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

Luca Arnaboldi LUCA.ARNABOLDI@EPFL.CH
Ecole Polytechnique Fédérale de Lausanne, Information, Learning and Physics lab. CH-1015 Lausanne, Switzerland.

Yatin Dandi YATIN.DANDI@EPFL.CH
Ecole Polytechnique Fédérale de Lausanne, Statistical Physics of Computation Laboratory. CH-1015 Lausanne, Switzerland. Ecole Polytechnique Fédérale de Lausanne, Information, Learning and Physics lab. CH-1015 Lausanne, Switzerland.

Florent Krzakala FLORENT.KRZAKALA@EPFL.CH
Ecole Polytechnique Fédérale de Lausanne, Information, Learning and Physics lab. CH-1015 Lausanne, Switzerland.

Luca Pesce LUCA.PESCE@EPFL.CH
Ecole Polytechnique Fédérale de Lausanne, Information, Learning and Physics lab. CH-1015 Lausanne, Switzerland.

Ludovic Stephan LUDOVIC.STEPHAN@EPFL.CH
Ecole Polytechnique Fédérale de Lausanne, Information, Learning and Physics lab. 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 preprocessing. 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 [16, 30, 36, 38]). 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 [11] and leap exponent [2, 3]. These analyses have sparked many follow-up theoretical works over the last few months, see, e.g. [8, 13, 17, 19, 20, 32, 33, 40], 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 [10]) 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 [39], Extragradient methods [26], or Sharpness Aware Minimization (SAM) [23], 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 [21] that iterating twice over large ($O(d)$) datasets was enough to alter the picture reached in [2, 3, 11]. Indeed [21] 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 [1, 11] 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 [39], Extragradient methods [26], or in the context of Sharpness Aware Minimization (SAM) [23], 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., [15, 28, 29, 31]). Our results thus demonstrate that shallow neural networks turns out to be significantly more powerful than previously believed [2–4, 11] 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) [18] (rather than the more limited correlation statistical queries (CSQ) bounds that were characteristic of the former consensus [3, 11]). 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 [11, 37].

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 $\sigma : \mathbb{R} \to \mathbb{R}$

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

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

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

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:

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

where $\mathcal{L} : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ 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 $\mathcal{A} : (\mathbf{w}_{j,t}, \gamma, \rho, n_b) \in \mathbb{R}^{d+3} \to \mathbf{w}_{t+1,j} \in \mathbb{R}^d$.

```
Algorithm 1: Optimizer Main Step
for $j \in [p]$ do
    $1_{\mathrm{st}}$ gradient computation –   $\tilde{\mathbf{w}}_{t,j}(\rho) = \mathbf{w}_{t,j} - \rho\nabla_{\mathbf{w}_{t,j}}\mathcal{L}\left(f(\mathbf{z}^\nu; W_t, \mathbf{a}_0), y^\nu\right),\ \forall\nu \in [n_b]$
    $2_{\mathrm{nd}}$ gradient computation –   $\mathbf{g}^\nu = \nabla_{\mathbf{w}_{t,j}}\mathcal{L}\left(f(\mathbf{z}^\nu; \tilde{W}_t(\rho), \mathbf{a}_0), y^\nu\right),\ \forall\nu \in [n_b]$
    Agglomerating step –   $\mathcal{A}(\mathbf{w}_t, \gamma, \rho, n_b) = \mathbf{w}_{t,j} - \frac{\gamma}{n_b}\sum_{\nu\in[n_b]}\mathbf{g}^\nu$
    Update Weights –   $\mathbf{w}_{t+1,j} = \mathcal{A}(\mathbf{w}_t, \gamma, \rho, n_b)$
end
```

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) [26], while with negative ρ have been recently used in the context of Sharpness Aware Minimization (SAM) [23]. 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 (D, E) 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 $W^\star$. 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., [7, 12, 19]).

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 $W^\star$, 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 $W^\star$, known as *weak recovery*.

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

$$V^\star = \mathrm{span}(\mathbf{w}_1^\star, \dots, \mathbf{w}_k^\star) \tag{4}$$

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

$$t_\eta^+ = \min\{t \geq 0 : \|W W^{\star\top}\|_F \geq \eta\} \tag{5}$$

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 [11] for the single-index case, then [2] 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 $V^\star$, defined in the single-index case as

$$\ell = \min\{k \in \mathbb{N} \,:\, \mathbb{E}_{x\sim\mathcal{N}(0,1)}\left[h^\star(x) H_k(x)\right] \neq 0\}. \tag{6}$$

where H_k is the k-th Hermite polynomial [34]. Under this framework, the sample complexity required to achieve weak recovery when the link function has Information Exponent ℓ is [11]

$$T(\ell) = \begin{cases} O(d^{\ell-1}) & \text{if } \ell > 2 \\ O(d \log d) & \text{if } \ell = 2 \\ O(d) & \text{if } \ell = 1. \end{cases} \tag{7}$$

These bounds have been improved by [17] up to order $d^{\ell/2}$; the latter matches the so-called *Correlated Statistical Query* lower bound, which considers queries of the form $\mathbb{E}\left[y\phi(\mathbf{z})\right]$.

