Topological Control of Optimization Dynamics on Evolving Manifolds

Xingrui Gu [1]

Abstract

Machine learning optimization is usually framed as a sequence of local responses to the current gradient. In high-dimensional nonconvex training, however, it is more naturally understood as a noise-driven process of trajectory evolution. Motivated by symplectic reduction, which compresses high-dimensional dynamics onto a lower-dimensional effective manifold while preserving essential structure, we propose **Trajectory-Adaptive Geometric Damping (TAGD)**, a geometry-topology control framework for first-order optimization. TAGD uses recent gradient history to estimate a low-rank effective subspace via online Gram-PCA, decomposes each update into tangential and normal components, and adaptively regulates normal suppression through multiscale topological stability feedback. This yields a trajectory-level control mechanism on a low-dimensional effective submanifold, rather than a purely local update rule. Theoretically, under local Fisher geometry, we show that the resulting update reduces information-geometric error, tightens the one-step descent bound, and remains robust to finite-window subspace estimation error; we further show that topological feedback acts as an online proxy for an otherwise unobservable orthogonal bias-variance tradeoff. Empirically, across CIFAR-10 and WikiText-103, TAGD improves performance in 11 of 12 settings without modifying the internal state of the base optimizer. On WikiText-103, relative to AdamW, it improves average best validation perplexity by 8.26% on LLaMA-style models and 5.96% on GPT-J-style models, for an overall gain of 7.11%. Trajectory and local loss-landscape visualizations further show that TAGD enters low-loss channels earlier, reduces lateral drift, and in some cases alters basin selection. Overall, TAGD unifies information geometry, low-rank subspace control, and topological stability feedback within an online optimization framework, offering a trajectory-centric view of deep learning optimization.

[1]Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, Berkeley, CA, USA. Correspondence to: Xingrui Gu <xingrui_gu@berkeley.edu>.

The 3rd *AI for Math Workshop* at the 43rd International Conference on Machine Learning (ICML), Seoul, South Korea, 2026.
Copyright 2026 by the author(s).

1. Introduction

Mainstream optimizers remain fundamentally trajectory-blind, refining updates via instantaneous local information while lacking mechanisms to control the long-horizon geometry of the optimization path (Bottou et al., 2018; Adam et al., 2014; Loshchilov & Hutter, 2017). In underspecified regimes, where optimization acts as a trajectory-dependent inductive bias (Cortesi et al., 2026), deep networks evolve along low-dimensional effective channels (Li et al., 2017; Aghajanyan et al., 2021); however, accumulated mini-batch noise often reshapes the diffusion geometry, driving locally reasonable steps into unstable orthogonal drift or bifurcations between metastable basins (Mandt et al., 2017; Hu et al., 2017; Raginsky et al., 2017). These failures reflect hidden functional divergence—such as fragmentation of descent channels or loss of connectivity (Garipov et al., 2018; Draxler et al., 2018; Sagun et al., 2017; Ghorbani et al., 2019), manifesting as shortcut attraction (Geirhos et al., 2020), catastrophic forgetting (Kirkpatrick et al., 2017), or delayed grokking (Power et al., 2022).

Inspired by symplectic reduction and empirical evidence that deep-network training trajectories concentrate in a low-dimensional subspace (Li et al., 2018; Gur-Ari et al., 2018; Gao et al., 2021), as shown in Fig 2, we propose **Trajectory-Adaptive Geometric Damping (TAGD)**, a geometry–topology control framework that treats optimization as a controlled stochastic trajectory rather than a purely local iterative update. TAGD uses recent gradient history to estimate a low-rank effective tangent space via online Gram-PCA, and reshapes each gradient by preserving motion along the dominant descent channel while selectively damping orthogonal components. In this way, the update is controlled not by a fixed projection, but by a time-varying anisotropic metric induced by the recent trajectory itself.

The framework makes three contributions. First, it intro-

duces a low-rank subspace control principle grounded in local information geometry, and we show that this controlled update enjoys explicit advantages in Fisher accuracy, one-step descent, and robustness under finite-window subspace estimation. Second, it turns topological stability from a post-hoc analytical tool into an online control signal: multi-scale connectivity and spectral-gap proxies extracted from gradient-history filtrations are used to adaptively regulate normal suppression, thereby operationalizing an otherwise unobservable bias–variance tradeoff. Third, it advances a trajectory-level view of optimization, in which training is understood as controlled evolution on a low-dimensional effective submanifold; the symplectic and dynamical-systems perspective provides the motivating language of reduction and flow structure, while the resulting method remains fully compatible with standard first-order optimizers.

2. Preliminaries

Let the parameter space be $\Theta \subseteq \mathbb{R}^D$, with model parameter $\theta \in \Theta$, and objective function $L : \Theta \to \mathbb{R}$. In classical first-order optimization, gradient descent updates are written as

$$\theta_{t+1} = \theta_t - \eta_t g_t, \tag{1}$$

where $\eta_t > 0$ is the learning rate and g_t is the gradient used at step t. For full-batch optimization, $g_t = \nabla L(\theta_t)$; under mini-batch training, one typically writes

$$g_t = \nabla L(\theta_t) + \xi_t, \tag{2}$$

where ξ_t denotes the sampling-induced perturbation, often assumed to have zero conditional mean. When the step size is sufficiently small, the discrete iteration can be viewed as the continuous gradient flow

$$\dot{\theta}(t) = -\nabla L(\theta(t)), \tag{3}$$

while stochastic gradient methods admit a diffusion approximation of the form

$$d\theta_t = -\nabla L(\theta_t)\, dt + \Sigma(\theta_t)^{1/2} dW_t, \tag{4}$$

where W_t is a Brownian motion and $\Sigma(\theta_t)$ denotes the noise covariance. Thus, modern optimization may be interpreted as a parameter evolution jointly driven by a drift term (descent along the loss landscape) and a diffusion term (stochastic exploration) (Robbins & Monro, 1951).

More generally, many optimizers can be written in the unified form

$$\theta_{t+1} = \theta_t - \eta_t M_t g_t, \tag{5}$$

where $M_t \in \mathbb{R}^{D\times D}$ is a time-varying linear operator, which may be interpreted as a preconditioner, a local metric, or an effective diffusion tensor. Under this view, optimization is no longer ordinary steepest descent in a fixed Euclidean geometry, but descent under a time-dependent effective geometry.

3. Realizing Geometry-Topology Control

The previous chapter established that history-induced low-rank subspace control can improve update quality in a local information-geometric sense and stabilize optimization through adaptive normal suppression. We now instantiate this principle as an executable optimization system. The implementation is not unique: any method realizing three ingredients may be viewed as an instance of the framework, namely **(1)** estimating a low-rank local information subspace from recent optimization history, **(2)** decomposing the current gradient into tangential and normal components, and **(3)** adaptively modulating normal suppression according to local structural stability. In this work, we adopt one such realization in the below.

3.1. Implementation Mapping

The control law derived in Equation 19 where P_t denotes the projector onto the local information subspace and $\lambda_{\perp,t} \in [0, 1]$ is the normal suppression coefficient. In implementation, P_t is estimated from recent gradient history, $\lambda_{\perp,t}$ is computed from topological and spectral stability signals of the projected trajectory, and $\hat{g}_t$ is injected as the consumed gradient of the base optimizer. The resulting system does not alter the optimizer's internal mechanics; it realizes the theory of Chapter A.2 as an optimizer-agnostic geometry–topology control layer applied immediately before the first-order update.

3.2. Trajectory-Adaptive Geometric Damping(TAGD)

Trajectory Subspace Estimation The first component is a finite trajectory history window used to estimate the local low-rank update subspace. Let

$$\mathcal{B}_t = \{g_{t-m+1}, \dots, g_t\} \tag{6}$$

denote the most recent m raw stochastic gradients. These gradients are treated as a short-horizon trajectory segment of the local optimization flow rather than as independent stepwise corrections. Stacking the gradients row-wise yields

$$\mathbf{G}_t \in \mathbb{R}^{m\times D}. \tag{7}$$

After centering the window, we construct the sample-space Gram matrix

$$C_t = \frac{1}{D}\mathbf{G}_t\mathbf{G}_t^\top \in \mathbb{R}^{m\times m}. \tag{8}$$

We then compute its top-k eigendirections and map them back to parameter space to obtain an orthonormal basis

$$U_t \in \mathbb{R}^{D\times k}, \qquad U_t^\top U_t = I_k, \tag{9}$$

which defines the empirical projector

$$P_t = U_t U_t^\top. \tag{10}$$

This provides a tractable surrogate for the oracle projector in Chapter A.2. The use of sample-space Gram PCA avoids explicit construction of a full $D \times D$ covariance operator and remains practical in high-dimensional settings. The resulting subspace should be interpreted as a history-induced approximation to the local information subspace. To reduce overhead and sensitivity to stepwise noise, the basis is updated periodically rather than at every iteration.

Topology Guided Control Given the basis U_t, we project the windowed trajectory history into the estimated low-rank space:

$$Z_t = (\mathbf{G}_t - \bar{\mathbf{G}}_t)U_t \in \mathbb{R}^{m\times k}, \tag{11}$$

where $\bar{\mathbf{G}}_t$ is the window mean. The projected cloud Z_t provides a low-dimensional representation of the recent local flow. In the theory of Chapter A.2, the optimal normal suppression coefficient is governed by an unobservable orthogonal bias–variance tradeoff. The role of topology in the implementation is not to reconstruct the full topology of the loss landscape, but to provide an online proxy for whether the projected trajectory remains organized around a stable dominant channel or is instead approaching fragmentation, weak connectivity, or transition-like behavior. To quantify this effect, we construct multiscale similarity graphs on the rows of Z_t. After centering and per-dimension normalization, we compute pairwise cosine similarities and, for each threshold τ in a predefined scale set, define

$$A_{ij}^{(\tau)} = \mathbf{1}\left[\frac{\langle z_i, z_j\rangle}{\|z_i\|\,\|z_j\|} \geq \tau\right]. \tag{12}$$

From these graphs, we extract a small set of stability descriptors: the number of connected components, a graph-based proxy for β_1, the Laplacian spectral gap, average directional agreement, and a near-criticality proxy from unusually small gradient norms in the window. Larger component count or cycle proxy indicates fragmentation or inconsistency, a smaller spectral gap indicates weaker connectivity, lower directional agreement indicates reduced coherence, and a larger near-critical fraction indicates proximity to weak-gradient or transition-like regions. We aggregate these descriptors into a scalar instability score

$$s_t \in [0, 1], \tag{13}$$

and map it to the normal suppression coefficient through a monotone control rule

$$\lambda_{\perp,t} = \Lambda(s_t), \qquad \Lambda : [0, 1] \to [\lambda_{\min}, 1]. \tag{14}$$

In our implementation, the control map is realized as a stability-weighted shrinkage rule: greater structural instability yields a smaller $\lambda_{\perp,t}$. Concretely, multiscale connectivity, cycle, directional-consistency, spectral-gap, and near-criticality indicators are aggregated into a scalar control signal that rescales a baseline shrinkage level and is clipped to a prescribed safe interval. Thus, topology does not determine the update direction itself; rather, it regulates the strength of orthogonal suppression through online stability feedback, providing a practical proxy for the latent bias–variance tradeoff characterized in Chapter A.2. This design also reflects a computational tradeoff between structural fidelity and online efficiency. Because full persistent homology is typically too expensive for routine training-time use, the default controller employs graph-topological and spectral surrogates that can be evaluated efficiently at each control step. When additional computational budget is available, richer persistence-based features can be incorporated without changing the control logic. The topology module should therefore be viewed not as an auxiliary embellishment, but as a lightweight and structurally grounded source of feedback.

