Does SGD really happen in tiny subspaces?

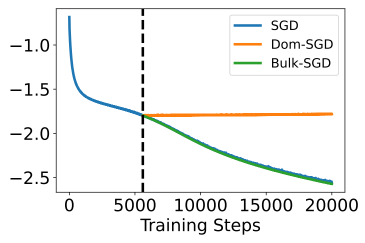
Minhak Song MINHAKSONG@KAIST.AC.KR
KAIST ISysE/Math, Daejeon, South Korea

Kwangjun Ahn KJAHN@MIT.EDU
MIT/Microsoft Research, Cambridge, MA, United States

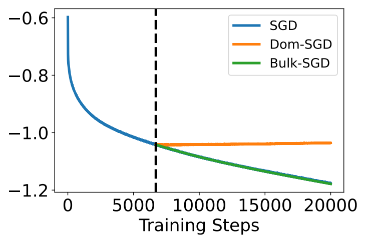
Chulhee Yun CHULHEE.YUN@KAIST.AC.KR
KAIST AI, Seoul, South Korea

Abstract

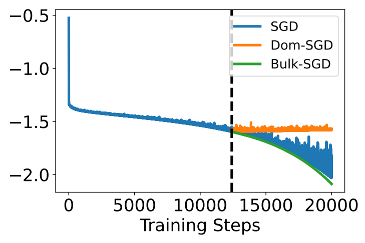
Understanding the training dynamics of deep neural networks is challenging due to their high-dimensional nature and intricate loss landscapes. Recent studies show that gradients approximately align with a low-rank eigenspace of the training loss Hessian, referred to as the dominant subspace. This paper investigates whether neural networks can be trained within this subspace. Our primary finding is that projecting the SGD update onto the dominant subspace does not reduce the training loss, suggesting the alignment between the gradient and dominant subspace is spurious. Surprisingly, excluding the dominant subspace component proves as effective as the original update. Similar observations are made for the large learning rate regime (also known as Edge of Stability) and Sharpness-Aware Minimization. We discuss the main causes and implications of this spurious alignment, shedding light on neural network training dynamics.



(*a*) MLP on MNIST-5k



(*b*) CNN on CIFAR10-5k



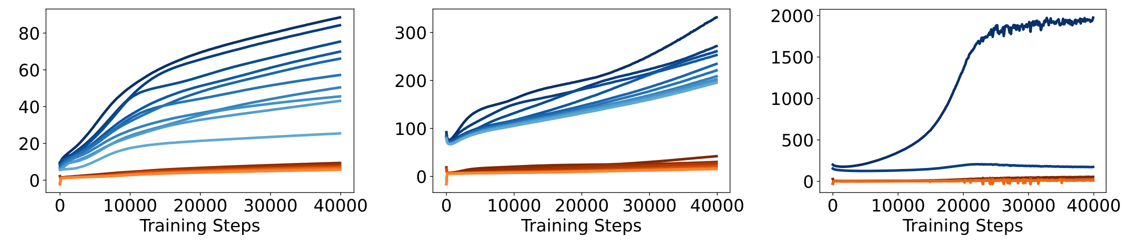
(*c*) Transformer on SST2-1k

Figure 1: The summary of our main results in Section 3 (training loss in log-scale). For neural network training, Gur-Ari et al. [20] observe that gradients approximately align with the dominant subspace, spanned by the dominant eigenvectors of the training loss Hessian. To see whether such phenomenon lets us train neural networks within the dominant subspace, we implement Dom-SGD, where each SGD update is projected onto the dominant subspace. Surprisingly, training stops after this modification, suggesting that **the dominant subspace is not where the learning happens**. In contrast, Bulk-SGD, where we project each SGD updates onto the bulk subspace orthogonal to the dominant subspace, is just as effective as the original update, despite removing the majority of original updates. Experimental details are provided in Section E.

1. Introduction

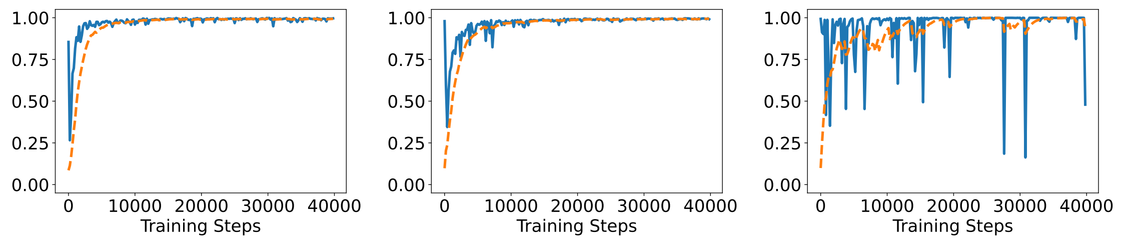
Understanding the optimization of deep neural networks presents a complex challenge, given their high-dimensional nature and the intricate characteristics of their training loss landscapes. Over the last

© M. Song, K. Ahn & C. Yun.



(a) MLP on MNIST-5k (b) CNN on CIFAR10-5k (c) Transformer on SST2-1k

Figure 2: **Low-rank structure of the Hessian.** The plot shows the top eigenvalues of the loss Hessian during SGD training. The blue curves represent the top-k eigenvalues, which are significantly larger than the rest of eigenvalues, shown in orange, where k is the number of classes for the classification task.



(a) MLP on MNIST-5k (b) CNN on CIFAR10-5k (c) Transformer on SST2-1k

Figure 3: **Alignment of gradients with dominant subspaces.** The plot illustrates $\chi_k(\nabla L(\theta_t))$ during SGD training (see Theorem 2). The orange dashed lines represent the exponential moving average (EMA) of $\chi_k(\nabla L(\theta_t))$. After a few early steps, $\chi_k(\nabla L(\theta_t))$ reaches and stays near 1, indicating the alignment between gradients and dominant subspaces.

decade, an abundance of studies has investigated the landscape of training loss $L : \mathbb{R}^p \to \mathbb{R}$ [21, 36]. In this work, we are interested in the following noteworthy phenomena:

- **Hessian is approximately low-rank.** Extensive research [18, 40, 41, 44, 45] has revealed that for k-class classification problems, the loss Hessian $\nabla^2 L$ exhibits a low-rank structure, characterized by k dominant eigenvalues significantly larger than the others. See Figure 2 for details.
- **Gradients approximately align with the low-rank eigenspace.** Gur-Ari et al. [20] demonstrated that during SGD training, gradients tend to align closely with the low-dimensional subspace spanned by the dominant eigenvectors of the loss Hessian. See Figure 3 for details.

Due to space limit, we provide a more extensive background on these phenomena in Section D. The eigenspace of the top-k eigenvalues of $\nabla^2 L(\theta)$, referred to as the **dominant subspace** at θ, is the main focus of this work. Motivated by Gur-Ari et al. [20], we ask:

Q. *Can deep neural networks be trained within the dominant subspace?* (Q1)

This question carries significant practical implications, potentially leading to more efficient training methods for neural networks. Furthermore, it offers insights into why deep learning optimization may not suffer from the curse of dimensionality despite operating in high-dimensional spaces.

2. Starting point: gradient aligns with the dominant subspace

In this section, we set the stage for our main results by reviewing the main observation of Gur-Ari et al. [20]. To that end, we first introduce some notations to ease our discussion.

Notations. Let $[n]$ denote the set $\{1, 2, \ldots, n\}$. For the Hessian to be well-defined, let $L : \mathbb{R}^p \to \mathbb{R}$ be a twice-differentiable training loss. For $\theta \in \mathbb{R}^p$, let $\lambda_1(\theta), \lambda_2(\theta), \ldots, \lambda_p(\theta)$ denote the eigenvalues of the loss Hessian $\nabla^2 L(\theta) \in \mathbb{R}^{p\times p}$ in descending order, and let $u_1(\theta), u_2(\theta), \ldots, u_p(\theta)$ denote the corresponding eigenvectors. Given these notations, we begin with the most important concept for our discussion, namely, the *dominant subspace*.

Definition 1 (Dominant subspace) *The top-k* ***dominant subspace*** *at θ is denoted by*

$$\mathcal{S}_k(\theta) := \operatorname{span}\{u_i(\theta) : i \in [k]\}\,,$$

and its orthogonal complement by $\mathcal{S}_k^\perp(\theta)$, referred to as the ***bulk subspace****. Unless specified otherwise, the default choice for k is the number of classes for the classification task.*

Definition 2 (Dominant subspace projection) *The projection matrix onto the dominant subspace $\mathcal{S}_k(\theta)$ is denoted by*

$$P_k(\theta) := \sum_{i=1}^{k} u_i(\theta)u_i(\theta)^\top \in \mathbb{R}^{p\times p}\,,$$

and the projection matrix onto $\mathcal{S}_k^\perp(\theta)$ by $P_k^\perp(\theta) := I - P_k(\theta)$. For a given vector $v \in \mathbb{R}^p$, we can decompose the vector into $v = P_k(\theta)v + P_k^\perp(\theta)v$. We say $P_k(\theta)v$ is the ***dominant component*** *of v, and $P_k^\perp(\theta)v$ is the* ***bulk component*** *of v. We denote the fraction of v in the dominant subspace by*

$$\chi_k(v;\theta) := \|P_k(\theta)v\|_2/\|v\|_2\,,$$

with $\chi_k(v;\theta) = 0$ if $\|v\|_2 = 0$. A vector $v \in \mathbb{R}^p$ is said to (approximately) align with the dominant subspace $\mathcal{S}_k(\theta)$ if $\chi_k(v;\theta)$ is close to 1, and align with $\mathcal{S}_k^\perp(\theta)$ if $\chi_k(v;\theta)$ is close to 0.

