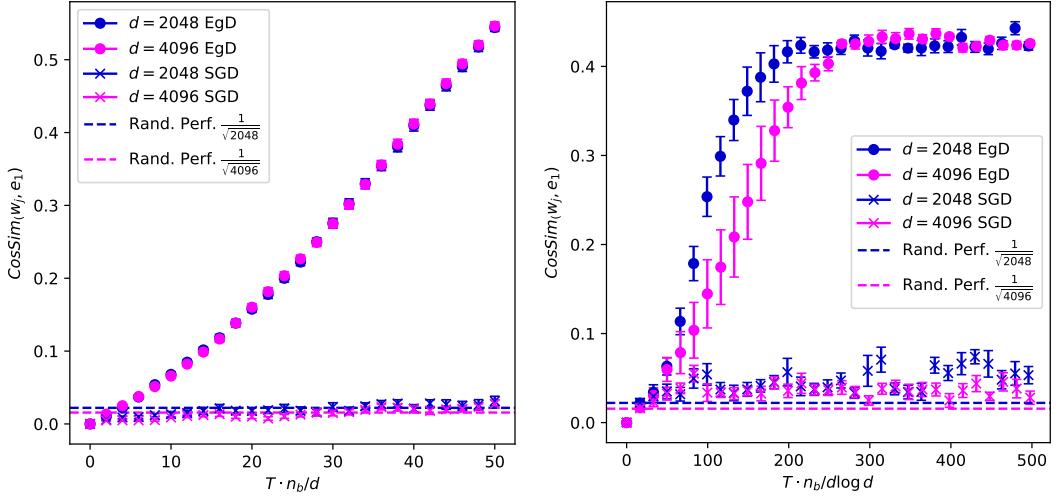


Figure 6: Single-index recovery for large-batch training. The dashed horizontal line $\frac{1}{\sqrt{d}}$ is a visual guide to place random performance. The plots show $\sigma = \text{relu}$, $\gamma = 0.01$, $\rho = 0.1$, and the performance is computed averaging across 10 different runs. **Left** $(\ell, \ell^*) = (3, 1)$: $h^* = \text{He}_3$. **Right** $(\ell, \ell^*) = (4, 2)$: $h^* = \text{He}_4$. See details in App. C.

proof of this fact.

- He_4 has generative exponent $\ell^* = 2$, that implies we need $O(d \log d)$ samples to weakly recover the target; when training with batch size $n_b = O(d)$, we have $T = O(\log d)$; Figure 6 (Right) is a numerical proof of this fact.

We can push our claims beyond single-index targets and illustrate the same phenomena for multi-index functions. In Figure 7 we test Extragradient Descent against two hard multi-index functions: once again the sample complexity is the same as the case $n_b = 1$, while the number of steps is reduced by a factor $n_b = d$.

C Implementation Details

In this section, we provide all the implementation details needed to reproduce the Figures of the paper. The full detailed implementation can be repository linked in the main paper.

Unless explicitly stated, we always run the plain version of Algorithm 1 with squared loss, as opposed to Assumption 1 which requires *correlation loss* and spherical gradient. Also, the activation function we always use is ReLU, which does not satisfy Assumption 2, but is closer to real cases. Finally, we always run simulations with $W^0 \perp W^*$, which does not fall under Assumption 3. Indeed, our numerical simulations deviate from the theoretical assumptions needed for the formal proofs, but the aim here is to show that our claims are valid beyond the technical limitations of the theory and that they can allow us to understand settings closer to what is used in practice.

In the plots with a single-index target we plot the *absolute value* of the cosine similarity between the student weight \mathbf{w} and the teacher weight \mathbf{w}^* , averaged across N different runs (where we vary both seen samples and initial conditions). If q in the index for different runs:

$$\text{yaxis}(t) = \frac{1}{N} \sum_{q=1}^N \left| \text{CosSim}(\mathbf{w}_t^{(q)}, \mathbf{w}^{*(q)}) \right|,$$

where

$$\text{CosSim}(\mathbf{w}, \mathbf{w}^*) = \frac{\mathbf{w} \cdot \mathbf{w}^*}{\|\mathbf{w}\|_2 \|\mathbf{w}^*\|_2}.$$

In the plots with a multi-index target, we take the maximum across all the network weights

$$\text{yaxis}(t) = \frac{1}{N} \sum_{q=1}^N \max_{j \in [p]} \left| \text{CosSim}(\mathbf{w}_{j,t}^{(q)}, \mathbf{w}^{*(q)}) \right|.$$