Scheduling and Optimizer Integration The controlled gradient is not applied uniformly throughout training. Instead, the consumed gradient is

$$g_t^{\text{used}} = (1 - \omega_t)g_t + \omega_t g_t^{\text{ctrl}}, \qquad \omega_t \in [0, 1], \tag{15}$$

where

$$g_t^{\text{ctrl}} = g_{\|,t} + \lambda_{\perp,t} g_{\perp,t}, \quad g_{\|,t} = P_t g_t, \quad g_{\perp,t} = (I - P_t)g_t. \tag{16}$$

This yields a continuous interpolation between the raw optimizer and the geometry-topology controlled update.

The implementation uses a stage-wise schedule. During *warm-up*, the method defaults to the base optimizer with $\lambda_{\perp,t} = 1$, allowing sufficient trajectory statistics to accumulate for subspace estimation. During *ramp-up*, the base shrinkage level is reduced gradually rather than abruptly. During *active control*, the low-rank basis is updated periodically and $\lambda_{\perp,t}$ is modulated online by topology-driven feedback. During *fade-out*, the intervention weight ω_t decreases to zero, so that training returns smoothly to the native behavior of the base optimizer. This schedule reflects the local nature of the theory in Chapter A.2: strong control is undesirable before the recent trajectory provides a reliable estimate of local geometry, and often unnecessary once large-scale structural drift becomes less prominent.

The resulting gradient g_t^{used} is then passed to the base optimizer. Since the optimizer state is left unchanged, the method remains compatible with SGD, AdamW, and other first-order optimizers. Algorithm 1 in Appendix. C summarizes the resulting geometry–topology control layer.

4. Experiment

4.1. Experiment Setup

This chapter evaluates the proposed trajectory-geometric control method on two settings: controlled small-scale vi-

Setting	Base	TAGD
Panel A: CIFAR-10 Accuracy (%) ↑		
MLP + AdamW	79.79	**79.93** (+0.14)
CNN + AdamW	82.99	82.99 (+0.00)
ViT + AdamW	73.02	**73.39** (+0.37)
MLP + SGD	76.27	**77.95** (+1.68)
CNN + SGD	84.95	**85.05** (+0.10)
ViT + SGD	69.75	**70.95** (+1.20)
Panel B: WikiText-103 PPL ↓		
LLaMA-100M	108.73	**97.76** (-10.97)
LLaMA-250M	139.91	**126.62** (-13.29)
LLaMA-350M	127.76	**120.93** (-6.83)
GPT-J-70M	114.50	**108.91** (-5.59)
GPT-J-125M	106.54	**104.68** (-1.86)
GPT-J-200M	143.82	**127.62** (-16.20)

Table 1. Performance on vision and language tasks. TAGD improves 11/12 settings, with larger gains under SGD on CIFAR-10 and in higher-perplexity regimes on WikiText-103.

sion experiments on CIFAR and autoregressive language modeling on WikiText-103. The main experiments use WikiText-103-raw-v1, with roughly 103M training tokens and 218K validation tokens, tokenized with the GPT-2 BPE tokenizer; the auxiliary CIFAR experiments cover three model classes, PatchMLP, SimpleCNN, and ViT (Model detail shown in Appendix D.1), to test transfer across modalities and architectures. All comparisons strictly control for model architecture, dataset, random seed, learning rate, weight decay, batch size, number of epochs, and training budget; the only difference is whether the proposed method is inserted before the base optimizer. The language-model experiments span two structurally distinct families, LLaMA-style and GPT-J-style (model details are provided in the appendix), each evaluated at multiple scales. The primary metric is best validation perplexity (best val PPL), which better reflects the best generalization point during training than final val PPL; for CIFAR, we additionally report loss and accuracy. The method does not modify the internal states of base optimizers such as AdamW or SGD, but only rewrites their input gradients, so it preserves the parameterization and core update rule of the baseline, with overhead arising only from the finite gradient buffer and low-rank geometric estimation. Beyond the main results, we further analyze architecture and scale effects, and use trajectory and local loss-landscape visualizations to examine the proposed mechanism.

4.2. Performance Results

This section evaluates the proposed method at two levels: controlled small-scale vision experiments on CIFAR-10(Krizhevsky et al., 2009) and large-scale autoregressive language modeling on WikiText-103(Merity et al., 2016). These two settings serve complementary purposes. The CIFAR experiments test whether the method generalizes across architectures and base optimizers in a controlled regime, while the WikiText-103 experiments provide the main evidence for its effectiveness on high-dimensional, long-horizon language-model training.

Overall, table 1 and 3 in Appendix, TAGD improves performance in both settings. On CIFAR-10, across PatchMLP, SmallCNN, and MicroViT, it does not degrade training and yields accuracy gains with lower loss in most cases: under AdamW, the average accuracy gain is about 0.17 points with an average loss reduction of 0.0048, while under SGD the gains are larger, with an average accuracy improvement of 0.99 points and an average loss reduction of 0.0251. This suggests that the benefit of history-induced subspace control is more pronounced when the base optimizer provides weaker local geometric adaptation. On WikiText-103, which constitutes the main benchmark, Table 1 shows that TAGD outperforms the AdamW baseline in all six language-model settings. The average best validation perplexity improves from 125.47 to 115.10 for the LLaMA family (8.26%), from 121.62 to 113.74 for the GPT-J family (5.96%), and from 123.54 to 114.42 overall (7.11%). The largest gains occur for GPT-J-200M (11.26%) and LLaMA-100M / 250M (10.09% / 9.50%), while the smallest gain is observed for GPT-J-125M (1.75%). Taken together, these results indicate that TAGD is not a task- or optimizer-specific trick, but a transferable trajectory-control mechanism whose gains arise from systematic intervention on optimization geometry rather than additional parameters or training budget. A detailed empirical analysis of wall-clock time, memory footprint, and amortized compute overhead is provided in Appendix D.2.

4.3. Trajectory and Local Loss-Landscape Analysis

As shown in Figure 1 and Figure 3, to better understand the effect of TAGD, we further examine training trajectories on locally projected loss landscapes (The Appendix D.3 has shown how to visualization Loss-Landscape. These visualizations are not intended to recover the true global geometry of the full parameter space; rather, they serve as mechanistic evidence for how TAGD changes trajectory organization relative to the baseline. Across the visualized settings, four consistent patterns emerge: **(1)** TAGD tends to leave high-loss boundary regions earlier and enter the dominant low-loss channel more quickly; **(2)** once inside the low-loss region, TAGD follows a more concentrated trajectory with reduced lateral drift, indicating stronger channel coherence; **(3)** the baseline more often slides along high-curvature boundaries or ridge-like regions, whereas TAGD more consistently progresses along the valley floor, suggesting reduced ineffective orthogonal motion; and **(4)** in several cases, the two methods converge to visibly different projected basins, with trajectory separation typically emerging only in the middle or later stages of training, consistent with the warm-up, ramp-up, and active-control schedule used

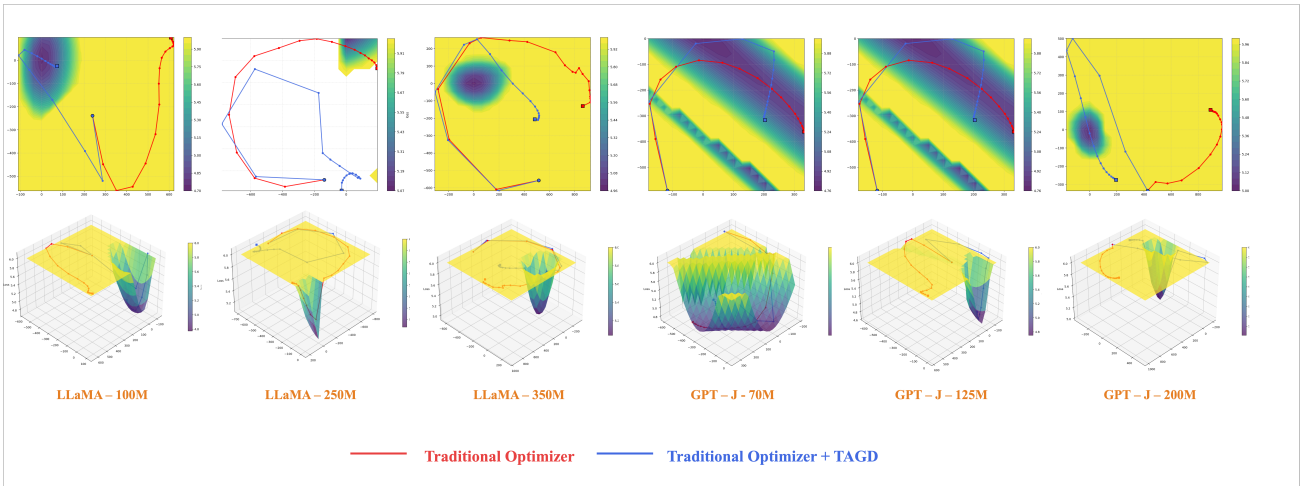

Figure 1. **Trajectory and local loss-landscape visualization on WikiText-103 language models.** We visualize projected training trajectories for three LLaMA-style models and three GPT-J-style models, comparing the AdamW baseline (red) with TAGD (blue). The top row shows 2D projected loss contours, and the bottom row shows the corresponding 3D local loss landscapes. Despite substantial variation across model families and scales, several consistent patterns emerge: TAGD tends to leave high-loss boundary regions earlier, follows low-loss channels with stronger coherence and reduced lateral drift, and in several cases converges to a visibly different projected basin from the baseline. The trajectory separation often appears only after the middle stage of training, which is consistent with the warm-up, ramp-up, and active-control schedule of TAGD. Overall, these visualizations suggest that TAGD changes how optimization enters, follows, and sometimes selects low-loss channels, supporting the trajectory-level control view of optimization developed in this paper.

by TAGD. Taken together, these observations suggest that TAGD does not merely perturb the final solution locally, but changes how optimization enters, follows, and occasionally selects low-loss channels. This behavior is consistent with the theoretical picture developed in Chapter A.2: a low-rank dominant channel is preserved, orthogonal instability is suppressed, and topology-driven feedback modulates the strength of that suppression over time.