When clear from context, we use shorthand notation such as $\lambda_i := \lambda_i(\theta)$, $u_i := u_i(\theta)$, $\nabla L := \nabla L(\theta)$, $\mathcal{S}_k := \mathcal{S}_k(\theta)$, $\mathcal{S}_k^\perp := \mathcal{S}_k^\perp(\theta)$, $P_k := P_k(\theta)$, $P_k^\perp := P_k^\perp(\theta)$, and $\chi_k(v) := \chi_k(v;\theta)$.

Using our notations, the striking observation of Gur-Ari et al. [20] can be formalized as follows.

Phenomenon 1 ***Gradient approximately aligns with the dominant subspace along SGD trajectories.*** *Consider the SGD trajectory $\{\theta_t\}$ with a constant learning rate. After a few initial steps, $\chi_k(\nabla L(\theta_t))$ quickly reaches and remains near 1.*

In Figure 3, we confirm the main results of Gur-Ari et al. [20] for various settings. Notice that $\chi_k(\nabla L(\theta_t))$ reaches 1 after a few early steps, indicating that the gradient $\nabla L(\theta_t)$ approximately aligns with the dominant subspace $\mathcal{S}_k(\theta_t)$. Given this alignment, it seems that the training can be done within the dominant subspace, which leads to the previously introduced question (Q1). In the next section, we conduct a set of experiments to investigate (Q1).

3. Neural networks cannot be trained within the dominant subspace

In this section, we present the first main observation of this paper regarding question (Q1). We start with a preliminary analysis using the local quadratic approximation of the neural network landscape.

3.1. What do we expect based on quadratic Taylor approximation?

To analyze the convergence of gradient-based optimization algorithms, a common approach is to use the local quadratic Taylor approximation (see, e.g., [17]). The "descent lemma" characterizes the one-step progress of the optimizer $L(\theta_{t+1}) - L(\theta_t)$ using this approximation, assuming the training loss L is smooth. Based on the local quadratic Taylor approximation, we have:

$$L(\theta_{t+1}) - L(\theta_t) \approx \underbrace{\langle \nabla L(\theta_t), \theta_{t+1} - \theta_t \rangle}_{=:\text{gradient correlation}} + \frac{1}{2}(\theta_{t+1} - \theta_t)^\top \nabla^2 L(\theta_t)(\theta_{t+1} - \theta_t)\,. \tag{1}$$

Let us denote the first term on the RHS by the *gradient correlation* term and the second term by the *second-order error* term. Since the SGD updates are defined as

$$\theta_{t+1} \leftarrow \theta_t - \eta g_t$$

for the stochastic gradient g_t at θ_t, the gradient correlation term is negative in expectation:

$$\mathbb{E}[\text{gradient correlation}] = \mathbb{E}[\langle \nabla L(\theta_t), -\eta g_t \rangle] = -\eta \left\| \nabla L(\theta_t) \right\|^2 < 0\,. \tag{2}$$

For the experiments in Figure 1, we use small learning rates to ensure SGD iterates decrease the training loss nearly monotonically, suggesting that the negative gradient correlation dominates the second-order error term in these cases.

Hence, if the quadratic Taylor approximation was accurate enough, based on the above analysis and Phenomenon 1, it is expected that one can decrease the training loss based on updates lying in the dominant subspace, as hypothesized in the question (Q1).

To directly test this hypothesis, we design the following critical experiment.

Our critical experiment: In the same settings as before, whenever Phenomenon 1 occurs, consider the following updates where each update of SGD is projected onto the dominant subspace:

$$\theta_{t+1} \leftarrow \theta_t - \eta P_k(\theta_t) g_t\,. \tag{Dom-SGD}$$

By Phenomenon 1, Dom-SGD has an approximately same gradient correlation as SGD as (2):

$$\mathbb{E}[\text{gradient correlation}] = \mathbb{E}[\langle \nabla L(\theta_t), -\eta P_k(\theta_t) g_t \rangle] \approx -\eta \left\| \nabla L(\theta_t) \right\|^2\,.$$

Therefore, based on the local quadratic Taylor approximation, Dom-SGD should be able to successfully train neural networks whenever Phenomenon 1 occurs. Is it really the case?

3.2. The "spurious" alignment with the dominant subspace

In the same settings as before, we first train neural networks with SGD up until we observe Phenomenon 1. Specifically, we track the exponential moving average (EMA) of $\chi_k(\nabla L(\theta_t))$ values (EMA factor set to 0.9), and switch from SGD to Dom-SGD whenever the EMA value exceeds 0.95. Note that we recompute the dominant subspace at every step when running Dom-SGD.

For various settings, we plot the training loss of Dom-SGD in Figure 1, comparing it with SGD under the same initialization. We employ a constant learning rate and mean squared error (MSE) loss for classification [8, 22]. Additional experiments, including those using cross-entropy loss and training standard architectures, are provided in Section F.

Consistently, we observe that **Dom-SGD fails to further decrease the training loss**, in contrast to SGD. This suggests that the *dominant component* of stochastic gradient g_t is in fact not beneficial for training, despite constituting the majority of g_t.

3.3. Bulk subspace is where the learning happens

To further strengthen our main observation, we conduct another set of experiments, wherein we switch from SGD to the following update scheme:

$$\theta_{t+1} \leftarrow \theta_t - \eta P_k^{\perp}(\theta_t) g_t \,. \tag{Bulk-SGD}$$

Essentially, Bulk-SGD discards the majority of the stochastic gradient g_t by removing its dominant component. Consequently, it seems less likely that the remaining bulk component of stochastic gradient would lead to successful training.

Surprisingly, as shown in Figure 1, **Bulk-SGD is as effective as SGD in decreasing the training loss**. This further highlights that it is indeed a small fraction of gradient that aligns with the bulk subspace that contributes to training loss decrease.

One can summarize our observation thus far as follows.

Phenomenon 2 *Although the gradient approximately aligns with the dominant subspace at each step, **the training loss does not decrease within the dominant subspace**, suggesting a "spurious" alignment. Surprisingly, with Bulk-SGD, where each update is projected onto bulk subspaces, the training remains as effective as the original update. This emphasizes the importance of a small component of the update that aligns with the bulk subspace.*

4. Summary of additional results

Due to space limit, we provide a summary of our additional results as below.

- In Section A, we identify that the spurious alignment between the gradient and the dominant subspace is caused by the stochastic noise inherent to SGD, by showing that alignment disappears when using full-batch GD (see Figure 4 and Figure 5). Additionally, we present a simple quadratic model which also captures the observed phenomena, providing insights into the role of stochastic noise in the spurious alignment (see Figure 6 and Figure 7).
- In Section B, we extend our observations to two other practical settings: (1) GD with large learning rates, especially in the Edge of Stability reigme [8], and (2) Sharpness-Aware Minimization (SAM) [14]. Our results hold for these two settings: each update approximately aligns with the dominant subspace, yet the aligned component of each update does not contribute to the loss decrement (see Figure 9 and Figure 10).
- In Section C, we demonstrate that momentum and adaptive optimizers, *e.g.* Adam, amplify the bulk subspace component of each update, partially explaining their success in neural network training (see Table 1 and Figure 11).

5. Conclusion

Motivated by the observation of Gur-Ari et al. [20] that the gradient aligns with a low-dimensional dominant eigenspace of the training loss Hessian, this work investigates the possiblity of training neural networks in the dominant subspace. Through several critical experiments, our main results show that neural networks *cannot* be trained in the dominant subspace. Although each update mostly aligns with the dominant subspace, we show that a small component of each update that aligns with the bulk subspace is crucial for neural network training.

References

[1] Kwangjun Ahn, Jingzhao Zhang, and Suvrit Sra. Understanding the unstable convergence of gradient descent. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 247–257. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/ahn22a.html.

[2] Kwangjun Ahn, Sebastien Bubeck, Sinho Chewi, Yin Tat Lee, Felipe Suarez, and Yi Zhang. Learning threshold neurons via edge of stability. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=9cQ6kToLnJ.

[3] Kwangjun Ahn, Xiang Cheng, Minhak Song, Chulhee Yun, Ali Jadbabaie, and Suvrit Sra. Linear attention is (maybe) all you need (to understand transformer optimization). In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=0uI5415ry7.

[4] Kwangjun Ahn, Zhiyu Zhang, Yunbum Kook, and Yan Dai. Understanding Adam optimizer via online learning of updates: Adam is FTRL in disguise. In *International Conference on Machine Learning*. PMLR, 2024.

[5] Maksym Andriushchenko and Nicolas Flammarion. Towards understanding sharpness-aware minimization. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 639–668. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/andriushchenko22a.html.

[6] Gerard Ben Arous, Reza Gheissari, Jiaoyang Huang, and Aukosh Jagannath. High-dimensional SGD aligns with emerging outlier eigenspaces. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=MHjigVnI04.

[7] Peter L. Bartlett, Philip M. Long, and Olivier Bousquet. The dynamics of sharpness-aware minimization: Bouncing across ravines and drifting towards wide minima. *Journal of Machine Learning Research*, 24(316):1–36, 2023. URL http://jmlr.org/papers/v24/23-043.html.

[8] Jeremy Cohen, Simran Kaur, Yuanzhi Li, J Zico Kolter, and Ameet Talwalkar. Gradient descent on neural networks typically occurs at the edge of stability. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=jh-rTtvkGeM.

[9] Romain Cosson, Ali Jadbabaie, Anuran Makur, Amirhossein Reisizadeh, and Devavrat Shah. Gradient descent with low-rank objective functions. In *2023 62nd IEEE Conference on Decision and Control (CDC)*, pages 3309–3314, 2023. doi: 10.1109/CDC49753.2023.10383652.

[10] Michael Crawshaw, Mingrui Liu, Francesco Orabona, Wei Zhang, and Zhenxun Zhuang. Robustness to unbounded smoothness of generalized signsgd. *Advances in neural information processing systems*, 35:9955–9968, 2022.