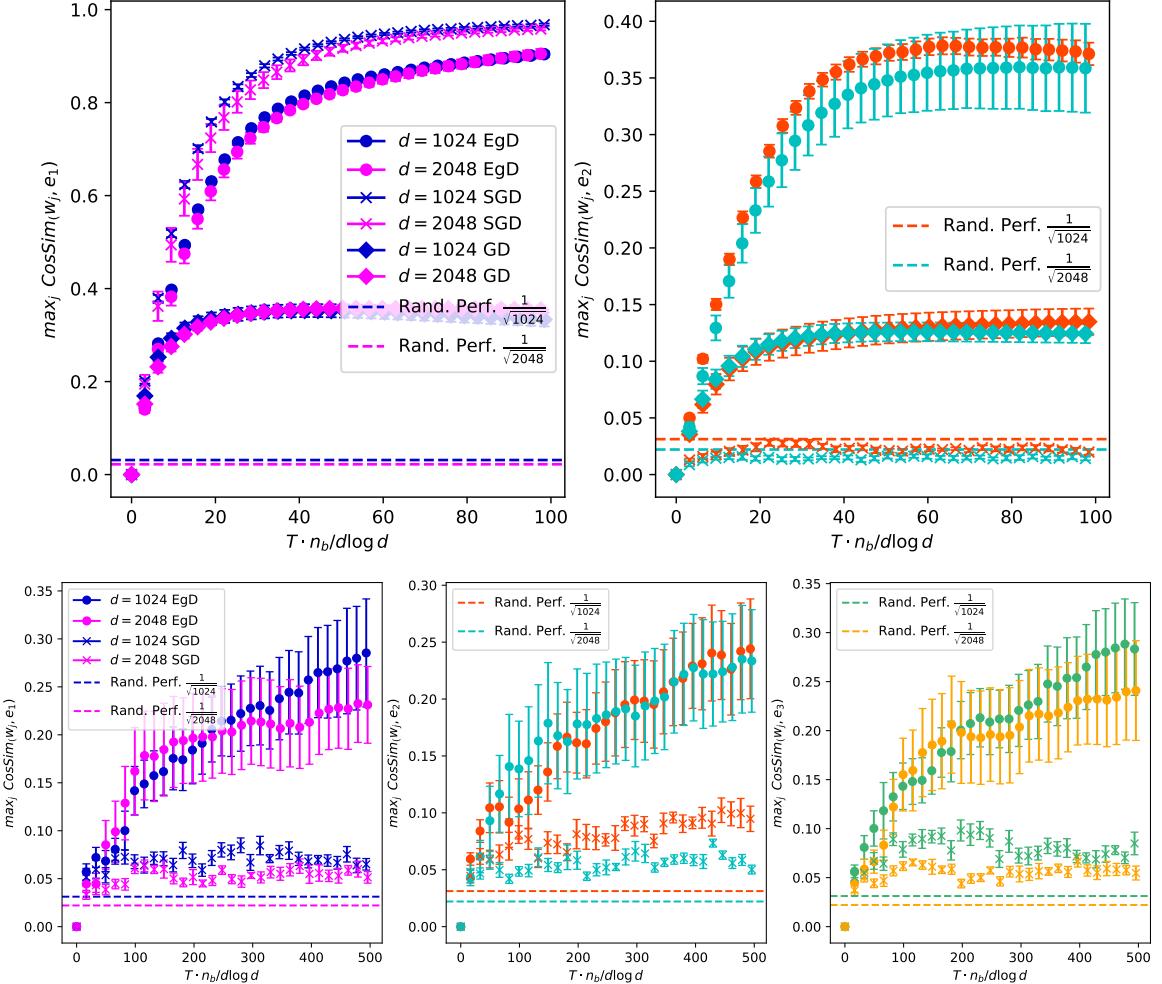


Figure 7: Multi-index recovery for large-batch training. The dashed horizontal line $\frac{1}{\sqrt{d}}$ is a visual guide to place random performance. The plots show $\sigma = \text{relu}$. **Upper** (ℓ_v, ℓ_v^*) = (3, 1): $h^*(\mathbf{z}) = \text{sign}(z_1 z_2)$. **Lower** (ℓ_v, ℓ_v^*) = (4, 2): $h^*(\mathbf{z}) = z_1 z_2 z_3$. See Appendix C for details.

C.1 Hyperparameters of the Figures

Here are the hyperparameters used in the figures:

- **Figure 1 (all 4):** $d = 8192, \gamma_0 = 0.01, \rho_0 = 0.1, N = 40$.
- **Figure 2 (left):** $p = 8, d = 2048, \gamma_0 = 0.01, \rho_0 = 0.1, N = 40, a_j^0 \sim \text{Rad}(1/2)$.
- **Figure 2 (center-left):** $p = 8, d = 2048, \gamma_0 = 0.01, \rho_0 = 0.1, N = 40, a_j^0 \sim \mathcal{N}(0, 1)$.
- **Figure 2 (center-right):** $p = 8, d = 2048, \gamma_0 = 0.1, \rho_0 = 0.1, N = 40, a_j^0 \sim \mathcal{N}(0, 1)$.
- **Figure 2 (right):** $p = 8, d = 2048, \gamma_0 = 0.1, \rho_0 = 0.1, N = 20, a_j^0 \sim \mathcal{N}(0, 1)$.
- **Figure 3 (left):** $p = 8, d = 2048, \gamma_0 = 0.1, \rho_0 = 0.1, N = 40, a_j^0 \sim \mathcal{N}(0, 1)$.
- **Figure 3 (right):** $p = 4, d = 1024, \gamma_0 = 0.01, \rho_0 = 0.1, N = 10, a_j^0 \sim \mathcal{N}(0, 1)$.
- **Figure 4:** $p = 4, d = 1024, \gamma_0 = 0.01, \rho_0 = 0.1, N = 10, a_j^0 \sim \mathcal{N}(0, 1)$.
- **Figure 5 (left):** SAM $d = 4096, 8192, \gamma_0 = 0.01, \rho_0 = 0.1, N = 40$.
- **Figure 5 (right):** 2-Lookahead $d = 4096, 8192, \gamma_0 = 0.01, \rho_0 = 0.1, N = 40$.
- **Figure 6 (all 2):** $d = 8192, n_b = 2d, \gamma_0 = 0.1, N = 40$.
- **Figure 7 (upper):** $p = 8, d = 2048, n_b = 2d, \gamma_0 = 0.1, \rho_0 = 0.1, N = 40, a_j^0 \sim \mathcal{N}(0, 1)$.
- **Figure 7 (lower):** $p = 8, d = 2048, n_b = 2d, \gamma_0 = 0.1, \rho_0 = 0.1, N = 20, a_j^0 \sim \mathcal{N}(0, 1)$.