5. Related Work

In recent years, topology and information geometry have both become increasingly relevant to machine learning and optimization, but mostly along separate lines. Persistent-homology-based methods typically treat topology as a regularizer, constraint, or optimization objective, encouraging models to satisfy desired topological properties (Nigmetov et al., 2024; Zhang et al., 2022). Topology has also been used for post hoc loss-landscape analysis, to characterize basin structure, connectivity, and geometric complexity rather than to guide training itself (Xie et al., 2024). By contrast, information geometry has mainly been used to identify locally effective directions through distribution-induced metrics and low-dimensional projection structures (Shumaylov et al., 2025). Overall, existing work shows that topology can serve as either a regularizer or an analysis tool, while information geometry provides a local criterion for useful updates. What is still missing is a unified framework that views optimization as a trajectory-evolution process rather than a sequence of local steps, and that turns the topological stability of low-dimensional projected trajectories into an online control signal for regulating updates that drift away from the locally effective submanifold.

6. Conclusion

This paper presented Trajectory-Adaptive Geometric Damping (TAGD), a framework that recasts deep learning optimization as controlled trajectory evolution on a low-dimensional effective submanifold rather than purely local iterative updates. TAGD estimates a low-rank effective subspace from recent gradient history, decomposes each update into tangential and normal components, and adaptively modulates normal suppression via topological stability feedback. Under local Fisher geometry, we showed that this controlled update improves information-geometric accuracy, yields a tighter one-step descent bound, and remains robust to finite-window subspace estimation error. Empirically, TAGD produced consistent gains on both CIFAR-10 and WikiText-103, while trajectory and local loss-landscape visualizations suggested that these gains arise from earlier entry into low-loss channels and more stable channel following. Overall, the results suggest that combining information geometry, low-rank subspace control, and topological stability feedback offers a principled perspective on the trajectory dynamics of deep learning optimization.

References

Adam, K. D. B. J. et al. A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 1412(6), 2014.

Aghajanyan, A., Gupta, S., and Zettlemoyer, L. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In *Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing (volume 1: long papers)*, pp. 7319–7328, 2021.

Black, S., Biderman, S., Hallahan, E., Anthony, Q., Gao, L., Golding, L., He, H., Leahy, C., McDonell, K., Phang, J., et al. Gpt-neox-20b: An open-source autoregressive language model. In *Proceedings of BigScience Episode# 5–Workshop on Challenges & Perspectives in Creating Large Language Models*, pp. 95–136, 2022.

Bottou, L., Curtis, F. E., and Nocedal, J. Optimization methods for large-scale machine learning. *SIAM review*, 60(2):223–311, 2018.

Cortesi, F. V., Iannone, G., Crippa, G., Poggio, T., and Beneventano, P. Same error, different function: The optimizer as an implicit prior in financial time series. *arXiv preprint arXiv:2603.02620*, 2026.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Draxler, F., Veschgini, K., Salmhofer, M., and Hamprecht, F. Essentially no barriers in neural network energy landscape. In *International conference on machine learning*, pp. 1309–1318. PMLR, 2018.

Gao, B., Son, N. T., Absil, P.-A., and Stykel, T. Geometry of the symplectic stiefel manifold endowed with the euclidean metric. In *International Conference on Geometric Science of Information*, pp. 789–796. Springer, 2021.

Garipov, T., Izmailov, P., Podoprikhin, D., Vetrov, D. P., and Wilson, A. G. Loss surfaces, mode connectivity, and fast ensembling of dnns. *Advances in neural information processing systems*, 31, 2018.

Geirhos, R., Jacobsen, J.-H., Michaelis, C., Zemel, R., Brendel, W., Bethge, M., and Wichmann, F. A. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

Ghorbani, B., Krishnan, S., and Xiao, Y. An investigation into neural net optimization via hessian eigenvalue density. In *International Conference on Machine Learning*, pp. 2232–2241. PMLR, 2019.

Gur-Ari, G., Roberts, D. A., and Dyer, E. Gradient descent happens in a tiny subspace. *arXiv preprint arXiv:1812.04754*, 2018.

Hu, W., Li, C. J., Li, L., and Liu, J.-G. On the diffusion approximation of nonconvex stochastic gradient descent. *arXiv preprint arXiv:1705.07562*, 2017.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.

LeCun, Y. and Bengio, Y. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 1998.

LeCun, Y., Bengio, Y., and Hinton, G. Deep learning. *nature*, 521(7553):436–444, 2015.

Li, C., Farkhoor, H., Liu, R., and Yosinski, J. Measuring the intrinsic dimension of objective landscapes. *arXiv preprint arXiv:1804.08838*, 2018.

Li, Q., Tai, C., et al. Stochastic modified equations and adaptive stochastic gradient algorithms. In *International Conference on Machine Learning*, pp. 2101–2110. PMLR, 2017.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Mandt, S., Hoffman, M. D., and Blei, D. M. Stochastic gradient descent as approximate bayesian inference. *Journal of Machine Learning Research*, 18(134):1–35, 2017.

Marsden, J. and Weinstein, A. Reduction of symplectic manifolds with symmetry. *Reports on mathematical physics*, 5(1):121–130, 1974.

Merity, S., Xiong, C., Bradbury, J., and Socher, R. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016.

Nigmetov, A., Krishnapriyan, A., Sanderson, N., and Morozov, D. Topological regularization via persistence-sensitive optimization. *Computational Geometry*, 120: 102086, 2024.

Power, A., Burda, Y., Edwards, H., Babuschkin, I., and Misra, V. Grokking: Generalization beyond overfitting on small algorithmic datasets. *arXiv preprint arXiv:2201.02177*, 2022.

Raginsky, M., Rakhlin, A., and Telgarsky, M. Non-convex learning via stochastic gradient langevin dynamics: a nonasymptotic analysis. In *Conference on Learning Theory*, pp. 1674–1703. PMLR, 2017.

Robbins, H. and Monro, S. A stochastic approximation method. *The annals of mathematical statistics*, pp. 400–407, 1951.

Sagun, L., Evci, U., Guney, V. U., Dauphin, Y., and Bottou, L. Empirical analysis of the hessian of over-parametrized neural networks. *arXiv preprint arXiv:1706.04454*, 2017.

Shumaylov, Z., Tsiaras, V., and Stylianou, Y. On information geometry and iterative optimization in model compression: Operator factorization. *arXiv preprint arXiv:2507.09428*, 2025.

Tolstikhin, I. O., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., Yung, J., Steiner, A., Keysers, D., Uszkoreit, J., et al. Mlp-mixer: An all-mlp architecture for vision. *Advances in neural information processing systems*, 34:24261–24272, 2021.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Xie, T., Geniesse, C., Chen, J., Yang, Y., Morozov, D., Mahoney, M. W., Maciejewski, R., and Weber, G. H. Evaluating loss landscapes from a topology perspective. *arXiv preprint arXiv:2411.09807*, 2024.

Zhang, Y., Yao, J., Wang, Y., and Chen, C. On the convergence of optimizing persistent-homology-based losses. *arXiv preprint arXiv:2206.02946*, 2022.

A. Theoretical Framework

A.1. Geometric Setup

We model deep learning training as a stochastic dynamical system driven jointly by drift and diffusion (LeCun et al., 2015). Rather than viewing training as a purely local response to the current gradient, we treat it as a controlled stochastic flow evolving under a geometry induced by recent trajectory history. To capture this local geometry, we define the empirical gradient covariance over a window of recent gradients,

$$G_t = \frac{1}{m}\sum_{i=1}^{m} g_{t-i} g_{t-i}^{\top}, \tag{17}$$

and let

$$G_t = U_t \Lambda_t U_t^{\top} \tag{18}$$

be its eigendecomposition. The leading eigenspace of G_t represents the dominant update channel supported by recent gradient history. We denote by $S_t \subset \mathbb{R}^D$ the corresponding low-dimensional local information subspace, and by $Q_t : \mathbb{R}^D \to \mathbb{R}^D$ the orthogonal projector onto S_t. Intuitively, S_t contains the directions that are stable and repeatedly reinforced by the recent trajectory, while its orthogonal complement Q_t captures transient perturbations, off-subspace drift, and noise-dominated variation.

To evaluate update quality, we replace Euclidean parameter-space metrics with the local Fisher geometry induced by the model output distribution (Appendix B.2). Under this geometry, we analyze a controlled update that preserves components aligned with the local information subspace while contracting orthogonal directions. In the original coordinates, the update is given by

$$\hat{g}_t = P_t g_t + \lambda_t (I - P_t) g_t, \qquad 0 \le \lambda_t \le 1, \tag{19}$$

where P_t is the empirical low-rank projector estimated from the recent gradient window. In whitened coordinates, the corresponding oracle form is

$$\tilde{u}_{t,\lambda} = Q_t \tilde{g}_t + \lambda (I - Q_t) \tilde{g}_t. \tag{20}$$

Thus, the tangential component is preserved while the orthogonal component is anisotropically damped. The special case $\lambda = 1$ recovers the raw stochastic gradient, whereas $\lambda = 0$ removes all orthogonal fluctuations. Finally, we make the following assumptions.

(A1) **Low-dimensional information subspace.** There exists a low-dimensional subspace $S_t \subseteq \mathbb{R}^D$ with $\dim(S_t) \ll D$ such that the ideal whitened update direction lies in S_t, i.e.,

$$Q_t u_t^* = u_t^*. \tag{21}$$

(A2) **Bounded noise covariance.** The conditional covariance of the whitened noise is uniformly bounded; equivalently, if

$$\tilde{g}_t = u_t^* + \tilde{\xi}_t, \qquad \mathbb{E}[\tilde{\xi}_t \mid \mathcal{F}_t] = 0, \tag{22}$$

then

$$\tilde{\Sigma}_t := \mathrm{Cov}(\tilde{\xi}_t \mid \mathcal{F}_t) \tag{23}$$

remains bounded.

(A3) **Local smoothness.** The local objective in whitened coordinates is β_t-smooth in a neighborhood of the current iterate.

(A4) **Controllable subspace estimation error.** The empirical projector $\hat{Q}_t$ constructed from the principal directions of G_t satisfies

$$\|\hat{Q}_t - Q_t\|_2 \le \delta_t, \tag{24}$$

where δ_t is sufficiently small and decreases as the historical estimate becomes more accurate.