[11] Ashok Cutkosky and Harsh Mehta. Momentum improves normalized sgd. In *International conference on machine learning*, pages 2260–2268. PMLR, 2020.

[12] Yan Dai, Kwangjun Ahn, and Suvrit Sra. The crucial role of normalization in sharpness-aware minimization. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=zq4vFneRiA.

[13] Alex Damian, Eshaan Nichani, and Jason D. Lee. Self-stabilization: The implicit bias of gradient descent at the edge of stability. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=nhKHA59gXz.

[14] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=6Tm1mposlrM.

[15] Jingwen Fu, Bohan Wang, Huishuai Zhang, Zhizheng Zhang, Wei Chen, and Nanning Zheng. When and why momentum accelerates sgd: An empirical study. *arXiv preprint arXiv:2306.09000*, 2023.

[16] Martin Gauch, Maximilian Beck, Thomas Adler, Dmytro Kotsur, Stefan Fiel, Hamid Eghbal-zadeh, Johannes Brandstetter, Johannes Kofler, Markus Holzleitner, Werner Zellinger, Daniel Klotz, Sepp Hochreiter, and Sebastian Lehner. Few-shot learning by dimensionality reduction in gradient space. In Sarath Chandar, Razvan Pascanu, and Doina Precup, editors, *Proceedings of The 1st Conference on Lifelong Learning Agents*, volume 199 of *Proceedings of Machine Learning Research*, pages 1043–1064. PMLR, 22–24 Aug 2022. URL https://proceedings.mlr.press/v199/gauch22a.html.

[17] Saeed Ghadimi and Guanghui Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM journal on optimization*, 23(4):2341–2368, 2013.

[18] Behrooz Ghorbani, Shankar Krishnan, and Ying Xiao. An investigation into neural net optimization via hessian eigenvalue density. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2232–2241. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/ghorbani19b.html.

[19] Frithjof Gressmann, Zach Eaton-Rosen, and Carlo Luschi. Improving neural network training in low dimensional random bases. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 12140–12150. Curran Associates, Inc., 2020.

[20] Guy Gur-Ari, Daniel A Roberts, and Ethan Dyer. Gradient descent happens in a tiny subspace. *arXiv preprint arXiv:1812.04754*, 2018.

[21] Haowei He, Gao Huang, and Yang Yuan. Asymmetric valleys: Beyond sharp and flat local minima. 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.

[22] Like Hui and Mikhail Belkin. Evaluation of neural architectures trained with square loss vs cross-entropy in classification tasks. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=hsFN92eQEla.

[23] Ali Jadbabaie, Anuran Makur, and Amirhossein Reisizadeh. Adaptive low-rank gradient descent. In *2023 62nd IEEE Conference on Decision and Control (CDC)*, pages 3315–3320, 2023. doi: 10.1109/CDC49753.2023.10383982.

[24] Stanisław Jastrzębski, Zachary Kenton, Nicolas Ballas, Asja Fischer, Yoshua Bengio, and Amost Storkey. On the relation between the sharpest directions of DNN loss and the SGD step length. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=SkgEaj05t7.

[25] Stanisław Jastrzębski, Maciej Szymczak, Stanislav Fort, Devansh Arpit, Jacek Tabor, Kyunghyun Cho*, and Krzysztof Geras*. The break-even point on optimization trajectories of deep neural networks. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=r1g87C4KwB.

[26] Yiding Jiang, Behnam Neyshabur, Hossein Mobahi, Dilip Krishnan, and Samy Bengio. Fantastic generalization measures and where to find them. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SJgIPJBFvH.

[27] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=H1oyRlYgg.

[28] Rahul Kidambi, Praneeth Netrapalli, Prateek Jain, and Sham Kakade. On the insufficiency of existing momentum schemes for stochastic optimization. In *2018 Information Theory and Applications Workshop (ITA)*, pages 1–9, 2018. doi: 10.1109/ITA.2018.8503173.

[29] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[30] Itai Kreisler, Mor Shpigel Nacson, Daniel Soudry, and Yair Carmon. Gradient descent monotonically decreases the sharpness of gradient flow solutions in scalar networks and beyond. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 17684–17744. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/kreisler23a.html.

[31] Alex Krizhevsky. Learning multiple layers of features from tiny images, 2009.

[32] Frederik Kunstner, Jacques Chen, Jonathan Wilder Lavington, and Mark Schmidt. Noise is not the main factor behind the gap between sgd and adam on transformers, but sign descent might be. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=a65YK0cqH8g.

[33] Frederik Kunstner, Robin Yadav, Alan Milligan, Mark Schmidt, and Alberto Bietti. Heavy-tailed class imbalance and why adam outperforms gradient descent on language models. *arXiv preprint arXiv:2402.19449*, 2024.

[34] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[35] Chunyuan Li, Heerad Farkhoor, Rosanne Liu, and Jason Yosinski. Measuring the intrinsic dimension of objective landscapes. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=ryup8-WCW.

[36] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[37] Tao Li, Lei Tan, Qinghua Tao, Yipeng Liu, and Xiaolin Huang. Low dimensional trajectory hypothesis is true: Dnns can be trained in tiny subspaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(03):3411–3420, mar 2023. ISSN 1939-3539. doi: 10.1109/TPAMI.2022.3178101.

[38] Philip M Long and Peter L Bartlett. Sharpness-aware minimization and the edge of stability. *arXiv preprint arXiv:2309.12488*, 2023.

[39] Yurii Nesterov et al. *Lectures on convex optimization*, volume 137. Springer, 2018.

[40] Vardan Papyan. Measurements of three-level hierarchical structure in the outliers in the spectrum of deepnet hessians. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5012–5021. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/papyan19a.html.

[41] Vardan Papyan. Traces of class/cross-class structure pervade deep learning spectra. *Journal of Machine Learning Research*, 21(252):1–64, 2020. URL http://jmlr.org/papers/v21/20-933.html.

[42] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. 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/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf.

[43] B.T. Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964. ISSN 0041-5553. doi: https://doi.org/10.1016/0041-5553(64)90137-5. URL https://www.sciencedirect.com/science/article/pii/0041555364901375.

[44] Levent Sagun, Leon Bottou, and Yann LeCun. Eigenvalues of the hessian in deep learning: Singularity and beyond. *arXiv preprint arXiv:1611.07476*, 2016.

[45] Levent Sagun, Utku Evci, V Ugur Guney, Yann Dauphin, and Leon Bottou. Empirical analysis of the hessian of over-parametrized neural networks. *arXiv preprint arXiv:1706.04454*, 2017.

[46] Jan Schneider, Pierre Schumacher, Simon Guist, Le Chen, Daniel Haeufle, Bernhard Schölkopf, and Dieter Büchler. Identifying policy gradient subspaces. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=iPWxqnt2ke.

[47] Dongkuk Si and Chulhee Yun. Practical sharpness-aware minimization cannot converge all the way to optima. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=nijJN0LHqM.

[48] Vikrant Singhal and Thomas Steinke. Privately learning subspaces. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=YBanVDVEbVe.

[49] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

[50] Minhak Song and Chulhee Yun. Trajectory alignment: Understanding the edge of stability phenomenon via bifurcation theory. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=PnJaA0A8Lr.

[51] Runzhe Wang, Sadhika Malladi, Tianhao Wang, Kaifeng Lyu, and Zhiyuan Li. The marginal value of momentum for small learning rate SGD. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=3JjJezzVkT.

[52] Kaiyue Wen, Tengyu Ma, and Zhiyuan Li. How sharpness-aware minimization minimizes sharpness? In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=5spDgWmpY6x.

[53] Jingfeng Wu, Vladimir Braverman, and Jason D. Lee. Implicit bias of gradient descent for logistic regression at the edge of stability. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=IT9mWLYNpQ.

[54] Shuo Xie and Zhiyuan Li. Implicit bias of adamw: ℓ_∞ norm constrained optimization. In *International Conference on Machine Learning*. PMLR, 2024.

[55] Jingzhao Zhang, Sai Praneeth Karimireddy, Andreas Veit, Seungyeon Kim, Sashank Reddi, Sanjiv Kumar, and Suvrit Sra. Why are adaptive methods good for attention models? In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 15383–15393. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/b05b57f6add810d3b7490866d74c0053-Paper.pdf.

[56] Yushun Zhang, Congliang Chen, Tian Ding, Ziniu Li, Ruoyu Sun, and Zhi-Quan Luo. Why transformers need adam: A hessian perspective. *arXiv preprint arXiv:2402.16788*, 2024.

[57] Yingxue Zhou, Steven Wu, and Arindam Banerjee. Bypassing the ambient dimension: Private {sgd} with gradient subspace identification. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=7dpmlkBuJFC.

[58] Xingyu Zhu, Zixuan Wang, Xiang Wang, Mo Zhou, and Rong Ge. Understanding edge-of-stability training dynamics with a minimalist example. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=p7EagBsMAEO.

Appendix

Appendix A. What causes the spurious alignment with the dominant subspaces?

In this section, we aim to explain the spurious alignment discussed in Phenomenon 2. This phenomenon is closely tied to the landscape of the training loss near SGD trajectories. Importantly, SGD introduces inherent *randomness* into its trajectories. We investigate how this *stochastic noise* affects the phenomenon.

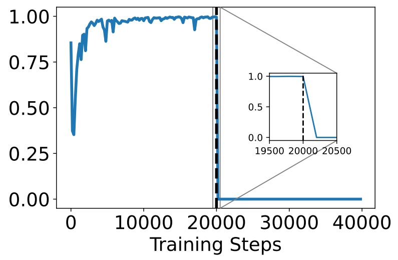


Figure 4: $\chi_k(\nabla L(\theta_t))$ when switching from SGD to GD at step 20000 while training MLP on MNIST-5k.