However, this Information Exponent is not the right measure of complexity for low-dimensional weak recovery. Indeed, [18] introduce a new Generative Exponent $\ell^\star$ (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

$$\ell^\star = \min\{k \in \mathbb{N} \,:\, \mathbb{E}_{x\sim\mathcal{N}(0,1)}\left[f(h^\star(x))H_k(x)\right] \neq 0 \text{ for some } f:\mathbb{R}\to\mathbb{R}\}, \tag{8}$$

which allows for the application of arbitrary transformations of the labels before performing the queries. Under this definition, the "hard" problems with $\ell^\star > 2$ are also impossible to learn for the best first-order algorithm, *Approximate Message Passing* [9, 14, 28, 29, 31].

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 [27] on the computational efficiency of SGD in learning sparse $k-$parity (associated with $\ell^\star = k$).

When the optimal transformation f is known, a simple SGD algorithm on $(\mathbf{z}, f(y))$ 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 $h^\star$, the situation drastically changes when multiple-pass algorithms are considered. Recently, [21] 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 2 (Informal) *There is a choice of hyperparameters such that if $h^\star$ 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. The formal results is stated in Appendix B with a concise proof scheme. The detailed proof can be found in Appendix. D.

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

[1] 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.

[2] 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, 2022.

[3] 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.

[4] 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.

[5] 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.

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

[7] 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.

[8] 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.

[9] 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.

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

[11] 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.

[12] 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.

[13] Alberto Bietti, Joan Bruna, and Loucas Pillaud-Vivien. On learning gaussian multi-index models with gradient flow. *arXiv preprint arXiv:2310.19793*, 2023.

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

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

[16] 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.

[17] 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.

[18] 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.

[19] 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.

[20] Yatin Dandi, Florent Krzakala, Bruno Loureiro, Luca Pesce, and Ludovic Stephan. How two-layer neural networks learn, one (giant) step at a time, 2023.

[21] 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.

[22] 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.

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

[24] 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.

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

[26] 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.

[27] 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.

[28] 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.

[29] 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.

[30] 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.

[31] 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.

[32] 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.

[33] 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.

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

[35] 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*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/3d3a9e085540c65dd3e5731361f9320e-Abstract-Conference.html.

[36] 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.

[37] 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.

[38] 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.

[39] 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.

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

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 [11].
- We unveil that the dynamics of the family of Algorithm 1 are governed by the generative exponent introduced in [18] 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 [2].
- 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., [28, 29, 31] for single index targets or [15] 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 [1] 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 anonymized Github Repository.

Appendix A. Further Related Works

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 [25] 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. [24]). 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., [6, 12, 19, 22, 35]). 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 E for the extension to larger batch sizes.

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 [15, 28, 29, 31]. 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 $W^\star$, i.e., the 'coldest' possible start.

Appendix B. Single-Index Model

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

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

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 the learning efficiency of the algorithmic family 1 and One-Pass SGD, limited by the Information Exponent of the target to be learned [11].

We start by introducing a restriction of the generative information exponent in [18] to polynomial transformations.

Definition 3 (Polynomial Generative Information exponent) *We define $\ell_p^\star$ as the smallest integer k such that there exists a polynomial $p : \mathbb{R} \to \mathbb{R}$ with:*

$$\mathbb{E}_{x\sim\mathcal{N}(0,1)}\left[p(h^\star(x))H_k(x)\right] \neq 0, \tag{9}$$

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

$$\mathbf{w}_{t+1} = \frac{\mathbf{w}_t - \gamma\nabla^\perp_{\mathbf{w}}\mathcal{L}\left(f(\mathbf{z}^\nu; \tilde{\mathbf{w}}_t(\rho), a_0), y^\nu\right)}{\|\mathbf{w}_t - \gamma\nabla^\perp_{\mathbf{w}}\mathcal{L}\left(f(\mathbf{z}^\nu; \tilde{\mathbf{w}}_t(\rho), a_0), y^\nu\right)\|} \tag{10}$$

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*

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

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

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

where ϵ depends on $h^\star$ and σ.

We are now ready to state our main result.