A.2. Main Theorems

Building on the geometric setup, we study the controlled update, showing improved local information-geometric accuracy, sharper one-step descent, and stability under finite-window subspace estimation. We then present three main results, with all quantities in the Fisher-whitened space unless noted otherwise. Let

$$\tilde{g}_t = u_t^* + \tilde{\xi}_t, \quad \text{where} \quad u_t^* := \mathbb{E}[\tilde{g}_t \mid \mathcal{F}_t], \\ \mathbb{E}[\tilde{\xi}_t \mid \mathcal{F}_t] = 0, \quad \mathrm{Cov}(\tilde{\xi}_t \mid \mathcal{F}_t) = \tilde{\Sigma}_t. \tag{25}$$

and let Q_t denote the oracle projector onto the local information subspace. The corresponding oracle controlled update is

$$\tilde{u}_{t,\lambda} = Q_t \tilde{g}_t + \lambda (I - Q_t)\tilde{g}_t, \qquad 0 \le \lambda \le 1. \tag{26}$$

Theorem A.1 (Information-geometric accuracy improvement)**.** *Suppose Assumption (A1) holds, i.e., $Q_t u_t^* = u_t^*$. Then the conditional mean-squared error of the controlled update satisfies*

$$\mathbb{E}\big[\|\tilde{u}_{t,\lambda} - u_t^*\|_2^2 \,\big|\, \mathcal{F}_t\big] = \mathrm{tr}(Q_t \tilde{\Sigma}_t) + \lambda^2 \,\mathrm{tr}\big((I - Q_t)\tilde{\Sigma}_t\big). \tag{27}$$

For the raw whitened gradient (corresponding to $\lambda = 1$),

$$\mathbb{E}\big[\|\tilde{g}_t - u_t^*\|_2^2 \,\big|\, \mathcal{F}_t\big] = \mathrm{tr}(\tilde{\Sigma}_t). \tag{28}$$

Hence the improvement is

$$\Delta_t(\lambda) = (1 - \lambda^2)\,\mathrm{tr}\big((I - Q_t)\tilde{\Sigma}_t\big), \tag{29}$$

which is always nonnegative, and is strictly positive whenever $\lambda < 1$ and the orthogonal noise variance is nonzero.

Theorem A.1 shows that the proposed control law does more than improve empirical stability: under the local Fisher geometry, it strictly reduces local function-space error in conditional mean-square sense by removing orthogonal noise energy. The proof is given in Appendix E.1.

Theorem A.2 (Improved one-step descent bound)**.** *Let $\bar{L}_t(z)$ denote the local objective in Fisher-whitened coordinates, and suppose that $\bar{L}_t$ is β_t-smooth in a neighborhood of the current iterate z, i.e.,*

$$\bar{L}_t(z - v) \le \bar{L}_t(z) - \langle \nabla \bar{L}_t(z),\, v\rangle + \frac{\beta_t}{2}\|v\|_2^2. \tag{30}$$

Assume further that the true whitened gradient lies in the local information subspace:

$$Q_t \nabla \bar{L}_t(z) = \nabla \bar{L}_t(z). \tag{31}$$

Then, for any stepsize $\eta > 0$,

$$\begin{aligned}\mathbb{E}[\bar{L}_t(z - \eta \tilde{u}_{t,\lambda}) \mid \mathcal{F}_t] \le \bar{L}_t(z) - \eta \|\nabla \bar{L}_t(z)\|_2^2 \\ + \frac{\beta_t \eta^2}{2}\Big(\|\nabla \bar{L}_t(z)\|_2^2 + \mathrm{tr}(Q_t \tilde{\Sigma}_t) \\ + \lambda^2 \,\mathrm{tr}\big((I - Q_t)\tilde{\Sigma}_t\big)\Big).\end{aligned} \tag{32}$$

By contrast, the corresponding upper bound for the raw stochastic gradient is

$$\begin{aligned}&\mathbb{E}\big[\bar{L}_t(z - \eta \tilde{g}_t) \,\big|\, \mathcal{F}_t\big] \\ &\quad \le \bar{L}_t(z) - \eta \|\nabla \bar{L}_t(z)\|_2^2 + \frac{\beta_t \eta^2}{2}\Big(\|\nabla \bar{L}_t(z)\|_2^2 + \mathrm{tr}(\tilde{\Sigma}_t)\Big).\end{aligned} \tag{33}$$

Therefore, the controlled update improves the upper bound by

$$\frac{\beta_t \eta^2}{2}(1 - \lambda^2)\,\mathrm{tr}\big((I - Q_t)\tilde{\Sigma}_t\big), \tag{34}$$

which is nonnegative for all $\lambda \in [0, 1]$ and strictly positive whenever $\lambda < 1$ and the orthogonal noise variance is nonzero.

Theorem A.2 shows that the geometric accuracy gain is not merely statistical: under local smoothness, it translates directly into a tighter one-step descent guarantee. The tangential component preserves the first-order descent signal, while orthogonal noise enters mainly through the quadratic penalty term. The proof is deferred to Appendix E.2.

Theorem A.3 (Robustness to subspace estimation error)**.** *Suppose the empirical projector $\hat{Q}_t$ satisfies*

$$\|\hat{Q}_t - Q_t\|_2 \le \delta_t. \tag{35}$$

Define the practical controlled update by

$$\hat{u}_{t,\lambda} = \hat{Q}_t \tilde{g}_t + \lambda(I - \hat{Q}_t)\tilde{g}_t, \qquad 0 \le \lambda \le 1. \tag{36}$$

Then the deviation between the practical update and the oracle controlled update obeys

$$\|\hat{u}_{t,\lambda} - \tilde{u}_{t,\lambda}\|_2 \le (1-\lambda)\delta_t \, \|\tilde{g}_t\|_2. \tag{37}$$

Moreover,

$$\begin{aligned} \mathbb{E}_t[\|\hat{u}_{t,\lambda} - u_t^*\|_2^2] \le{}& 2\mathbb{E}_t[\|\tilde{u}_{t,\lambda} - u_t^*\|_2^2] \\ &+ 2(1-\lambda)^2 \delta_t^2 \mathbb{E}_t[\|\tilde{g}_t\|_2^2]. \end{aligned} \tag{38}$$

Hence the use of the estimated projector introduces only an additional perturbation of order

$$O\big((1-\lambda)^2 \delta_t^2\big), \tag{39}$$

so the oracle-level Fisher-geometric advantage remains stable whenever the subspace estimation error is sufficiently small.

Theorem A.3 connects the oracle analysis to the practical algorithm: finite-window estimation does not change the mechanism established in Theorems A.1 and A.2, but only adds a controlled second-order perturbation. The proof is provided in Appendix E.3.

A.3. Interpretation and Extensions

Theorems A.1–A.3 establish the main oracle-to-practice guarantee of the proposed control law. We now extend this picture in three directions: subspace mismatch induces a bias–variance tradeoff, this tradeoff motivates adaptive normal suppression, and topological feedback provides a practical proxy for the resulting control rule.

Proposition A.4 (Bias–variance tradeoff under subspace mismatch)**.** *Suppose the oracle direction admits the decomposition*

$$u_t^* = u_{t,\parallel}^* + u_{t,\perp}^*, \qquad u_{t,\parallel}^* = Q_t u_t^*, \qquad u_{t,\perp}^* = (I - Q_t)u_t^*. \tag{40}$$

Then, for the controlled update

$$\tilde{u}_{t,\lambda} = Q_t \tilde{g}_t + \lambda(I - Q_t)\tilde{g}_t, \qquad 0 \le \lambda \le 1, \tag{41}$$

the conditional mean-squared error satisfies

$$\begin{aligned} \mathbb{E}[\|\tilde{u}_{t,\lambda} - u_t^*\|_2^2 \mid \mathcal{F}_t] ={}& \mathrm{tr}(Q_t \tilde{\Sigma}_t) \\ &+ \lambda^2 \mathrm{tr}\big((I - Q_t)\tilde{\Sigma}_t\big) + (1-\lambda)^2 \|u_{t,\perp}^*\|_2^2. \end{aligned} \tag{42}$$

Consequently, the oracle-optimal shrinkage coefficient is

$$\lambda_t^* = \frac{\|u_{t,\perp}^*\|_2^2}{\mathrm{tr}\big((I - Q_t)\tilde{\Sigma}_t\big) + \|u_{t,\perp}^*\|_2^2}. \tag{43}$$

Proposition A.4 shows that normal suppression is fundamentally a bias–variance tradeoff: reducing λ suppresses orthogonal noise more strongly, but may also remove useful off-subspace signal. The optimal amount of shrinkage therefore depends on the local ratio between orthogonal signal and orthogonal noise, rather than remaining fixed throughout training.

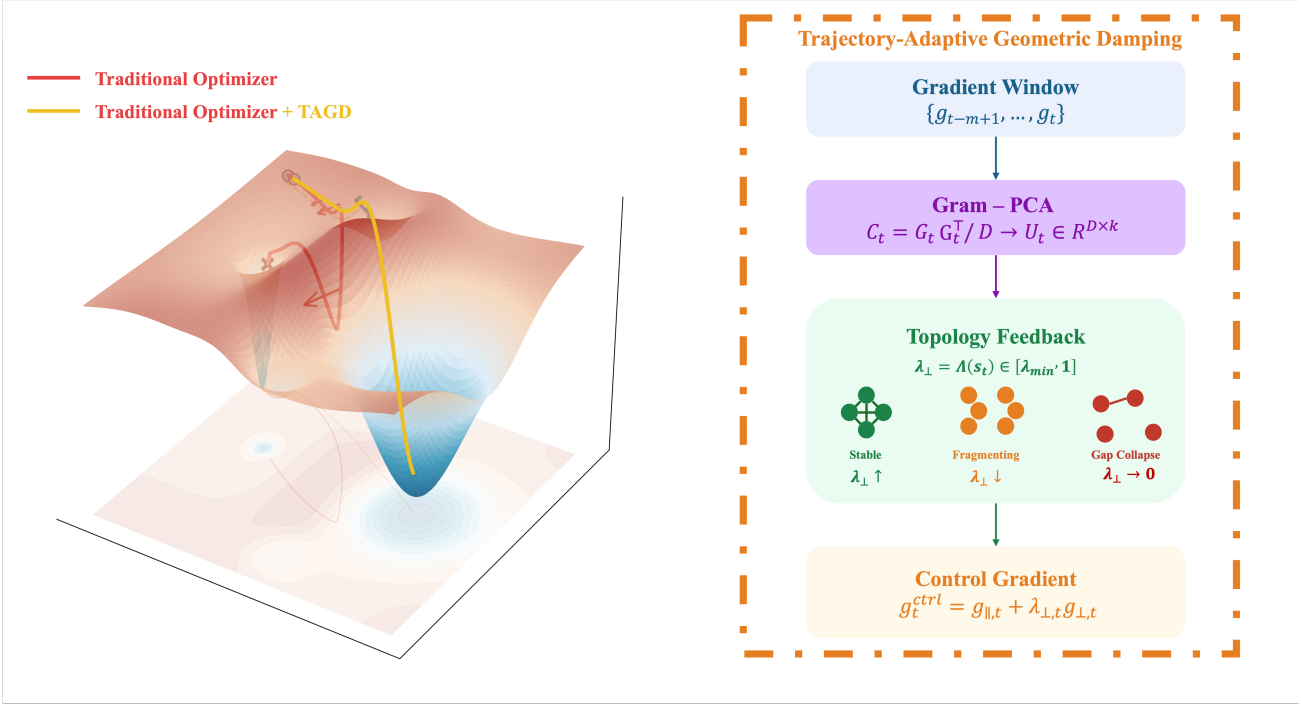

Figure 2. **TAGD as trajectory-level geometry–topology control.** Conventional optimization follows local gradients and can drift across unstable directions before entering a low-loss basin (*left*). TAGD instead treats training as controlled trajectory evolution: a recent gradient window defines a low-rank dominant subspace via Gram–PCA, and topological stability of the projected trajectory provides an online signal for adaptive normal suppression (*right*). The resulting update follows low-loss channels more coherently by preserving dominant descent directions while damping orthogonal instability.