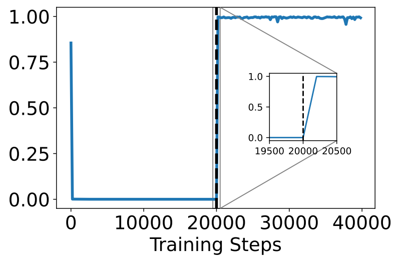


Figure 5: $\chi_k(\nabla L(\theta_t))$ when switching from GD to SGD.

A.1. Stochastic noise of SGD is the main cause

We examine the role of stochastic noise by contrasting the behavior of SGD with (full-batch) GD, isolating the effect of stochastic noise.

First, we demonstrate the crucial role of stochastic noise in the alignment by switching from SGD to GD when Phenomenon 1 is observed. Strikingly, as shown in Figure 4, the **alignment disappears as soon as we switch to GD**. More specifically, $\chi_k(\nabla L(\theta_t))$ quickly becomes 0 as soon as the switch occurs. This sharp transition indicates that the stochastic noise must play a crucial role. See Section G.3 for more results.

In Section G.1, when training neural networks with GD using small learning rates, we observe **no alignment between gradients and dominant subspaces**, in contrast to SGD. In this case, $\chi_k(\nabla L(\theta_t))$ quickly reaches and remains near 0, indicating alignment of gradients with bulk subspaces. Remarkably, despite despite differences in gradient alignment (GD: $\chi_k(\nabla L) \approx 0$, SGD: $\chi_k(\nabla L) \approx 1$), GD and SGD trajectories are close to each other (see Section G.2). This suggests that the presence of small stochastic noise results in a drastically different behavior in the alignment.

Remark 3 *In this section, we consider the small learning rates, where the training loss monotonically decreases along the GD trajectory. The case of larger learning rates (especially in the Edge of Stability regime [8]) will be discussed in Section B.1.*

In Figure 5, we switch our optimizer from GD to SGD, complementary to the experiment in Figure 4. This time we observe that the alignment sharply appears, *i.e.*, $\chi_k(\nabla L(\theta_t))$ quickly becomes 1, as soon as the switch occurs. To sum up, one can summarize the findings of this subsection as follows.

Phenomenon 3 (The spurious alignment is due to stochastic noise) *The alignment between the gradient and the dominant subspace quickly disappears when switching the optimizer from SGD to GD. Moreover, the alignment quickly reappears when switching GD back to SGD. Hence, the spurious alignment is mainly due to the stochastic noise of SGD.*

Given this observation, one might question how the presence of small stochastic noise leads to a drastically different alignment. We investigate this next using a simple model.

A.2. Understanding the role of stochastic noise via a toy quadratic model

Towards understanding Phenomena 1–3, this section introduces a simple example that recovers all the phenomena.

Given the typical ill-conditioned nature of neural network training, we consider a 2-dimensional ill-conditioned quadratic loss, $L(x, y) = \frac{1}{2}(1000x^2 + y^2)$, where $\theta = (x, y) \in \mathbb{R}^2$. We define $\ell_1(x, y) = L(x, y) + 100xy$ and $\ell_2(x, y) = L(x, y) - 100xy$, resulting in $L(x, y) = \frac{1}{2}(\ell_1(x, y) + \ell_2(x, y))$.

We conduct GD with learning rate η as

$$\theta_{t+1}^{\mathrm{GD}} \leftarrow \theta_t^{\mathrm{GD}} - \eta \nabla L(\theta_t^{\mathrm{GD}}) \,,$$

and SGD using random sampling with the same learning rate η as

$$\theta_{t+1}^{\mathrm{SGD}} \leftarrow \theta_t^{\mathrm{SGD}} - \eta \nabla \ell_k(\theta_t^{\mathrm{SGD}}) \,, \quad \text{where } k \sim \mathrm{Unif}(\{1, 2\}) \,.$$

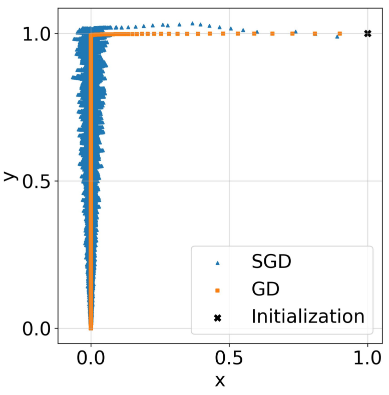


Figure 6: GD and SGD trajectories when training a 2-dimensional ill-conditioned toy quadratic model.

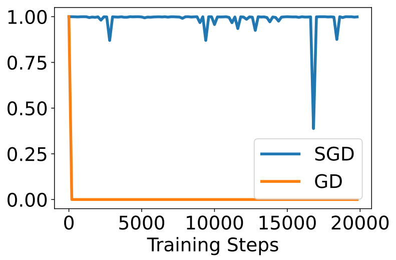


Figure 7: $\chi_1(\nabla L(\theta_t))$ during GD and SGD for Figure 6.

In Figure 6, we visualize the optimization trajectories of GD and SGD with an initialization $\theta_0^{\mathrm{GD}} = \theta_0^{\mathrm{SGD}} = (1, 1)$ and a learning rate $\eta = 10^{-4}$. The Hessian of the quadratic loss remains constant during training, with eigenvalues $\lambda_1 = 1000$ and $\lambda_2 = 1$, and corresponding eigenvectors $e_1 = (1, 0)$ and $e_2 = (0, 1)$. We compute the fraction of gradient in the dominant subspace as $\chi_1(\nabla L(\theta)) := \frac{|\langle \nabla L(\theta), e_1 \rangle|}{\|\nabla L(\theta)\|_2}$, as shown in Figure 7.

Notably, **this simple quadratic model recovers all the observed phenomena** (Phenomena 1–3). In both GD and SGD trajectories, x_t quickly converges to 0 due to the sharper direction along e_1 ($\lambda_1 \gg \lambda_2$). Subsequently, both trajectories remain close to the y-axis throughout the remaining of the training. However, in GD, $\chi_1(\nabla L(\theta_t^{\mathrm{GD}}))$ quickly approaches and remains near 0 (Phenomenon 3), while in SGD, $\chi_1(\nabla L(\theta_t^{\mathrm{SGD}}))$ stays close to 1 (Phenomenon 1). Notice that if we run Dom-SGD, the updates will be done only in the x direction, hence training stops after switching to Dom-SGD (Phenomenon 2).

The discrepancy in alignment between GD and SGD arises from the ill-conditioned nature of the loss landscape, where the small stochastic noise of SGD in the x direction induces a large gradient component along the x direction, resulting in a larger $\chi_1(\nabla L)$ value.

A.3. Revisiting our preliminary analysis (Section 3.1)

At this point, some readers might wonder how we reconcile the results with our preliminary analysis (Section 3.1). Based on our investigations so far, we propose one plausible explanation that the training loss landscape is locally "ill-conditioned-valley"-like. This landscape has two key features causing the spurious alignment:

- The landscape is locally valley-shaped, where it is steep along the dominant subspace and flat along the bulk subspace. In particular, the curvature along the dominant subspace is much larger than that along the bulk subspace.
- The bottom of the valley is connected along the bulk subspace. Moreover, there is a direction within the bulk subspace along which the bottom of the valley descends.

To aid readers' understanding, we provide a simple illustration of an ill-conditioned-valley-like landscape in Figure 8. With these features, Phenomena 1–3 can occur without contradicting our preliminary analysis from Section 3.1:

- Due to stochastic noise, the SGD iterates slightly deviate from the bottom of the valley. Subsequently, the high curvature along the dominant subspace causes gradients to align with this subspace, *i.e.*, the iterates exhibit Phenomenon 1.
- However, Dom-SGD fails to further decrease the training loss, as observed in Phenomenon 2, since it fails to follow the true progress direction along the bulk subspace.
- Moreover, without stochastic noise, the iterates quickly approach the bottom of the valley, where the alignment disappears, as described in Phenomenon 3.

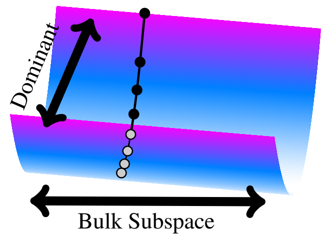


Figure 8: Illustration demonstrating how spurious alignment can occur with an ill-conditioned-valley-like training loss. Dom-SGD iterates (depicted with dots) fail to progress along the bulk subspace where the training loss decreases.

In Section G.4, we measure the distance of the weights from the step where we switch from SGD to Dom-SGD and Bulk-SGD for the experiments in Figure 1. We observe that weights do not move far from the switching step for Dom-SGD, in contrast to SGD and Bulk-SGD, which is consistent with the proposed explanation based on an ill-conditioned-valley-like landscape.

Given our results for SGD thus far, one natural question is whether these phenomena are also observed for other practical optimization algorithms.

Appendix B. Edge of Stability and Sharpness-Aware Minimization

In this section, we extend our investigations to two other practical settings: (1) GD with large learning rates in the Edge of Stability (EoS) regime [8], and (2) Sharpness-Aware Minimization (SAM) [14]. We show that the same phenomena are observed for the two settings: the update direction aligns with the dominant subspace, but the alignment is again spurious.

B.1. Edge of Stability

Recent empirical studies [8, 25] have observed that when training neural networks using full-batch GD with large learning rates η, the sharpness λ_1 increases until it reaches the stability threshold of $2/\eta$, and saturates at the threshold (see Figure B.1). Cohen et al. [8] call this phenomenon as the *Edge of Stability* (EoS).

In Figure B.1, we observe that gradients closely align with dominant subspaces in the EoS regime. This phenomenon stands in contrast with GD with small learning rates (Phenomenon 3), where $\chi_k(\nabla L(\theta_t))$ remains near 0 (see Section G.1 for details).