Theorem 4 *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 $\ell_p^\star = 1$, choosing $\gamma = \gamma_0(\delta)d^{-1}$, then $\mathbb{P}(t_\eta^+ \leq C(\delta)\cdot d) \geq 1 - \delta$,*
- *If $\ell_p^\star = 2$, 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 4. For any vector $\mathbf{w}$, under the correlation loss, we have

$$\nabla_{\mathbf{w}} \mathcal{L}(f(\mathbf{z}; \mathbf{w}, a_0), y) = a_0 y \sigma'(\langle \mathbf{w}, z \rangle) \cdot \mathbf{z}, \tag{11}$$

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

$$\nabla_{\mathbf{w}} \mathcal{L}\left(f(\mathbf{z}; \tilde{\mathbf{w}}(\rho), a_0), y\right) = a_0 y \sigma'\big(\langle \mathbf{w}, \mathbf{z} \rangle + a_0 \rho y \sigma'(\langle \mathbf{w}, \mathbf{z} \rangle) \cdot \underbrace{\|\mathbf{z}\|^2}_{\approx d}\big) \cdot \mathbf{z} \tag{12}$$

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 (12) is a *non-linear* function of y, as opposed to the linear function of (11).

The latter property enables us to show that the dynamics driven by equation (12) 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 [11], but with an information exponent equal to $\ell_p^\star$.

B.1. 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 $\mathbf{w}^\star$. We expect however Thm.4 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 5 *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 [11], 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 [18], we show that it is enough to allow learning of a large class of functions, namely polynomials:

Theorem 6 *Assume that $h^\star$ is a polynomial function. Then the Polynomial Generative Information exponent of $h^\star$ is always at most 2, and is equal to 1 whenever $h^\star$ is not even.*

This theorem implies that the Extragradient algorithm class allows us to match the sample complexity of the algorithm of [15], without the need for *ad hoc* preprocessing.

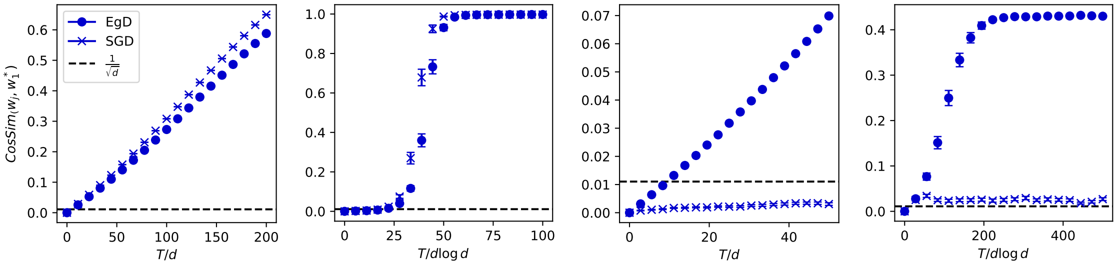

Figure 1: **Learning single-index targets –** Evolution of the Cosine Similarity attained by EgD (crosses) and SGD (dots) as a function of normalized iteration time. The dashed horizontal line $\frac{1}{\sqrt{d}}$ is a visual guide to place random performance. For details on the target functions and parameters used for the simulation, see Section B.2 and App. F.

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 $\mathbf{z}^\nu$, 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.

B.2. 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 $\mathbf{w}_\star$ 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 $\ell^\star$ are given in Thm. 4, while the mini-batch size is fixed to $n_b = 1$ (See App. E 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 F.

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

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

SGD-hard non-symmetric targets ($\ell > 2, \ell^\star = 1$) – The Center-Right section of Fig. 1 shows $h^\star(x) = \mathrm{He}_3(x)$. Here Algortihm 1 learns in $T = O(d)$ while One Pass SGD suffers from the limitations detailed in eq. (7), i.e. $T = O(d^{\ell-1})$ with $\ell = 3$ for this case.