Topology as an online proxy. The difficulty is that the orthogonal signal term $u^*_{t,\perp}$ is not directly observable, so the oracle-optimal coefficient above cannot be implemented literally. This is where topological feedback enters the framework. Our idea is to use the projected trajectory in the low-rank space as an online proxy for the latent orthogonal signal-to-noise balance. When the projected trajectory exhibits weakened connectivity, collapsed spectral gaps, or splitting of dominant direction clusters, the local structure is likely unstable; in such regimes, the orthogonal component is more likely to reflect cross-channel drift or unstable transitions, and $\lambda_\perp$ should be reduced. When the projected trajectory remains stable across scales, orthogonal exploration is more likely to contain useful signal, and a larger $\lambda_\perp$ is justified. In this sense, topology is not merely diagnostic: it provides a computable feedback signal for approximating the latent tradeoff in Proposition A.4.

Dynamical interpretation. The control law also admits a useful dynamical interpretation. Consider the continuous controlled flow

$$\dot{x}_t = -M_t \nabla \phi(x_t), \qquad M_t = Q_t + \lambda(I - Q_t). \tag{44}$$

A perturbation v_t along a reference trajectory evolves according to

$$\dot{v}_t = -M_t \nabla^2 \phi(x_t) v_t. \tag{45}$$

If the Hessian is approximately block-diagonal with respect to $S_t \oplus S_t^\perp$, then the orthogonal block is effectively rescaled from $H_{\perp,t}$ to $\lambda H_{\perp,t}$. Thus, when the orthogonal directions coincide with high-curvature, noise-sensitive, or structurally unstable modes, taking $\lambda < 1$ attenuates their amplification and reduces cross-channel drift. This viewpoint complements, rather than underpins, the main proofs. Although the framework is conceptually inspired by geometric mechanics, our analysis does not rely on a strict symplectic or Hamiltonian formalism; stochastic gradient training remains a noisy dissipative system.

B. Supplementary Concepts and Assumptions

B.1. Low-Dimensional Structure and Geometric Reduction in Optimization

Considering the optimization trajectory concentrates near a low-dimensional embedded manifold $\mathcal{M} \subset \mathbb{R}^D$, then its local motion is more naturally described by the tangent space $T_\theta\mathcal{M}$. Let P_θ denote the orthogonal projector onto $T_\theta\mathcal{M}$. Then any vector $g \in \mathbb{R}^D$ admits the decomposition

$$g = P_\theta g + (I - P_\theta)g =: g_\parallel + g_\perp, \tag{46}$$

where $g_\parallel$ is the tangential component and $g_\perp$ is the normal component. This notation provides a convenient language for distinguishing effective directions from off-manifold deviations. Such a viewpoint is motivated by empirical observations that deep-network training trajectories often evolve in a much lower-dimensional effective subspace than the ambient parameter dimension (Li et al., 2018; Gur-Ari et al., 2018).

To place this low-dimensional viewpoint in a broader geometric context, recall the formulation of classical Hamiltonian systems. Let $(\mathcal{P}, \omega)$ be a symplectic manifold, where ω is a closed and non-degenerate 2-form, i.e.,

$$d\omega = 0, \qquad \omega \text{ is non-degenerate.} \tag{47}$$

Given a Hamiltonian function $H : \mathcal{P} \to \mathbb{R}$, the associated Hamiltonian vector field X_H is defined by

$$\iota_{X_H}\omega = dH. \tag{48}$$

In local canonical coordinates (q, p), the dynamics are given by Hamilton's equations

$$\dot{q}_i = \frac{\partial H}{\partial p_i}, \qquad \dot{p}_i = -\frac{\partial H}{\partial q_i}. \tag{49}$$

When the system admits symmetry, one may reduce its degrees of freedom through symplectic reduction. Specifically, if a Lie group G acts on $(\mathcal{P}, \omega)$ in a symplectic manner and admits a momentum map $J : \mathcal{P} \to \mathfrak{g}^*$, then under suitable regularity conditions, the Marsden–Weinstein theorem states that for any regular value $\mu \in \mathfrak{g}^*$, the reduced space

$$\mathcal{P}_\mu = J^{-1}(\mu)/G_\mu \tag{50}$$

is again a symplectic manifold (Marsden & Weinstein, 1974). In other words, symmetry allows a high-dimensional dynamical system to be compressed onto a lower-dimensional space while preserving its essential dynamical structure.

B.2. Fisher geometry

Let

$$F_t := \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim p(y|x,\theta_t)}\left[\nabla_\theta \log p(y \mid x, \theta_t)\, \nabla_\theta \log p(y \mid x, \theta_t)^\top\right] \tag{51}$$

be the local Fisher information matrix at θ_t. Under standard regularity conditions, for any sufficiently small perturbation $\delta\theta$,

$$\mathrm{KL}(p_{\theta_t} \,\|\, p_{\theta_t+\delta\theta}) = \frac{1}{2}\,\delta\theta^\top F_t \delta\theta + o(\|\delta\theta\|^2), \tag{52}$$

so the Fisher norm

$$\|\delta\theta\|_{F_t}^2 := \delta\theta^\top F_t \delta\theta \tag{53}$$

measures local distortion in function space rather than raw parameter displacement. This is particularly important in deep networks, where many parameter directions are functionally redundant due to overparameterization. For analysis, we introduce Fisher-whitened coordinates

$$z = F_t^{1/2}\theta, \qquad \tilde{g}_t = F_t^{-1/2} g_t, \tag{54}$$

where $F_t^{1/2}$ and its inverse are used only in theory and are never computed explicitly in practice. In this whitened space, Fisher-norm comparisons reduce to Euclidean ones. We define the ideal whitened update direction by

$$u_t^* := \mathbb{E}[\tilde{g}_t \mid \mathcal{F}_t], \tag{55}$$

where $\mathcal{F}_t$ denotes the filtration generated by the optimization history up to time t.

Algorithm 1 Geometry–Topology Trajectory Control

Input: base optimizer $\mathcal{O}$, history length m, subspace rank k, update interval p, scale set $\mathcal{T}$, schedule parameters
Initialize history buffer $\mathcal{B} \leftarrow \emptyset$
for each training iteration t **do**
  Compute stochastic gradient g_t
  Append g_t to $\mathcal{B}$ and keep the most recent m gradients
  if the history window is not yet full **then**
    Use g_t as the consumed gradient and apply $\mathcal{O}$
    continue
  end if
  if t is a subspace-update step **then**
    Form the centered trajectory matrix $\mathbf{G}_t$
    Construct $C_t = \frac{1}{D}\mathbf{G}_t\mathbf{G}_t^\top$
    Compute the top-k eigendirections and obtain U_t
    Set $P_t = U_t U_t^\top$
  end if
  Project the trajectory: $Z_t = (\mathbf{G}_t - \bar{\mathbf{G}}_t)U_t$
  for each scale $\tau \in \mathcal{T}$ **do**
    Build the similarity graph $A^{(\tau)}$
    Compute graph-based stability descriptors
  end for
  Aggregate descriptors into an instability score s_t
  Compute $\lambda_{\perp,t} = \Lambda(s_t)$
  Compute

$$g_{\parallel,t} = P_t g_t, \qquad g_{\perp,t} = (I - P_t)g_t$$

  Form

$$g_t^{\mathrm{ctrl}} = g_{\parallel,t} + \lambda_{\perp,t} g_{\perp,t}$$

  Compute the intervention weight ω_t
  Form

$$g_t^{\mathrm{used}} = (1-\omega_t)g_t + \omega_t g_t^{\mathrm{ctrl}}$$

  Replace the optimizer input by g_t^{used} and apply $\mathcal{O}$
end for

C. Algorithm Pseudocode

D. Experiment Setup

D.1. Architectures Evaluated

Small-scale vision models on CIFAR-10. We evaluate TAGD on three compact vision architectures with distinct inductive biases and optimization geometries.

SmallCNN (∼340K parameters).(LeCun & Bengio, 1998) SmallCNN is a lightweight convolutional network composed of three sequential convolutional blocks, each following a CONV → BN → RELU → MAXPOOL pattern, with channel widths increasing from $3 \to 32 \to 64 \to 128$. The resulting 4×4 feature map is passed to a two-layer fully connected classification head. Since the architecture contains no residual connections, its optimization dynamics are less buffered by shortcut pathways and therefore provide a direct testbed for examining the effect of TAGD under genuinely curved loss geometry.

PatchMLP (∼1.1M parameters).(Tolstikhin et al., 2021) PatchMLP partitions each image into 64 non-overlapping 4×4 patches, projects them linearly into a latent embedding space, and processes them with four MLP-Mixer-style blocks. Each block alternates between a token-mixing MLP operating across patches and a channel-mixing MLP operating across feature

dimensions, followed by global average pooling and a linear classifier. In contrast to convolutional models, PatchMLP does not impose spatial weight sharing, which typically yields a broader effective gradient subspace. As a result, TAGD generally requires a larger subspace rank k to capture its local update geometry.

MicroTransformer ($\sim$870K parameters).(Dosovitskiy et al., 2020) MicroTransformer first applies a convolutional patch embedding layer with kernel size 4×4 and stride 4, then prepends a `[CLS]` token and adds sinusoidal positional encodings. The resulting token sequence is processed by a four-layer standard `TransformerEncoder` with multi-head self-attention, feed-forward networks, and pre-normalization. The final prediction is obtained by feeding the `[CLS]` representation through batch normalization and a linear classification layer. In early training, the attention maps of this model are often close to uniform, which makes the gradient field approximately isotropic. In this regime, the topological signals used by TAGD, such as β_0 and the spectral gap, are particularly interpretable.

Language models. We further consider autoregressive language models spanning multiple parameter scales and architectural organizations.

LLaMA-2-style models (100M / 350M parameters).(Touvron et al., 2023) These models follow a LLaMA-2-style decoder-only Transformer design with rotary positional embeddings (RoPE), bias-free RMSNorm, and SwiGLU feed-forward networks. RoPE injects positional structure into the query and key representations through parameter-free rotations, while the SwiGLU block uses three projection matrices, W_{gate}, W_{up}, and W_{down}, whose gated interaction introduces nontrivial cross-coupling in the gradient flow. The bias-free design typically yields a cleaner gradient covariance structure, whereas RoPE induces position-sensitive directional variation in gradients. Together, these properties make this family a challenging setting for TAGD, especially for the Gram-PCA subspace estimation step.