Given that the gradient approximately aligns with the dominant subspace, we run experiments analogous to Section 3. Specifically, we train neural networks using GD with a large learning rate η until it reaches the EoS regime. We then switch GD to the following update schemes:

$$\theta_{t+1} \leftarrow \theta_t - \eta P_k(\theta_t)\nabla L(\theta_t)\,, \tag{Dom-GD}$$

$$\theta_{t+1} \leftarrow \theta_t - \eta P_k^{\perp}(\theta_t)\nabla L(\theta_t)\,. \tag{Bulk-GD}$$

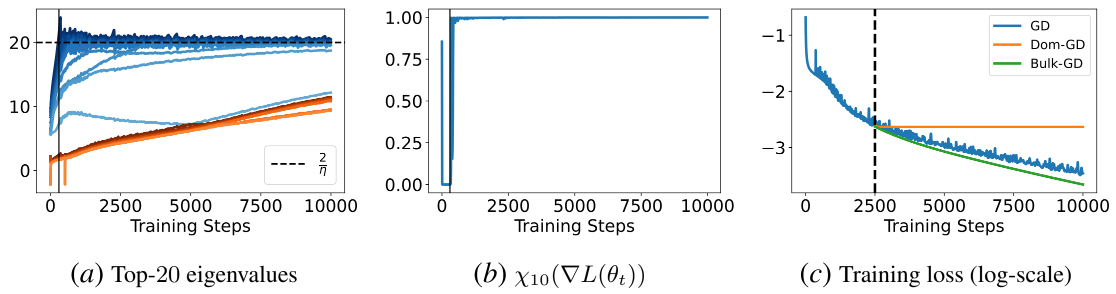


Figure 9: **Gradients approximately align with dominant subspaces in the EoS regime.** Training MLP on MNIST-5k with GD using a large learning rate $\eta = 0.1$. (a) The plot shows the top-10 eigenvalues in blue and the next top-10 eigenvalues in orange. After a few steps, GD enters the EoS regime, where the sharpness stabilizes near $2/\eta$. (b) As the sharpness reaches $2/\eta$, $\chi_{10}(\nabla L(\theta_t))$ approaches and remains near 1. (c) We switch the optimizer from GD to Dom-GD and Bulk-GD at step 2500. **Dom-GD fails to further decrease the training loss, in contrast to GD and Bulk-GD.**

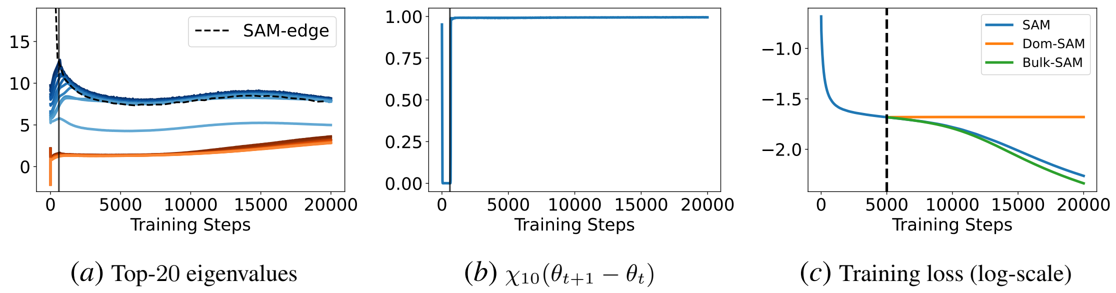


Figure 10: **SAM updates approximately lie in dominant subspaces.** Training MLP on MNIST-5k with SAM using a learning rate $\eta = 0.01$ and a perturbation radius $\rho = 0.1$. (a) The plot shows the top-10 eigenvalues in blue and the next top-10 eigenvalues in orange. After a few steps, SAM operates in the EoS regime, where the sharpness stabilizes near the SAM-edge. (b) As the sharpness reaches the SAM-edge, $\chi_{10}(\theta_{t+1} - \theta_t)$ approaches and remains near 1. (c) We switch the optimizer from SAM to Dom-SAM and Bulk-SAM at step 5000. **Dom-SAM fails to further decrease the training loss, in contrast to SAM and Bulk-SAM.**

As shown in Figure B.1, we observe that **Dom-GD fails to further decrease the training loss**, unlike GD. Moreover, **Bulk-GD is as effective as GD in decreasing the training loss**, despite only a small fraction of updates aligning with the bulk subspace.

B.2. Sharpness-Aware Minimization

Sharpness-Aware Minimization (SAM) [14] is a gradient-based optimization method designed to find flat minima. SAM has gained significant attention for its success in practice, especially in improving the generalization performance of deep learning models. For concreteness, we focus on the full-batch version of SAM applied to GD as the base optimizer. This leads to the following update equation:

$$\theta_{t+1} \leftarrow \theta_t - \eta \nabla L \left(\theta_t + \rho \frac{\nabla L(\theta_t)}{\|\nabla L(\theta_t)\|_2} \right),$$

where η is the learning rate and ρ represents the perturbation radius.

Method	**Mean Dom-LR**	**Mean Bulk-LR**
GD(-m)	0.0100 (0.0000)	**0.0100** (0.0000)
GD(+m)	0.0070 (0.0232)	**0.0828** (0.0576)
Adam(-m)	0.0325 (0.0054)	**0.4672** (0.3555)
Adam(+m)	0.0004 (0.0101)	**2.6639** (1.2480)

Table 1: Mean effective learning rates over the first 1000 steps (numbers in parentheses show standard deviation). Training Transformer on SST2-1k using GD and Adam with (+m) and without (-m) momentum. GD uses a learning rate of 0.01, and Adam uses a learning rate of 0.001. Momentum is set to $\beta = 0.9$.

A recent study [38] highlights that SAM also operates in its own Edge of Stability regime, wherein the sharpness $\lambda_1(\theta_t)$ saturates near SAM's stability threshold (see Figure B.2). This threshold, denoted as the *SAM-edge*, is defined as:

$$\frac{\|\nabla L(\theta_t)\|_2}{2\rho}\left(\sqrt{1+\frac{8}{\eta\,\|\nabla L(\theta_t)\|_2}}-1\right). \tag{SAM-edge}$$

Note that GD's stability threshold $2/\eta$ remains constant during training, while SAM-edge is a decreasing function of the norm of the gradient, so it tends to decrease during training.

As shown in Figure B.2, we observe that update vectors of SAM approximately align with dominant subspaces when sharpness saturates near the SAM-edge, similar to GD in the EoS regime. Next, we conduct experiments akin to Section 3: we train neural networks using SAM until the alignment occurs. Then, we switch SAM to the following schemes:

$$\theta_{t+1} \leftarrow \theta_t - \eta P_k(\theta_t)\nabla L\left(\theta_t + \rho\frac{\nabla L(\theta_t)}{\|\nabla L(\theta_t)\|_2}\right), \tag{Dom-SAM}$$

$$\theta_{t+1} \leftarrow \theta_t - \eta P_k^{\perp}(\theta_t)\nabla L\left(\theta_t + \rho\frac{\nabla L(\theta_t)}{\|\nabla L(\theta_t)\|_2}\right). \tag{Bulk-SAM}$$

We observe that **Dom-SAM fails to further decrease the training loss, unlike SAM and Bulk-SAM**, as depicted in Figure B.2. Our investigation with various practical algorithms suggests that *neural networks cannot be trained within the dominant subspace, and the bulk subspace plays an important role in learning.*

Appendix C. Momentum and adaptive methods amplify updates in bulk subspaces

In this section, we also extend our investigation to momentum optimizers, *e.g.*, SGD-momentum, and adaptive optimizers, *e.g.*, Adam. Now with momentum and adaptive optimizers, the update direction is no longer the same as the gradient direction, and Phenomenon 1 no longer holds for this case. However, Phenomenon 2 still holds, *i.e.*, the bulk subspace is still where the learning happens.

We build on our results so far and propose explanations for why momentum and adaptive methods are effective for neural network training. At a high level, we claim that they speed up training by amplifying the *bulk component* of each update step. Formally, we introduce the following notion.

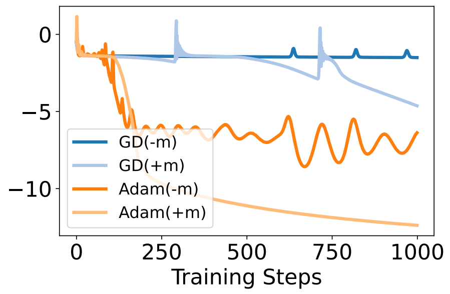


Figure 11: Training loss in log-scale for the experiments in Table 1.

Definition 4 (Effective learning rate) *For a given optimization trajectory* $\{\theta_t\}$, *we define the dominant effective learning rate (Dom-LR) at step t as:*

$$\eta_t^{\text{dom}} := \frac{\langle \theta_{t+1} - \theta_t, P_k(\theta_t)\nabla L(\theta_t)\rangle}{\|P_k(\theta_t)\nabla L(\theta_t)\|_2^2}\,, \tag{Dom-LR}$$

and the bulk effective learning rate (Bulk-LR) at step t as:

$$\eta_t^{\text{bulk}} := \frac{\left\langle \theta_{t+1} - \theta_t, P_k^{\perp}(\theta_t)\nabla L(\theta_t)\right\rangle}{\left\|P_k^{\perp}(\theta_t)\nabla L(\theta_t)\right\|_2^2}\,. \tag{Bulk-LR}$$

To understand the above notion better, we first consider the case of GD:

- GD with learning rate η has effective learning rates $\eta_t^{\text{dom}} = \eta_t^{\text{bulk}} = \eta$ for all steps t.
- Dom-GD with learning rate η has effective learning rates $\eta_t^{\text{dom}} = \eta$ and $\eta_t^{\text{bulk}} = 0$ for all steps t.
- Bulk-GD with learning rate η has effective learning rates $\eta_t^{\text{dom}} = 0$ and $\eta_t^{\text{bulk}} = \eta$ for all steps t.