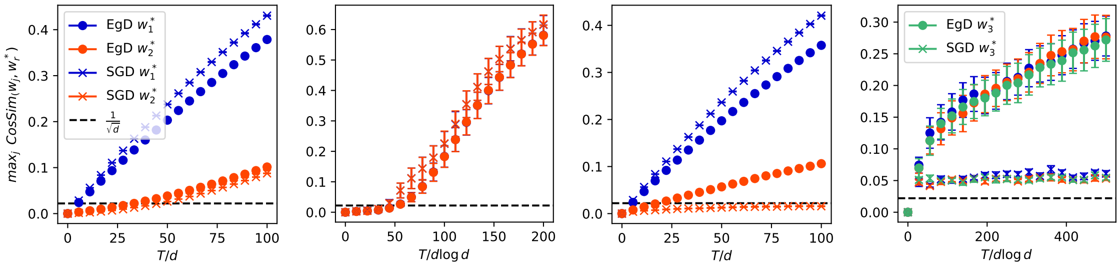

Figure 2: **Learning multi-index targets –** Evolution of the maximum Cosine Similarities attained by EgD (crosses) and SGD (dots) as a function of the normalized iteration time. Different target directions $\{\mathbf{w}_r^\star\}_{r\in[k]}$ are identified by different colors: $\mathbf{w}_1^\star$ (blue), $\mathbf{w}_2^\star$ (orange), $\mathbf{w}_3^\star$ (green). The dashed horizontal line $\frac{1}{\sqrt{d}}$ is a visual guide to place random performance. For details on the target functions and parameters used for the simulation, see Section C.1 and App. F.

SGD-hard symmetric targets ($\ell > 2, \ell^\star = 2$) – The Right section of Fig. 1 shows $h^\star(x) = \mathrm{He}_4(x)$. Here Algortihm 1 learns in $T = O(d \log d)$ while One Pass SGD suffers from the time scaling law detailed in eq. (7) $T = O(d^{\ell-1})$.

Appendix C. 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 $V^\star$. To quantify this behavior, we define directional versions of the Information and Generative Exponents:

Definition 7 *Let $\mathbf{v} \in V^\star$. The Information Exponent $\ell_{\mathbf{v}}$ (resp. Generative exponent $\ell^\star_{\mathbf{v}}$) of y in the direction $\mathbf{v}$ is the smallest k such that*

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

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

The identification of the class of hard functions has been subject of intense theoretical scrutiny [1–3, 13, 20]; 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. (7) 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. 7).

C.1. Numerical investigation

We measure the maximal cosine similarity between the hidden layer neurons and the different rows of the target's weight matrix $W^\star$ 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 F for the implementation details.

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

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

Non-symmetric targets with SGD-easy and hard directions ($\ell_{\mathbf{v}} > 2, \ell^\star_{\mathbf{v}} = 1$) – Fig. 2, center-right, shows $h^\star(x_1, x_2) = x_1 + \mathrm{He}_3(x_2)$. Here both SGD and Alg. 1 learn in $T = O(d)$ the direction $\mathbf{w}_1^\star$, that satisfies $\ell_{\mathbf{w}_1^\star} = \ell^\star_{\mathbf{w_1}^\star} = 1$. However, since $\ell_{\mathbf{w}_2^\star} = 3$ while $\ell^\star_{\mathbf{w}_2^\star} = 1$, One-Pass SGD suffers from the limitations detailed in eq. (7) and requires $\Omega(d^2)$ samples to learn the second direction $\mathbf{w}_2^\star$, . This contrasts with the behaviour of Alg. 1 that learns also $\mathbf{w}_2^\star$ in $T = O(d)$ steps.

SGD-hard symmetric targets ($\ell > 2, \ell^\star = 2$) – The right section Fig. 2 shows $h^\star(x_1, x_2, x_3) = x_1 x_2 x_3$. 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 $\{\mathbf{w}_1^\star, \mathbf{w}_2^\star, \mathbf{w}_3^\star\}$ since the Generative Exponents $\ell^\star_{\mathbf{w}_r^\star}$ are all equal to two.

C.2. Learning hard functions through a hierarchical mechanism

The investigation of the hierarchical nature of SGD learning has attracted noticeable attention [1–3, 20]. 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 [1], i.e. $h^\star(\mathbf{x}) = x_1 + x_1 \mathrm{He}_3(x_2)$. While one-pass SGD needs to learn hierarchically first the direction $\mathbf{w}_1^\star$ in $T = O(d)$ and then $\mathbf{w}_2^\star$ 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, $h^\star_{\text{sign}}(\mathbf{x}) = \mathrm{sign}(x_1 x_2 x_3)$ and $h^\star_{\text{stair}}(\mathbf{x}) = \mathrm{He}_2(x_1) + \mathrm{sign}(x_1 x_2 x_3)$. We show in App. D that $3-$sparse parities of the form $\mathrm{sign}(x_1 x_2 x_3)$ are hard 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 $h^\star_{\text{stair}}$, our simulations predict that the first direction $\mathbf{w}_1^\star$ is learned in $T = O(d \log d)$ iterations (see the rightmost panel of Fig. 2). However, once the direction $\mathbf{w}_1^\star$ is learned, it can be used to obtain order-one correlation with $\{\mathbf{w}_2^\star, \mathbf{w}_3^\star\}$ again in $T = O(d \log d)$ steps. While the latter example belongs to the class of "CSQ" staircase function depicted in [1], novel "SQ" hierarchical mechanisms arise in our framework, such as for instance the function $h^\star(\mathbf{x}) = \mathrm{He}_4(x_1) + \mathrm{sign}(x_1 x_2 x_3)$. We refer to Appendix E for additional discussion (Fig. 7) on this interesting phenomenon.