GPT-J-style models (70M / 125M / 200M parameters).(Black et al., 2022) The most important difference between GPT-J-style models and more standard sequential Transformer variants lies in the block structure. Instead of applying attention and MLP sublayers serially, GPT-J adopts a parallel residual form,

$$x = x + \mathrm{Attn}(\mathrm{Norm}(x)) + \mathrm{MLP}(\mathrm{Norm}(x)). \tag{56}$$

That is, the attention branch and the MLP branch share the same normalized upstream representation, and their gradient contributions are added in parallel rather than propagated through a strictly sequential pipeline. This architectural choice tends to reduce the effective rank of the gradient buffer relative to sequential designs, while simultaneously concentrating gradient directions more strongly. Empirically, this often leads to a more stable spectral gap, which is favorable for TAGD's topology-aware control mechanism.

D.2. Computational Overhead

TAGD introduces only modest overhead beyond the base optimizer. Its additional cost has three sources: maintaining a short gradient buffer, periodically estimating a low-rank update subspace via Gram-PCA, and reshaping the current gradient using the estimated basis. Among these, buffer maintenance is negligible, requiring only an $O(D)$ write per step for a parameter vector of dimension D. The main extra computation arises from subspace estimation. Let $G \in \mathbb{R}^{m \times D}$ denote the buffer of the most recent m gradients. Instead of forming and diagonalizing a $D \times D$ covariance matrix, TAGD performs PCA in Gram space by constructing

$$C = GG^\top / D \in \mathbb{R}^{m \times m}, \tag{57}$$

followed by eigendecomposition in the sample dimension. This yields a dominant complexity of $O(m^2 D)$, with the $O(m^3)$ eigendecomposition term negligible in practice since $m \ll D$. After the basis is obtained, each training step only requires two matrix-vector products for gradient decomposition and reconstruction, resulting in an additional $O(Dk)$ cost, where k is the retained subspace rank.

In practice, this design keeps the overhead small even for medium-scale language models. Across the architectures considered in this work, the amortized compute overhead is approximately 5%–13% of a standard training step, depending on model size and the refresh interval of Gram-PCA. The key reason is that subspace estimation is performed only once every several optimization steps, so the $O(m^2 D)$ cost is effectively amortized over time. Importantly, TAGD avoids any quadratic dependence on the parameter dimension in the PCA stage, which would be prohibitive for modern networks.

The memory overhead is also simple and predictable: it scales linearly with the buffer size as $O(mD)$, corresponding to storing m past gradients. Overall, TAGD adds a moderate and controllable cost while remaining substantially cheaper than redesigning the optimizer around second-order or full-history curvature estimation.

D.3. Loss Landscape Visualization

To visualize and compare the optimization geometry induced by the baseline optimizer and TAGD, we construct a shared two-dimensional loss landscape from the recorded training trajectories. Direct visualization in the original parameter space is infeasible because the model parameters lie in a very high-dimensional space. Our strategy is therefore to first identify a meaningful global two-dimensional coordinate system from the optimization trajectories, and then to evaluate the loss surface on this coordinate system using a local geometric approximation. This procedure produces a shared landscape on which the two trajectories can be overlaid and compared directly.

Step 1: Global PCA plane. We first collect parameter snapshots from training. Let the recorded trajectory be

$$\theta_0, \theta_1, \ldots, \theta_n, \tag{58}$$

where each snapshot is stored once every fixed number of optimization steps. We then merge the snapshots from the baseline run and the TAGD run into a single trajectory set,

$$\mathcal{U} = \left[\theta_0^{\text{base}}, \ldots, \theta_{n_b}^{\text{base}}, \theta_0^{\text{tagd}}, \ldots, \theta_{n_t}^{\text{tagd}}\right] \in \mathbb{R}^{N \times D}, \tag{59}$$

where D denotes the parameter dimension and N is the total number of stored snapshots from both runs. Rather than centering the visualization at the final model, we use the empirical mean of the union trajectory as the reference point,

$$\theta_{\text{ref}} = \frac{1}{N} \sum_{i=1}^{N} \mathcal{U}_i. \tag{60}$$

This choice yields a symmetric global coordinate system that reflects the joint geometry of the two trajectories instead of privileging a single endpoint.

After centering, we form the matrix

$$X = \mathcal{U} - \mathbf{1}\theta_{\text{ref}}^{\top}, \tag{61}$$

and compute its singular value decomposition,

$$X = P \Sigma Q^{\top}. \tag{62}$$

The first two principal directions are then taken as the dominant right singular vectors,

$$d_1 = Q_{:,1}, \qquad d_2 = Q_{:,2}. \tag{63}$$

These two directions define a global two-dimensional plane shared by the baseline and TAGD trajectories. Any parameter vector can be projected into this plane by

$$\alpha(\theta) = \langle \theta - \theta_{\text{ref}}, d_1 \rangle, \qquad \beta(\theta) = \langle \theta - \theta_{\text{ref}}, d_2 \rangle. \tag{64}$$

To make the resulting plots visually consistent and easier to interpret, we apply a sign correction to the principal directions. Let the net displacement of the merged trajectory be

$$\Delta = \theta_{\text{final}} - \theta_{\text{init}}. \tag{65}$$

If a principal direction is negatively aligned with this displacement, we reverse its sign:

$$d_j \leftarrow -d_j \quad \text{if } \langle d_j, \Delta \rangle < 0, \qquad j \in \{1, 2\}. \tag{66}$$

This convention ensures that the trajectories move approximately from the lower-left region toward the upper-right region of the plot, which improves interpretability without changing the underlying geometry.

Step 2: Local-plane blended loss surface. A uniform loss evaluation directly on the global PCA plane can be inaccurate when the queried point lies far from the actual training trajectory, because the global PCA plane only provides a coarse linear approximation to a highly curved optimization path. To reduce this distortion, we construct the surface using a local-plane blending procedure.

We first choose a set of anchor points uniformly along the merged trajectory. Around each anchor point, we define a local tangent plane using discrete velocity and acceleration estimates. For an anchor centered at time index t, we define the local velocity as

$$v_t = \theta_t - \theta_{t-k}, \tag{67}$$

and the local acceleration as

$$a_t = v_t - (\theta_{t-k} - \theta_{t-2k}). \tag{68}$$

The first basis vector of the local plane is the normalized velocity direction,

$$e_{1,t} = \frac{v_t}{\|v_t\|}. \tag{69}$$

The second basis vector is obtained by removing the component of the acceleration along the velocity direction and then normalizing:

$$\widetilde{e}_{2,t} = a_t - \langle a_t, e_{1,t} \rangle e_{1,t}, \tag{70}$$

$$e_{2,t} = \frac{\widetilde{e}_{2,t}}{\|\widetilde{e}_{2,t}\|}. \tag{71}$$

This is a Gram–Schmidt orthogonalization step, and it yields a local two-dimensional basis adapted to the instantaneous geometry of the trajectory. Each anchor therefore defines a local tangent approximation consisting of a center point and two orthonormal directions,

$$\Pi_t = (\theta_t, e_{1,t}, e_{2,t}). \tag{72}$$

Next, we evaluate the loss on a regular grid in the global PCA coordinate system. Let a query location on the visualization grid be denoted by

$$q = (u, v). \tag{73}$$

This point is first mapped back into parameter space using the global PCA basis,

$$w_{\text{query}} = \theta_{\text{ref}} + u d_1 + v d_2. \tag{74}$$

For each local plane Π_t, we compute the local coordinates of this query point by projection,

$$\alpha_t = \langle w_{\text{query}} - \theta_t, e_{1,t} \rangle, \qquad \beta_t = \langle w_{\text{query}} - \theta_t, e_{2,t} \rangle. \tag{75}$$

The corresponding locally evaluated point is then

$$w_{\text{eval}}^{(t)} = \theta_t + \alpha_t e_{1,t} + \beta_t e_{2,t}. \tag{76}$$

Intuitively, this is the nearest point to the query on the local tangent approximation around anchor t.

To blend local evaluations into a smooth global surface, we assign each anchor a Gaussian weight based on the distance between the query point and the anchor center:

$$g_t(q) = \exp\left(-\frac{\|q - c_t\|^2}{2\sigma^2}\right), \tag{77}$$

where c_t denotes the projection of anchor θ_t into the global PCA plane, and σ controls the locality of the blending. Let $L(\cdot)$ denote the empirical loss evaluated by loading the parameter vector into the model and running forward inference on a small number of mini-batches. The final blended surface value at query point q is given by

$$Z(q) = \frac{\sum_t g_t(q)\, L\left(w_{\text{eval}}^{(t)}\right)}{\sum_t g_t(q)}. \tag{78}$$

This blended construction yields a surface that remains faithful to the local geometry of the training path while still providing a single smooth visualization over the shared global plane.

Step 3: Trajectory overlay. After the surface has been constructed, we project both the baseline and TAGD trajectories into the same PCA coordinate system and overlay them on top of the landscape. For any recorded checkpoint θ_s, its two-dimensional coordinates are

$$u_s = \langle \theta_s - \theta_{\mathrm{ref}}, d_1 \rangle \,, \qquad v_s = \langle \theta_s - \theta_{\mathrm{ref}}, d_2 \rangle \,. \tag{79}$$

To place the trajectory on the three-dimensional surface, we recover its height by interpolation from the sampled grid,

$$z_s = \mathrm{Interp}(Z, u_s, v_s) \,, \tag{80}$$

where Interp denotes bilinear interpolation on the evaluated grid. The resulting path

$$(u_s, v_s, z_s) \tag{81}$$

is then rendered on both the three-dimensional surface plot and the corresponding two-dimensional contour plot. In the figures, we mark the start of each trajectory with a filled circle and the end with a filled square, which makes it possible to compare not only the final destination of optimization but also the geometric route taken to reach it.

Evaluation details and hyperparameters. Unless otherwise specified, we use a square visualization grid of size 31×31, which corresponds to 961 query locations in the global PCA plane. From the merged trajectory, we select at most 20 local tangent planes, sampled approximately uniformly along the path. The anchor stride used to compute local finite-difference velocity and acceleration is 10 trajectory steps. The Gaussian blending bandwidth is set to $\sigma = 0.30$, measured in the coordinates of the PCA plane. To prevent extremely large losses near initialization from flattening the dynamic range of the plot, we clip the displayed loss values by

$$L_{\mathrm{disp}}(q) = \min\left(Z(q), 3.0\right). \tag{82}$$

For computational efficiency, each surface query is evaluated using at most 2 mini-batches. During training, we save one model snapshot every 200 optimization steps, which determines the temporal resolution of the recorded trajectory and therefore the fidelity of the visualization.

Overall, this procedure produces a shared, trajectory-aware loss landscape that is substantially more faithful than direct evaluation on a single global linear plane. The global PCA basis provides a common coordinate system for comparing optimizers, while the local-plane blending mechanism preserves the local curvature structure of the trajectory and reduces the extrapolation artifacts that arise when the loss is evaluated far from the optimization path.