Our primary claim in this section is as follows: since Dom-GD fails to decrease the training loss, while Bulk-GD is as effective as GD in reducing the training loss, we claim that Bulk-LR serves as a good indicator for training speed, unlike Dom-LR.

To support this claim, we measure the effective learning rates of various optimization methods, including (full-batch) GD with and without momentum, and (full-batch) Adam with and without momentum. Table 1 presents the effective learning rates when training Transformer on SST2-1k, and Figure 11 depicts the corresponding training loss plot. Additional experiments on other architectures and datasets or provided in Section H.

Across different settings, we consistently observe that Bulk-LR positively correlates with the training speed. Moreover, momentum and adaptive methods seem to amplify Bulk-LR. This offers new insights into the effectiveness of momentum and adaptive methods.

Appendix D. Related work

Gradient descent in tiny subspaces. This work is largely inspired by previous research demonstrating low-rank structures of the training loss Hessian and the gradient in deep neural network training [20, 44, 45]. In particular, Jastrzębski et al. [24] also observe that the SGD update direction is highly aligned with the sharpest direction of the loss landscape. Recently, Schneider et al. [46] observe that policy gradient algorithms in reinforcement learning also seem to operate in low-dimensional subspaces.

Motivated by such prevalent observations, several follow-up works investigate the possibility of training neural networks in a low-dimensional subspace. If feasible, it has wide applications, including few-shot learning [16] and differential privacy [48, 57].

Li et al. [35] and Gressmann et al. [19] train neural networks with a small fraction of parameters using random projections, and Li et al. [37] train ResNet8 on CIFAR10 with a 15-dimensional subspace without sacrificing test accuracy. Note that the results of these work do not contradict our main observation since the low-dimensional subspaces they chose are not the dominant subspace. In particular, Li et al. [37] take an adaptive approach to identify the 15-dimensional subspace based on given optimization trajectories. Along this line, Cosson et al. [9] and Jadbabaie et al. [23] design efficient optimization algorithms for the setting when the gradient aligns a the low-dimensional subspace.

From a theoretical perspective, Arous et al. [6] rigorously prove that the SGD update aligns with the dominant subspace in multi-class logistic regression and XOR classification with a two-layer network. However, whether their results can be extended to more practical settings considered in this work is not clear. Specifically, their results show that GD with small learning rates also aligns with the dominant subspace, which is not the case in our settings (see Section G.1 for details).

Edge of Stability. Most analyses of GD have focused on settings where the learning rate is sufficiently small to ensure that the training loss monotonically decreases. However, recent empirical studies [24, 25] observe that GD with practically large learning rates decreases the loss non-monotonically and finds flatter minima. Cohen et al. [8] call this the *Edge of Stability* (EoS) phenomenon. Subsequent theoretical works have made progress towards understanding the mechanisms of EoS [1, 13, 53]. Moreover, several recent works theoretically analyze precise training dynamics under simplified models [2, 30, 50, 58].

Sharpness-Aware Minimization. Inspired by prior works [26, 27] showing that flat minima often lead to better generalization, Foret et al. [14] propose an optimization method called *Sharpness-Aware Minimization* (SAM), designed to find flat minima. SAM has shown great success in practice, and subsequently, several works theoretically investigate the dynamics of SAM and its convergence properties [5, 7, 12, 47, 52]. Recently, Long and Bartlett [38] empirically show that SAM also goes through unstable dynamics, akin to EoS.

The role of momentum and Adam. Momentum [39, 43] and adaptive methods [29] are workhorses for training deep neural network models. Adaptive methods, such as Adam, have gained renewed interest due to their success in training language models [55]. However, the current understanding of their effectiveness for neural network training remains incomplete.

The role of momentum is quite well understood for convex settings, through acceleration mechanism [28, 39]. For nonconvex settings, the provable benefits of momentum are investigated for variants of SGD, such as normalized SGD [11] and signSGD [10]. A recent work by Wang et al. [51] shows that the benefit of momentum is marginal when the learning rate is small and gradient noise is dominant. Moreover, Fu et al. [15] empirically demonstrate the benefits of momentum for large learning rates from a sharpness perspective.

Adam has been observed to be particularly effective in training transformers [56], even for simplified shallow linear transformers trained on linear regression tasks [3]. Its superiority over SGD has been attributed to factors like heavy-tailed class imbalances in language tasks [33]. A recent line of work shows that full-batch Adam is a smoothed version of SignGD [32, 54]. Additionally, Ahn et al. [4] study the benefits of Adam from an online learning perspective.

Appendix E. Experimental details

In this section, we provide the details of our experiments which are not covered in the main text.

E.1. Architectures

The main experiments are conducted on three types of architectures: MLP, CNN, and Transformer. Additional experiments conducted on standard architectures are provided in Section F.2.

- **MLP**: We use a 3-layer MLP with a width of 200 and $\tanh$ activation functions, following the architecture used in Cohen et al. [8].
- **CNN**: We use a 3-layer CNN with a width of 32 and ReLU activation functions, also based on the architecture from Cohen et al. [8].
- **Transformer**: We use a 2-layer Transformer with a hidden dimension of 64 and 8 attention heads, based on the architecture used in Damian et al. [13].

E.2. Data

The main experiments are conducted on three datasets: MNIST-5k, CIFAR10-5k, and SST2-1k. The primary task is classification with categorical MSE loss, and additional experiments with cross-entropy loss are provided in Section F.1.

- **MNIST-5k**: We use the first 5000 samples of MNIST dataset [34] for multi-class classification. The number of classes is 10.
- **CIFAR10-5k**: We use the first 5000 samples of CIFAR10 dataset [31] for multi-class classification. The number of classes is 10.
- **SST2-1k**: We use the first 1000 samples of SST2 dataset [49] for binary classification.

E.3. Experimental setup

Throughout this paper, all experiments are conducted using a constant learning rate. For experiments using SGD, we use a batch size of 50 for all experiments. Below, we provide details on the choice of learning rates for each experiment, which are not specified in the main text.

- Figure 1, Figure 2, and Figure 3: The training loss, eigenvalues of the loss Hessian, and $\chi_k(\nabla L(\theta_t))$ are computed on the same run of SGD. The learning rates used are:
  - MLP on MNIST-5k: 0.01,
  - CNN on CIFAR10-5k: 0.001,
  - Transformer on SST2-1k: 0.001.
- Figure 4 and Figure 5: We train MLP on MNIST-5k using GD or SGD with a learning rate of 0.01, under the same initialization.
- Figure 12, Figure 13, and Figure 14: The eigenvalues of the loss Hessian, $\chi_k(\nabla L(\theta_t))$, and the training loss are computed on the same run of SGD. The learning rates used are:
  - MLP on MNIST-5k: 0.1,
  - CNN on CIFAR10-5k: 0.001,
  - Transformer on SST2-1k: 0.001.

- Figure 15, Figure 16, and Figure 17: The eigenvalues of the loss Hessian, $\chi_k(\nabla L(\theta_t))$, and the training loss are computed on the same run of SGD. The learning rates used are:
  - VGG11 on CIFAR10-5k: 0.01,
  - ResNet8 on CIFAR10-5k: 0.01.

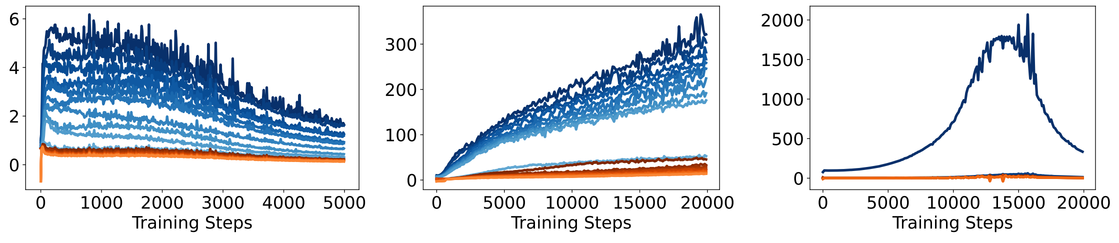
Our experiments were conducted using Pytorch [42], and we referred to the GitHub repository at `https://github.com/locuslab/edge-of-stability` to replicate the experimental setup described in Cohen et al. [8]. All experiments were performed on a single server equipped with 4 NVIDIA RTX 3090 GPUs.

Appendix F. Additional experiments for Section 3

In this section, we provide additional experimental results to support the observations made in Section 3. These experiments demonstrate that our critical observation—that Dom-SGD fails to further decrease the training loss—also holds when using cross-entropy loss and training with standard architectures.

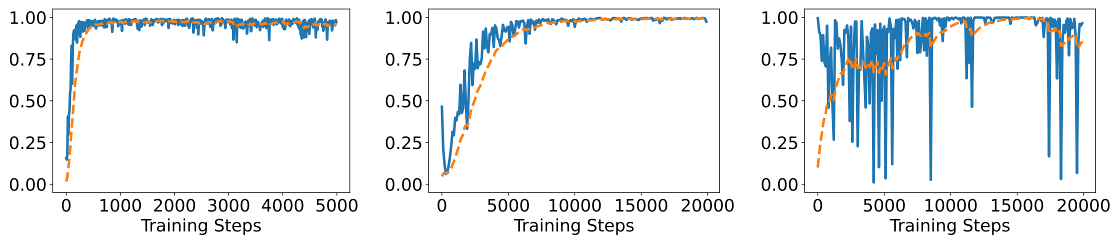
F.1. Cross-entropy loss

We use cross-entropy loss instead of MSE loss for classification tasks, and provide the results analgous to Figure 1, Figure 2, and Figure 3.