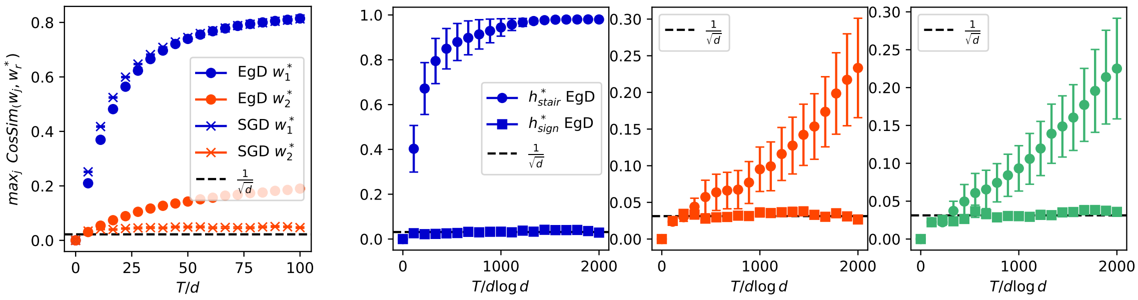

Figure 3: **Hierarchical picture of learning – Left:** Evolution of the maximum Cosine Similarities attained by EgD (crosses) and SGD (dots) as a function of the normalized iteration time. **Right:** Evolution of the maximum Cosine Similarities obtained by EgD for the target function $h^\star_{\text{stair}}$ (dots) and $h^\star_{\text{sign}}$ (squares). Different target directions $\{\mathbf{w}^\star_r\}_{r\in[k]}$ are identified by different colors: $\mathbf{w}^\star_1$ (blue), $\mathbf{w}^\star_2$ (orange), $\mathbf{w}^\star_3$ (green), respectively plotted from left to right. For details on the target functions and parameters used for the simulation, see Section C.2 and App. F.

Appendix D. Proof of the Main Results

D.1. Proof of Theorem 4

We formalize here the proof sketch of Theorem 4. For simplicity, we assume that $a_0 = 1$. We track the dynamics of $\mathbf{w}_t$ through the following sufficient statistic:

$$m_t := \langle \mathbf{w}_t, \mathbf{w}^\star \rangle. \tag{13}$$

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:

$$\Phi(\mathbf{z}) = \nabla_{\mathbf{w}} \mathcal{L}\left(f(\mathbf{z}; \tilde{\mathbf{w}}(\rho), a_0), y\right) \tag{14}$$

$$\Psi(x, x^\star) = h^\star(x^\star)\sigma'(x + \rho_0 \sigma'(x) h^\star(x^\star)) x^\star \tag{15}$$

$$\phi(m) = \mathbb{E}[\Psi(x, x^\star)] \quad \text{for} \quad \begin{pmatrix} x \\ x^\star \end{pmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{pmatrix} 1 & m \\ m & 1 \end{pmatrix}\right). \tag{16}$$

Using these definitions, we have for the update equation (10)

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

This places us under the framework of [11], 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 [11] 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 8 *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

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

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

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

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

Lemma 9 *Deterministically, the following bound holds:*

$$|\mathbb{E}\left[\langle \Phi(\mathbf{z}_t), \mathbf{w}^\star \rangle\right] - \phi(m)| \leq \frac{C}{\sqrt{d}} \tag{17}$$

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

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

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

Finally, we can obtain the desired difference equation:

Proposition 10 *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 8 and 9,

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

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. ■

Proving Theorem 4 will therefore ensue from the study of the function ϕ. In particular, the results of [11] imply the following theorem:

Theorem 11 *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 10, we have

$$\mathbb{P}\left(m_0 \geq \frac{\kappa(\delta)}{\sqrt{d}}, \quad m_t \geq \frac{m_0}{4} + \frac{c\gamma}{2}t \quad \forall t \leq \bar{T} \wedge t_\eta^+ \right) \geq 1 - \delta \tag{18}$$

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,

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

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. ■

Analysis of ϕ To prove Theorem 4, it remains to show the following proposition:

Proposition 12 *If $\ell_p^\star = 1$, then for almost every choice of ρ_0, we have $\phi(0) \neq 0$. On the other hand, if $\ell_p^\star = 2$, 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 13 *Under Assumption 3, the function $\phi(m;\rho_0)$ is analytic in ρ_0. Further, we have*

$$\left.\frac{\partial^k \phi(m;\rho_0)}{\partial \rho_0^k}\right|_{\rho_0=0} = \mathbb{E}\left[h^\star(x^\star)^{(k+1)} x^\star \sigma^{k+1}(x)\sigma'(x)^k\right] \tag{20}$$

where $x, x^\star$ follow the same distribution as in eq. (16).