D.4. TAGD Hyperparameters

Table 2 summarizes the main TAGD hyperparameters used across all experiments. We report both the optimization-related control parameters of TAGD and the corresponding training configurations for each model family.

Hyperparameter rationale. The subspace rank k is chosen according to the effective dimensionality of recent gradient trajectories. Sequential architectures such as CNNs and LLaMA-style models use $k = 16$, while GPT-J-style parallel blocks use a smaller value ($k = 12$), since the attention and MLP branches share the same upstream normalized signal and thus typically exhibit a lower effective gradient rank. The gradient buffer size m is set by trading off statistical stability and memory overhead: smaller vision models use larger buffers ($m = 16$–32), whereas larger language models use smaller buffers ($m = 8$–16) to keep the memory cost manageable. The orthogonal retention coefficient $\lambda_\perp$ is set lower for LLaMA-style models (0.10–0.15), where the gated SwiGLU feed-forward structure induces richer cross-coupled gradient components outside the dominant tangent subspace; in contrast, GPT-J uses a larger value (0.20–0.25), reflecting its more concentrated gradient geometry and the greater utility of preserving orthogonal exploratory directions. The intervention fraction is set to 1.0 for language modeling, where sustained intervention consistently improves optimization efficiency, while for CIFAR-10 it is limited to 0.5 so that the final stage of training can proceed with less geometric constraint. Finally, the PCA refresh interval is chosen according to the stability of the local gradient subspace: GPT-J uses the largest interval (`pca_every`=40) because its parallel block structure yields more stable dominant directions, LLaMA uses 30 due to somewhat faster directional variation induced by positional rotation, and CIFAR-10 models use 20–30 depending on architecture.

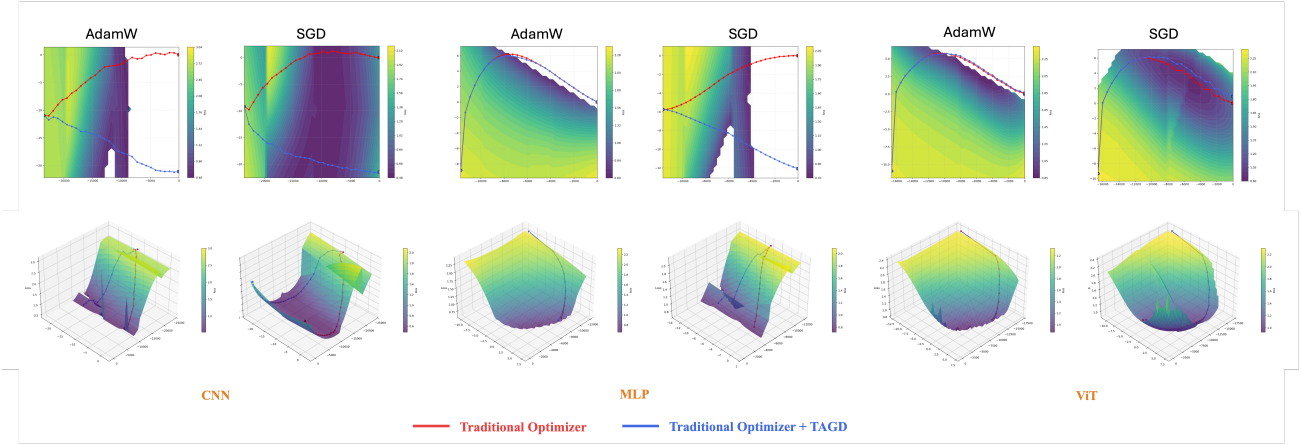

Figure 3. **Trajectory and local loss-landscape visualization on CIFAR-10.** Red denotes the baseline optimizer and blue denotes TAGD, shown for CNN, MLP, and ViT under AdamW and SGD. The top row gives 2D projected loss contours and the bottom row the corresponding 3D local landscapes. TAGD typically enters and follows low-loss channels more coherently, with reduced lateral drift relative to the baseline, especially under SGD.

D.5. More Result

E. Proof

E.1. Proof of Theorem A.1

Proof. Under Assumption (A1), the oracle controlled update in whitened coordinates is

$$\tilde{u}_{t,\lambda} = Q_t \tilde{g}_t + \lambda(I - Q_t)\tilde{g}_t. \tag{83}$$

Using the decomposition

$$\tilde{g}_t = u_t^* + \tilde{\xi}_t, \qquad \mathbb{E}[\tilde{\xi}_t \mid \mathcal{F}_t] = 0, \qquad \mathrm{Cov}(\tilde{\xi}_t \mid \mathcal{F}_t) = \tilde{\Sigma}_t, \tag{84}$$

together with the assumption

$$Q_t u_t^* = u_t^*, \tag{85}$$

we obtain

$$\tilde{u}_{t,\lambda} = Q_t(u_t^* + \tilde{\xi}_t) + \lambda(I - Q_t)(u_t^* + \tilde{\xi}_t) \tag{86}$$

$$= u_t^* + Q_t\tilde{\xi}_t + \lambda(I - Q_t)\tilde{\xi}_t. \tag{87}$$

Hence, the estimation error in whitened coordinates is

$$\tilde{u}_{t,\lambda} - u_t^* = Q_t\tilde{\xi}_t + \lambda(I - Q_t)\tilde{\xi}_t. \tag{88}$$

Taking the squared Euclidean norm and using the orthogonality relation

$$Q_t(I - Q_t) = 0, \tag{89}$$

we have

$$\|\tilde{u}_{t,\lambda} - u_t^*\|_2^2 = \|Q_t\tilde{\xi}_t\|_2^2 + \lambda^2\|(I - Q_t)\tilde{\xi}_t\|_2^2. \tag{90}$$

Now take conditional expectation given $\mathcal{F}_t$. Since Q_t is an orthogonal projector, it satisfies

$$Q_t^\top = Q_t, \qquad Q_t^2 = Q_t. \tag{91}$$

Therefore,

$$\mathbb{E}\Big[\|Q_t\tilde{\xi}_t\|_2^2 \,\Big|\, \mathcal{F}_t\Big] = \mathbb{E}\Big[\tilde{\xi}_t^\top Q_t^\top Q_t\tilde{\xi}_t \,\Big|\, \mathcal{F}_t\Big] \tag{92}$$

$$= \mathbb{E}\Big[\tilde{\xi}_t^\top Q_t\tilde{\xi}_t \,\Big|\, \mathcal{F}_t\Big] \tag{93}$$

$$= \operatorname{tr}\Big(Q_t\,\mathbb{E}[\tilde{\xi}_t\tilde{\xi}_t^\top \mid \mathcal{F}_t]\Big) \tag{94}$$

$$= \operatorname{tr}(Q_t\tilde{\Sigma}_t). \tag{95}$$

Similarly,

$$\mathbb{E}\Big[\|(I-Q_t)\tilde{\xi}_t\|_2^2 \,\Big|\, \mathcal{F}_t\Big] = \operatorname{tr}\Big((I-Q_t)\tilde{\Sigma}_t\Big). \tag{96}$$

Substituting the above identities yields

$$\mathbb{E}\big[\|\tilde{u}_{t,\lambda} - u_t^*\|_2^2 \,\big|\, \mathcal{F}_t\big] = \operatorname{tr}(Q_t\tilde{\Sigma}_t) + \lambda^2 \operatorname{tr}\big((I-Q_t)\tilde{\Sigma}_t\big). \tag{97}$$

For the raw whitened gradient, corresponding to $\lambda = 1$, we have

$$\mathbb{E}\big[\|\tilde{g}_t - u_t^*\|_2^2 \,\big|\, \mathcal{F}_t\big] = \operatorname{tr}(Q_t\tilde{\Sigma}_t) + \operatorname{tr}\big((I-Q_t)\tilde{\Sigma}_t\big) \tag{98}$$

$$= \operatorname{tr}(\tilde{\Sigma}_t). \tag{99}$$

Hence the improvement is

$$\Delta_t(\lambda) := \mathbb{E}\big[\|\tilde{g}_t - u_t^*\|_2^2 - \|\tilde{u}_{t,\lambda} - u_t^*\|_2^2 \,\big|\, \mathcal{F}_t\big] \tag{100}$$

$$= (1-\lambda^2)\operatorname{tr}\big((I-Q_t)\tilde{\Sigma}_t\big). \tag{101}$$

Since $\lambda \in [0,1]$, we have

$$1 - \lambda^2 \geq 0. \tag{102}$$

Moreover, because $(I-Q_t)\tilde{\Sigma}_t$ is positive semidefinite in trace,

$$\operatorname{tr}\big((I-Q_t)\tilde{\Sigma}_t\big) \geq 0. \tag{103}$$

Therefore,

$$\Delta_t(\lambda) \geq 0. \tag{104}$$

The inequality is strict whenever $\lambda < 1$ and

$$(I-Q_t)\tilde{\Sigma}_t \neq 0, \tag{105}$$

that is, whenever the orthogonal complement of S_t carries nonzero noise variance. This completes the proof. □

E.2. Proof of Theorem A.2

Proof. By the β_t-smoothness of $\bar{L}_t$, for any v in a local neighborhood of z,

$$\bar{L}_t(z-v) \leq \bar{L}_t(z) - \langle \nabla\bar{L}_t(z), v\rangle + \frac{\beta_t}{2}\|v\|_2^2. \tag{106}$$

Applying this inequality with $v = \eta\tilde{u}_{t,\lambda}$ gives

$$\bar{L}_t(z-\eta\tilde{u}_{t,\lambda}) \leq \bar{L}_t(z) - \eta\langle \nabla\bar{L}_t(z), \tilde{u}_{t,\lambda}\rangle + \frac{\beta_t\eta^2}{2}\|\tilde{u}_{t,\lambda}\|_2^2. \tag{107}$$

Taking conditional expectation with respect to $\mathcal{F}_t$, we analyze the linear and quadratic terms separately. Recall that

$$\tilde{g}_t = u_t^* + \tilde{\xi}_t, \qquad \mathbb{E}[\tilde{\xi}_t \mid \mathcal{F}_t] = 0, \qquad \operatorname{Cov}(\tilde{\xi}_t \mid \mathcal{F}_t) = \tilde{\Sigma}_t, \tag{108}$$

and that the controlled update is

$$\tilde{u}_{t,\lambda} = Q_t \tilde{g}_t + \lambda (I - Q_t)\tilde{g}_t. \tag{109}$$