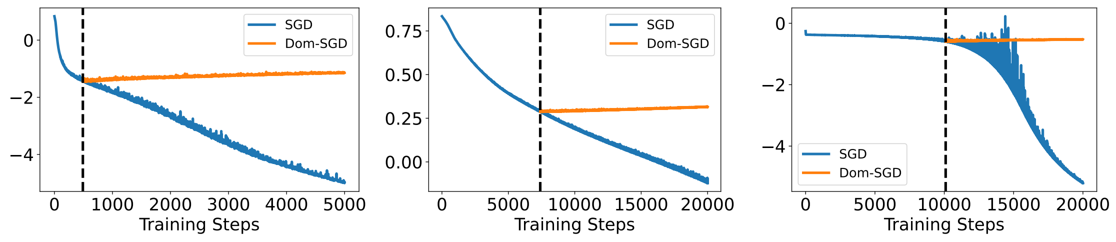
(a) MLP on MNIST-5k (b) CNN on CIFAR10-5k (c) Transformer on SST2-1k

Figure 12: **Low-rank structure of the Hessian.** The plot shows the top eigenvalues of the loss Hessian during SGD training. The blue curves represent the top-k eigenvalues, which are significantly larger than the next top-k eigenvalues, shown in orange, where k is the number of classes for the classification task.



(a) MLP on MNIST-5k (b) CNN on CIFAR10-5k (c) Transformer on SST2-1k

Figure 13: **Alignment of gradients with dominant subspaces.** The plot illustrates $\chi_k(\nabla L(\theta_t))$ during SGD training. The orange dashed lines represent the exponential moving average (EMA) of $\chi_k(\nabla L(\theta_t))$. After a few early steps, $\chi_k(\nabla L(\theta_t))$ reaches and stays near 1, indicating the alignment between gradients and dominant subspaces.

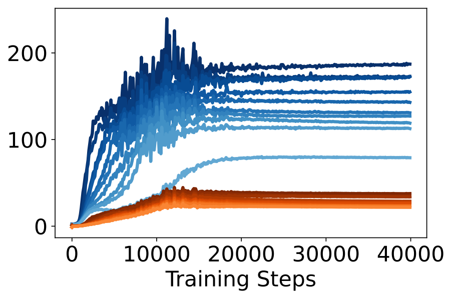


(a) MLP on MNIST-5k (b) CNN on CIFAR10-5k (c) Transformer on SST2-1k

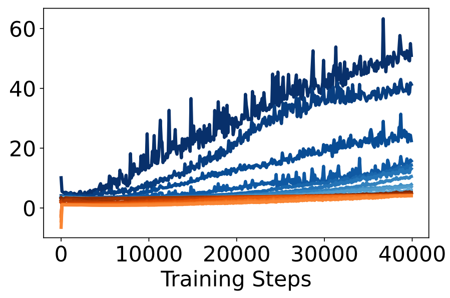
Figure 14: **Dom-SGD fails to further decrease the training loss** in contrast to SGD, despite the gradients aligning approximately with the dominant subspace. We switch from SGD to Dom-SGD whenever the EMA value of $\chi_k(\nabla L(\theta_t))$ exceeds 0.95. Training loss plots are shown in log-scale.

F.2. Standard architectures on CIFAR10-5k

To ensure the generality of our observations, we conducted experiments on standard architectures, specifically VGG11 and ResNet8, using CIFAR10-5k dataset, using MSE loss.

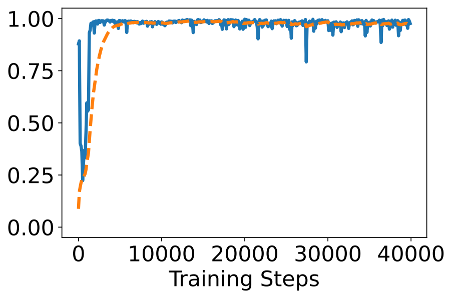


(*a*) VGG11 on CIFAR10-5k

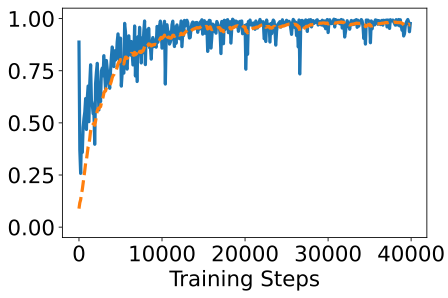


(*b*) ResNet8 on CIFAR10-5k

Figure 15: **Low-rank structure of the Hessian.** The plot shows the top eigenvalues of the loss Hessian during SGD training. The blue curves represent the top-10 eigenvalues, which are significantly larger than the next top-10 eigenvalues, shown in orange.

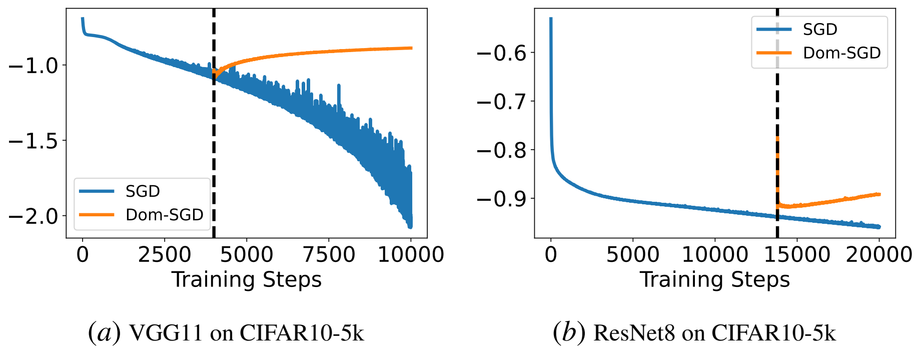


(*a*) VGG11 on CIFAR10-5k



(*b*) ResNet8 on CIFAR10-5k

Figure 16: **Alignment of gradients with dominant subspaces.** The plot illustrates $\chi_{10}(\nabla L(\theta_t))$ during SGD training. The orange dashed lines represent the exponential moving average (EMA) of $\chi_{10}(\nabla L(\theta_t))$. After a few early steps, $\chi_{10}(\nabla L(\theta_t))$ reaches and stays near 1, indicating the alignment between gradients and dominant subspaces.



(a) VGG11 on CIFAR10-5k (b) ResNet8 on CIFAR10-5k

Figure 17: **Dom-SGD fails to further decrease the training loss** in contrast to SGD, despite the gradients aligning approximately with the dominant subspace. We switch from SGD to Dom-SGD whenever the EMA value of $\chi_k(\nabla L(\theta_t))$ exceeds 0.95. Training loss plots are shown in logarithmic scale.

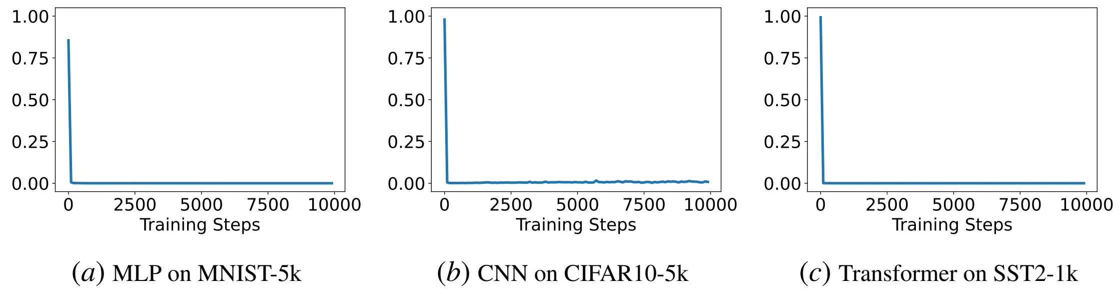
In conclusion, these additional experiments support our primary observation that neural networks cannot be effectively trained in the dominant subspace alone and that the bulk subspace plays a crucial role, regardless of the loss function or architecture used.

Appendix G. Additional experiments for Section A

This section provides additional experimental results to support the observations made in Section A.

G.1. No alignment with dominant subspaces along GD trajectories

We run (full-batch) GD under the same settings as Figure 3, using the same (small) learning rates and initializations. As shown in Figure 18, $\chi_k(\nabla L(\theta_t))$ quickly approaches and remains near 0, indicating that gradients do not align with dominant subspaces, unlike in SGD.



(a) MLP on MNIST-5k (b) CNN on CIFAR10-5k (c) Transformer on SST2-1k

Figure 18: **Gradients do not align with dominant subspaces on GD trajectories.** The plot illustrates $\chi_k(\nabla L(\theta_t))$ during GD training. After a few early steps, $\chi_k(\nabla L(\theta_t))$ reaches and stays near 0, indicating alignment with bulk subspaces. The same learning rates and initializations as Figure 3 are used.

G.2. SGD trajectories track gradient flow

Figure 3 and Figure 18 suggest that GD and SGD exhibit different alignments with dominant subspaces, even when using the same learning rate and initialization. However, both should track their continuous dynamics, i.e., the gradient flow, if the learning rate is sufficiently small. In Figure 19, we confirm that the trajectory of GD $\{\theta_t^{\mathrm{GD}}\}$ and the trajectory of SGD $\{\theta_t^{\mathrm{SGD}}\}$ are close to each other. Specifically, we observe that $\left\|\theta_t^{\mathrm{GD}} - \theta_t^{\mathrm{SGD}}\right\| \ll \left\|\theta_t^{\mathrm{GD}} - \theta_0^{\mathrm{GD}}\right\| \approx \left\|\theta_t^{\mathrm{SGD}} - \theta_0^{\mathrm{SGD}}\right\|$, indicating the trajectories are quite similar.