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

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

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

We define the following quantities related to Assumption 3:

$$u_0^{(k)} = \mathbb{E}\left[\sigma^{(k)}(x)\sigma'(x)^{k-1}\right] \qquad u_1^{(k)} = \mathbb{E}\left[x\sigma^{(k)}(x)\sigma'(x)^{k-1}\right]$$

$$v_0^{(k)} = \mathbb{E}\left[x^\star h^\star(x^\star)^k\right] \qquad v_1^{(k)} = \mathbb{E}\left[[(x^\star)^2-1]h^\star(x^\star)^k\right]$$

Then, by Proposition 11.37 from [34], we have

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

This allows us to show Proposition 12:
Proof Assume first that $\ell_p^\star = 1$. Then $\phi(0;\rho_0)$ is an analytic function of ρ_0 with

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

Since $\ell_p^\star = 1$, 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

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

■

D.2. Proof of Lemma 5

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 5 thus ensues from the following result:

Lemma 14 *Let f be an analytic and non-identically zero function. Then*

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

for almost every b under the Lebesgue measure.

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

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

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

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

■

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 5.

D.3. Proof of Theorem 6

We decompose the polynomial $h^\star$ 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

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

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

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

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

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

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

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

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.

D.4. Hardness of sign functions

We show in the appendix the following statement:

Proposition 15 *Let $m \in \mathbb{N}$, and*

$$h^\star(\mathbf{x}) = \operatorname{sign}(x_1 \dots x_m)$$

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

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 z of degree k, and therefore each monomial term is missing at least one variable. We show that this implies the lower bound of Proposition 15 through the following lemma:

Lemma 16 *Let $f : \{-1,1\} \to \mathbb{R}$ be an arbitrary function, and $g : \mathbb{R}^m \to \mathbb{R}$ a function which is independent from x_m. Then*

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

Proof We simply write

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

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 $h^\star$. ■

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

$$\mathbb{E}\left[f(h^\star(x))H_k(\langle \mathbf{v}, \mathbf{x}\rangle)\right] = \frac{f(1)+f(-1)}{2}\mathbb{E}\left[H_k(\langle \mathbf{v}, \mathbf{x}\rangle)\right] \tag{21}$$

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

For the upper bound, let $\mathbf{v} = \mathbf{e}_1 + \cdots + \mathbf{e_m}$. 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,

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

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

Appendix E. 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.

E.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 4 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.

Altought 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* [39]) 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 4 Right.

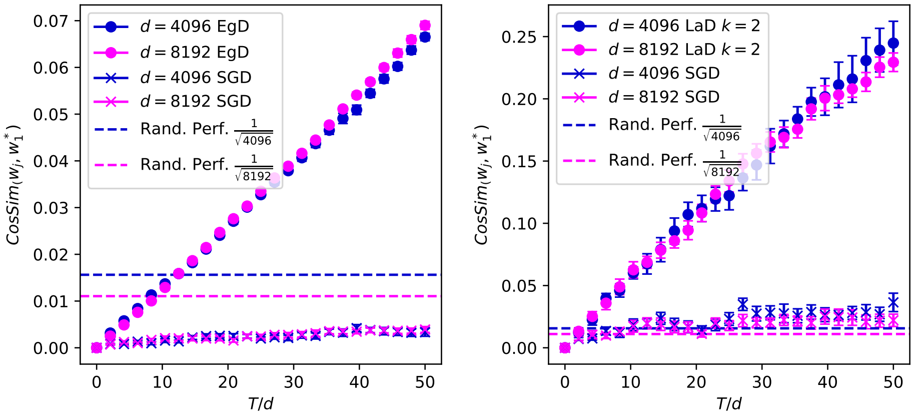

Figure 4: example of two different algorithm with data repetitions learning an hard single-index target $h^\star(\mathbf{z}) = \mathrm{He}_3(z_1)$, $\ell = 3, \ell^\star = 1$. **Left**: SAM, with $\rho_0 = 0.1, \gamma_0 = 0.01$. **Right**: 2-Lookahed with $\gamma_0 = 0.1$. See details in App. F.

E.2. Extensive batch size

In this section, we want to present some evidence that our result does not change when the batch size is $1 < n_b \leq O(d^{\frac{\ell^\star}{2}})$. 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 [5] that the total sample complexity does not change when using batch sizes $n_b \leq O\left(d^{\frac{\ell}{2}}\right)$. 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 $n_b \leq O\left(d^{\frac{\ell^\star}{2}}\right)$)

$$T \cdot n_b \sim \begin{cases} O(d^{\ell-1}) & \text{if } \ell^\star > 2 \\ O(d \log d) & \text{if } \ell^\star = 2 \\ O(d) & \text{if } \ell^\star = 1 \end{cases}. \tag{22}$$

Note that this drastically reduces the time steps needed to learn the teacher's directions, although the sample complexity does not change. [21] 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 $\ell^\star = 1$, 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 [21]; Figure 5 (Left) is a numerical proof of this fact.

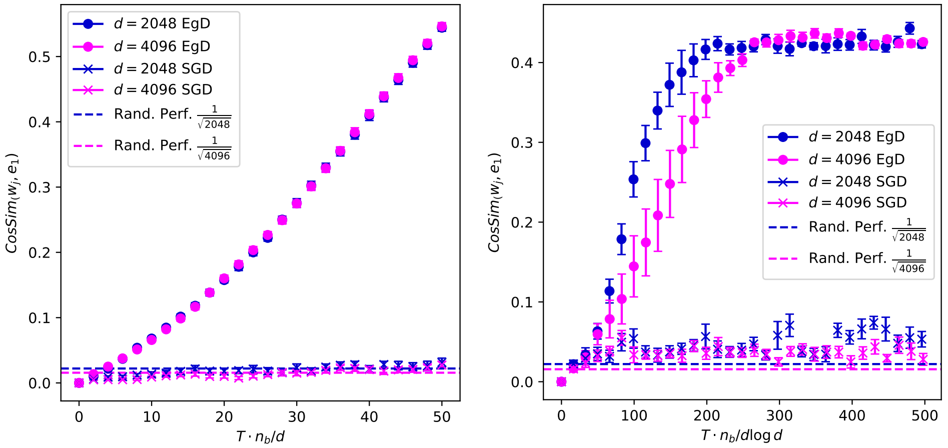

Figure 5: Single-index recovery for large-batch training. The dashed horizontal line $\frac{1}{\sqrt{d}}$ is a visual guide to place random performance. The plots show $\sigma = \text{relu}$, $\gamma = 0.01$, $\rho = 0.1$, and the performance is computed averaging across 10 different runs. **Left** $(\ell, \ell^\star) = (3, 1)$**:** $h^\star = \text{He}_3$. **Right** $(\ell, \ell^\star) = (4, 2)$**:** $h^\star = \text{He}_4$. See details in App. F.

- He_4 has generative exponent $\ell^\star = 2$, 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 5 (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 6 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 $n_b = d$.

E.3. An example of SQ-Staircase function

As we discussed in the main, the introduction generative exponent for multi-index model unveil a new type of stair mechanism, playing the same role as the information exponent in [1]. We refer to the latter as *CSQ-staircase* targets, while we informally call the new class we are presenting *SQ-staircase functions*.

The plot in Figure 3 is showing the staircase mechanism in action for a target that is both *SQ-staircase* and *CSQ-staircase*. Here, we push forward showing the same for the target $h^\star(\mathbf{z}) = \text{He}_4(z_1) + 5\text{sign}(z_1 z_2 z_3)$, in Figure 7. The information exponent of $h^\star$ is $\ell = 3$: the three direction $w_1^\star, w_2^\star, w_3^\star$ are all equally hard and they are learned all together with $O(d^2)$ samples; the present of the addend He_4 does not change the hardness of the target since it has information exponent = 4 if considered alone. On the other end, the generative exponent of $h^\star$ is 2, so the direction $w_1^\star$ can be learned in $O(d \log d)$ steps, and then helping learn $w_2^\star, w_3^\star$ in another $O(d \log d)$ steps. Figure 7 clearly shows that at first only the first direction is learned, and only after that also the other two can be weakly correlated.