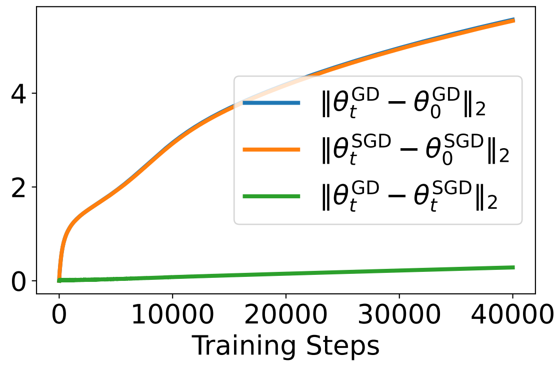
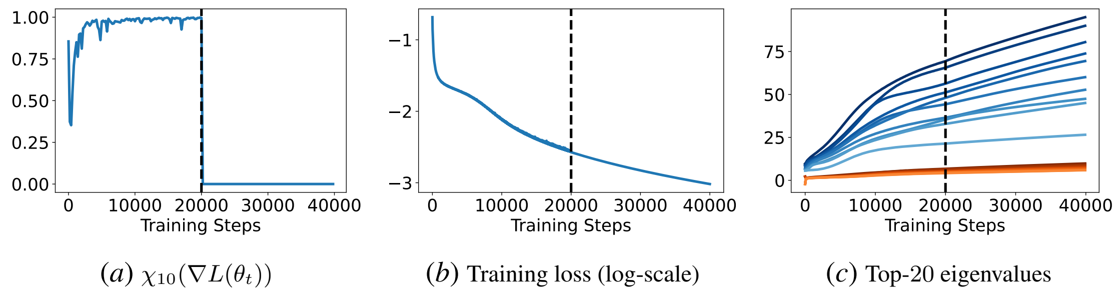


Figure 19: Trajectories of GD and SGD are close to each other.

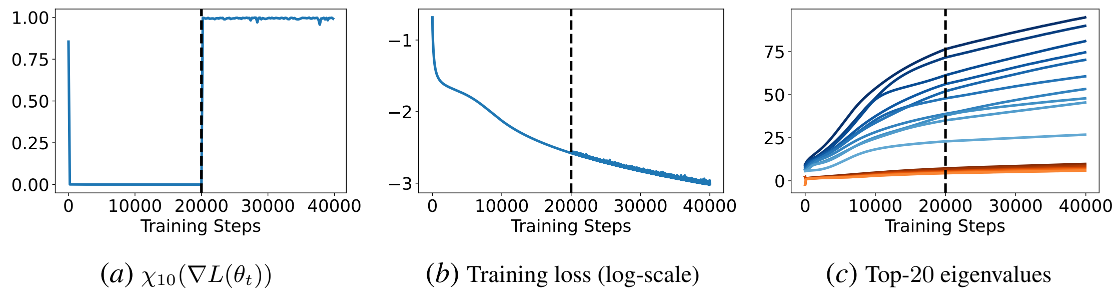
G.3. Switching between GD and SGD

Here, we provide additional plots for the experiments shown in Figure 4 and Figure 5. Figure 20 illustrates $\chi_k(\nabla L(\theta_t))$, training loss, and top eigenvalues of the loss Hessian when switching from SGD to GD, corresponding to the experiment in Figure 4. Figure 21 illustrates $\chi_k(\nabla L(\theta_t))$, training loss, and top eigenvalues of the loss Hessian when switching from GD to SGD, corresponding to the experiment in Figure 5.



(a) $\chi_{10}(\nabla L(\theta_t))$ (b) Training loss (log-scale) (c) Top-20 eigenvalues

Figure 20: The left plot shows $\chi_k(\nabla L(\theta_t))$ when training MLP on MNIST-5k. A sharp transition in gradient alignment with the dominant subspace is observed when switching from SGD to GD (same plot as Figure 4). The plots of the training loss and top eigenvalues of the loss Hessian change relatively smoothly when switching from SGD to GD, in contrast to $\chi_{10}(\nabla L(\theta_t))$. In the right plot, the blue curves represent the top-10 eigenvalues, and the orange curves represent the next top-10 eigenvalues.

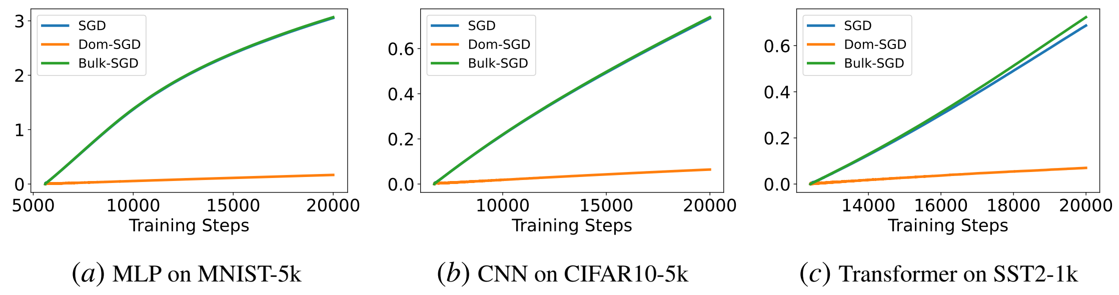


(a) $\chi_{10}(\nabla L(\theta_t))$ (b) Training loss (log-scale) (c) Top-20 eigenvalues

Figure 21: The left plot shows $\chi_k(\nabla L(\theta_t))$ when training MLP on MNIST-5k. A sharp transition in gradient alignment with the dominant subspace is observed when switching from GD to SGD (same plot as Figure 5). The plots of the training loss and top eigenvalues of the loss Hessian change relatively smoothly when switching from GD to SGD, in contrast to $\chi_{10}(\nabla L(\theta_t))$. In the right plot, the blue curves represent the top-10 eigenvalues, and the orange curves represent the next top-10 eigenvalues.

G.4. Distance from "switching" step

We measure the ℓ_2 distance of the weights from the step where we switch from SGD to Dom-SGD and Bulk-SGD for the experiments in Figure 1. As shown in Figure 22, the distance from the switching step remains relatively small and tends to saturate for Dom-SGD. In contrast, for both SGD and Bulk-SGD, the distance from the switching step increases much faster and in a similar manner. This observation supports our illustration in Figure 8.



(a) MLP on MNIST-5k (b) CNN on CIFAR10-5k (c) Transformer on SST2-1k

Figure 22: The plots illustrate the ℓ_2 distance of the weights from the step where we switch from SGD to Dom-SGD and Bulk-SGD for the experiments in Figure 1. We observe that Dom-SGD fails to make further progress in terms of distance, in contrast to SGD and Bulk-SGD, supporting our illustration in Figure 8.

Appendix H. Additional experiments for Section C

This section provides additional experimental results to support the observations made in Section C. We measure the effective learning rates for (full-batch) GD with and without momentum, and (full-batch) Adam with and without momentum on different architectures and datasets. Tables 2 and 3 present the effective learning rates when training an MLP on MNIST-5k and a CNN on CIFAR10-5k. Figures 23 and 24 show the corresponding training loss plots. We consistently observe that (1) a higher Bulk-LR positively correlates with increased training speed, and (2) momentum and adaptive learning rates in Adam amplify Bulk-LR, resulting in a larger Bulk-LR compared to Dom-LR.

Method	Mean Dom-LR	Mean Bulk-LR
GD(-m)	0.0100 (0.0000)	**0.0100** (0.0000)
GD(+m)	0.0012 (0.0092)	**0.1005** (0.0068)
Adam(-m)	0.3317 (0.0571)	**0.4021** (0.0868)
Adam(+m)	0.0356 (0.0194)	**0.7301** (0.4717)

Table 2: Mean effective learning rates over the first 1000 steps (numbers in parentheses show standard deviation). Training MLP on MNIST-5k using GD and Adam with (+m) and without (-m) momentum. GD uses a learning rate of 0.01, Adam uses a learning rate of 0.001. Momentum is set to $\beta = 0.9$.

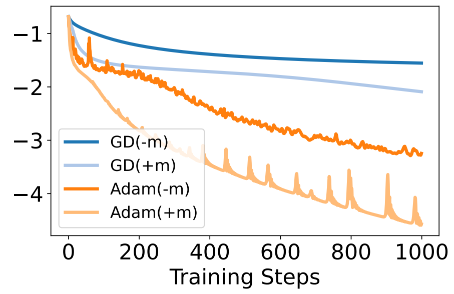


Figure 23: Training loss in log-scale for the experiments in Table 2.

Method	Mean Dom-LR	Mean Bulk-LR
GD(-m)	0.0010 (0.0000)	**0.0010** (0.0000)
GD(+m)	0.0010 (0.0023)	**0.0101** (0.0007)
Adam(-m)	0.0191 (0.0020)	**0.0289** (0.0053)
Adam(+m)	0.0046 (0.0026)	**0.0802** (0.0194)

Table 3: Mean effective learning rates over the first 1000 steps (numbers in parentheses show standard deviation). Training CNN on CIFAR10-5k using GD and Adam with (+m) and without (-m) momentum. GD uses a learning rate of 0.001, and Adam uses a learning rate of 10^{-4}. Momentum is set to $\beta = 0.9$.

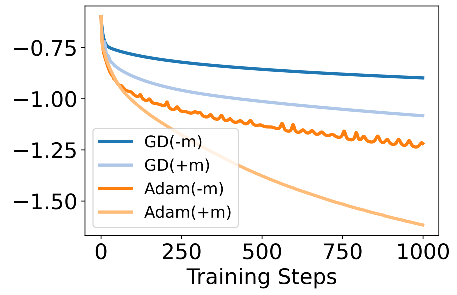


Figure 24: Training loss in log-scale for the experiments in Table 3.