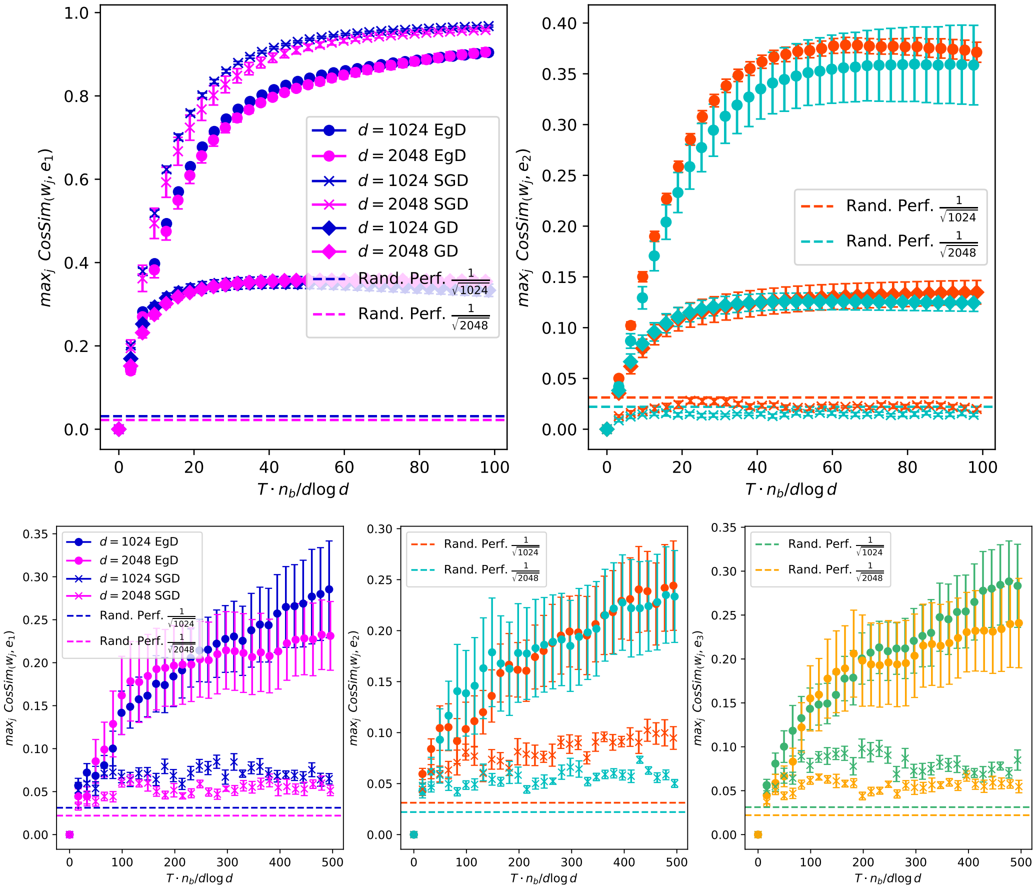

Figure 6: Multi-index recovery for large-batch training. The dashed horizontal line $\frac{1}{\sqrt{d}}$ is a visual guide to place random performance. The plots show $\sigma = \text{relu}$. **Upper** $(\ell_{\mathbf{v}}, \ell^\star_{\mathbf{v}}) = (3, 1)$**:** $h^\star(\mathbf{z}) = \text{sign}(z_1 z_2)$. **Lower** $(\ell_{\mathbf{v}}, \ell^\star_{\mathbf{v}}) = (4, 2)$**:** $h^\star(\mathbf{z}) = z_1 z_2 z_3$. See Appendix F for details.

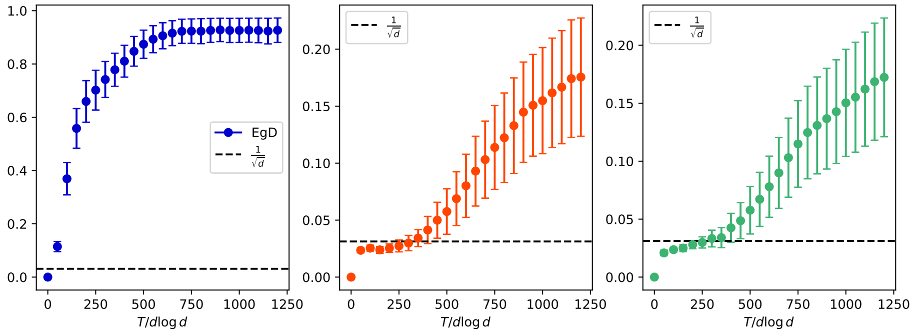

Figure 7: learning dynamic of the three hidden direction of the SQ-staircase function $h^\star(\mathbf{z}) = \mathrm{He}_4(z_1) + 5\mathrm{sign}(z_1 z_2 z_3)$, that is not CSQ-staircase. $\gamma_0 = 0.01$, $\rho_0 = 0.1$, $d = 1024$, $\sigma = \mathrm{ReLU}$. See details in App. F

Appendix F. 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 $W^0 \perp W^\star$, 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 $\mathbf{w}^\star$, averaged across N different runs (where we vary both seen samples and initial conditions). If q in the index for different runs:

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

where

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

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

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

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