Using Assumption (A1), namely $Q_t u_t^* = u_t^*$, we obtain

$$\tilde{u}_{t,\lambda} = Q_t(u_t^* + \tilde{\xi}_t) + \lambda(I - Q_t)(u_t^* + \tilde{\xi}_t) \tag{110}$$

$$= u_t^* + Q_t\tilde{\xi}_t + \lambda(I - Q_t)\tilde{\xi}_t. \tag{111}$$

We first consider the inner-product term. Since the conditional mean of the whitened gradient equals the true local whitened gradient, we have

$$u_t^* = \nabla \bar{L}_t(z). \tag{112}$$

Therefore,

$$\mathbb{E}\big[\langle \nabla \bar{L}_t(z), \tilde{u}_{t,\lambda}\rangle \,\big|\, \mathcal{F}_t\big] = \big\langle \nabla \bar{L}_t(z), u_t^* \big\rangle + \mathbb{E}\Big[\langle \nabla \bar{L}_t(z), Q_t\tilde{\xi}_t\rangle \,\Big|\, \mathcal{F}_t\Big] + \lambda\, \mathbb{E}\Big[\langle \nabla \bar{L}_t(z), (I - Q_t)\tilde{\xi}_t\rangle \,\Big|\, \mathcal{F}_t\Big]. \tag{113}$$

The first term is

$$\langle \nabla \bar{L}_t(z), u_t^*\rangle = \|\nabla \bar{L}_t(z)\|_2^2. \tag{114}$$

The second term vanishes because $\mathbb{E}[\tilde{\xi}_t \mid \mathcal{F}_t] = 0$. The third term also vanishes since

$$Q_t \nabla \bar{L}_t(z) = \nabla \bar{L}_t(z) \quad \Longrightarrow \quad (I - Q_t)\nabla \bar{L}_t(z) = 0. \tag{115}$$

Hence,

$$\mathbb{E}\big[\langle \nabla \bar{L}_t(z), \tilde{u}_{t,\lambda}\rangle \,\big|\, \mathcal{F}_t\big] = \|\nabla \bar{L}_t(z)\|_2^2. \tag{116}$$

We next analyze the quadratic term. Expanding the squared norm gives

$$\|\tilde{u}_{t,\lambda}\|_2^2 = \|u_t^*\|_2^2 + \|Q_t\tilde{\xi}_t\|_2^2 + \lambda^2\|(I - Q_t)\tilde{\xi}_t\|_2^2 + 2\langle u_t^*, Q_t\tilde{\xi}_t\rangle + 2\lambda\langle u_t^*, (I - Q_t)\tilde{\xi}_t\rangle. \tag{117}$$

Taking conditional expectation and using $\mathbb{E}[\tilde{\xi}_t \mid \mathcal{F}_t] = 0$, both cross terms vanish, so

$$\mathbb{E}\big[\|\tilde{u}_{t,\lambda}\|_2^2 \,\big|\, \mathcal{F}_t\big] = \|u_t^*\|_2^2 + \mathbb{E}\Big[\|Q_t\tilde{\xi}_t\|_2^2 \,\Big|\, \mathcal{F}_t\Big] + \lambda^2\mathbb{E}\Big[\|(I - Q_t)\tilde{\xi}_t\|_2^2 \,\Big|\, \mathcal{F}_t\Big]. \tag{118}$$

Since Q_t is an orthogonal projector,

$$Q_t^\top = Q_t, \qquad Q_t^2 = Q_t, \tag{119}$$

and therefore

$$\mathbb{E}\Big[\|Q_t\tilde{\xi}_t\|_2^2 \,\Big|\, \mathcal{F}_t\Big] = \mathbb{E}\Big[\tilde{\xi}_t^\top Q_t\tilde{\xi}_t \,\Big|\, \mathcal{F}_t\Big] = \mathrm{tr}(Q_t\tilde{\Sigma}_t), \tag{120}$$

$$\mathbb{E}\Big[\|(I - Q_t)\tilde{\xi}_t\|_2^2 \,\Big|\, \mathcal{F}_t\Big] = \mathrm{tr}\big((I - Q_t)\tilde{\Sigma}_t\big). \tag{121}$$

Using again $u_t^* = \nabla \bar{L}_t(z)$, we obtain

$$\mathbb{E}\big[\|\tilde{u}_{t,\lambda}\|_2^2 \,\big|\, \mathcal{F}_t\big] = \|\nabla \bar{L}_t(z)\|_2^2 + \mathrm{tr}(Q_t\tilde{\Sigma}_t) + \lambda^2\,\mathrm{tr}\big((I - Q_t)\tilde{\Sigma}_t\big). \tag{122}$$

Substituting the above bounds into the smoothness inequality yields

$$\mathbb{E}\big[\bar{L}_t(z - \eta\tilde{u}_{t,\lambda}) \,\big|\, \mathcal{F}_t\big] \le \bar{L}_t(z) - \eta\|\nabla \bar{L}_t(z)\|_2^2 + \frac{\beta_t\eta^2}{2}\Big(\|\nabla \bar{L}_t(z)\|_2^2 + \mathrm{tr}(Q_t\tilde{\Sigma}_t) + \lambda^2\,\mathrm{tr}\big((I - Q_t)\tilde{\Sigma}_t\big)\Big), \tag{123}$$

which proves the claimed one-step upper bound.

For the raw whitened gradient, corresponding to $\lambda = 1$, the same argument yields

$$\mathbb{E}\big[\bar{L}_t(z - \eta\tilde{g}_t) \,\big|\, \mathcal{F}_t\big] \le \bar{L}_t(z) - \eta\|\nabla\bar{L}_t(z)\|_2^2 + \frac{\beta_t\eta^2}{2}\Big(\|\nabla\bar{L}_t(z)\|_2^2 + \mathrm{tr}(\tilde{\Sigma}_t)\Big). \tag{124}$$

Subtracting the controlled bound from the raw bound gives

$$\frac{\beta_t\eta^2}{2}(1-\lambda^2)\,\mathrm{tr}\big((I - Q_t)\tilde{\Sigma}_t\big), \tag{125}$$

which is nonnegative for all $\lambda \in [0, 1]$, and strictly positive whenever $\lambda < 1$ and

$$\mathrm{tr}\big((I - Q_t)\tilde{\Sigma}_t\big) > 0. \tag{126}$$

This completes the proof. □

E.3. Proof of Theorem A.3

Proof. From the definitions of the oracle and empirical controlled updates,

$$\tilde{u}_{t,\lambda} = Q_t\tilde{g}_t + \lambda(I - Q_t)\tilde{g}_t = \lambda\tilde{g}_t + (1-\lambda)Q_t\tilde{g}_t, \tag{127}$$

$$\hat{u}_{t,\lambda} = \hat{Q}_t\tilde{g}_t + \lambda(I - \hat{Q}_t)\tilde{g}_t = \lambda\tilde{g}_t + (1-\lambda)\hat{Q}_t\tilde{g}_t. \tag{128}$$

Subtracting the two expressions yields

$$\hat{u}_{t,\lambda} - \tilde{u}_{t,\lambda} = (1-\lambda)(\hat{Q}_t - Q_t)\tilde{g}_t. \tag{129}$$

Therefore, by sub-multiplicativity of the operator norm,

$$\|\hat{u}_{t,\lambda} - \tilde{u}_{t,\lambda}\|_2 \le (1-\lambda)\|\hat{Q}_t - Q_t\|_2\,\|\tilde{g}_t\|_2 \le (1-\lambda)\delta_t\|\tilde{g}_t\|_2. \tag{130}$$

Next, decompose the empirical estimation error as

$$\hat{u}_{t,\lambda} - u_t^* = (\tilde{u}_{t,\lambda} - u_t^*) + (\hat{u}_{t,\lambda} - \tilde{u}_{t,\lambda}). \tag{131}$$

Applying the inequality

$$\|a + b\|_2^2 \le 2\|a\|_2^2 + 2\|b\|_2^2, \tag{132}$$

we obtain

$$\|\hat{u}_{t,\lambda} - u_t^*\|_2^2 \le 2\|\tilde{u}_{t,\lambda} - u_t^*\|_2^2 + 2\|\hat{u}_{t,\lambda} - \tilde{u}_{t,\lambda}\|_2^2. \tag{133}$$

Taking conditional expectation with respect to $\mathcal{F}_t$ gives

$$\mathbb{E}\big[\|\hat{u}_{t,\lambda} - u_t^*\|_2^2 \,\big|\, \mathcal{F}_t\big] \le 2\mathbb{E}\big[\|\tilde{u}_{t,\lambda} - u_t^*\|_2^2 \,\big|\, \mathcal{F}_t\big] + 2\mathbb{E}\big[\|\hat{u}_{t,\lambda} - \tilde{u}_{t,\lambda}\|_2^2 \,\big|\, \mathcal{F}_t\big]. \tag{134}$$

Using the pointwise bound

$$\|\hat{u}_{t,\lambda} - \tilde{u}_{t,\lambda}\|_2^2 \le (1-\lambda)^2\delta_t^2\|\tilde{g}_t\|_2^2, \tag{135}$$

we conclude that

$$\mathbb{E}\big[\|\hat{u}_{t,\lambda} - u_t^*\|_2^2 \,\big|\, \mathcal{F}_t\big] \le 2\mathbb{E}\big[\|\tilde{u}_{t,\lambda} - u_t^*\|_2^2 \,\big|\, \mathcal{F}_t\big] + 2(1-\lambda)^2\delta_t^2\mathbb{E}\big[\|\tilde{g}_t\|_2^2 \,\big|\, \mathcal{F}_t\big]. \tag{136}$$

This proves the claim. □

Table 2. TAGD hyperparameter configurations across all experiments. Panel A reports the settings for compact vision models on CIFAR-10. Panel B reports the settings for autoregressive language models on WikiText-103.

Panel A: CIFAR-10 small models

Hyperparameter	**SmallCNN**	**PatchMLP**	**MicroTransformer**
Base optimizer	AdamW / SGD	AdamW / SGD	AdamW / SGD
Subspace rank k	8	16	16
Gradient buffer m	16	32	32
`pca_every`	20	30	20
$\lambda_\perp$	0.20	0.20	0.20
`warmup_frac`	0.20	0.20	0.20
`ramp_frac`	0.30	0.30	0.30
`intervene_frac`	0.50	0.50	0.50
`fade_frac`	0.05	0.05	0.05
`use_topo`	✓	✓	✓
Epochs	30	30	30
Batch size	128	128	128
Learning rate (AdamW)	3e-4	3e-4	3e-4
Learning rate (SGD)	1e-2	1e-2	1e-2

Panel B: WikiText-103 language models

Hyperparameter	**LLaMA-100M**	**LLaMA-250M**	**LLaMA-350M**
Base optimizer	AdamW	AdamW	AdamW
Subspace rank k	16	16	16
Gradient buffer m	16	8	8
`pca_every`	30	30	30
$\lambda_\perp$	0.15	0.15	0.10
`warmup_tagd`	0.10	0.10	0.10
`ramp_tagd`	0.20	0.20	0.20
`intervene_frac`	1.00	1.00	1.00
`fade_frac`	0.05	0.05	0.05
`use_topo`	✓	✓	✓
Epochs	3	3	3
Batch size	16	8	4
Sequence length	512	512	512
Learning rate	3e-4	2e-4	2e-4
Weight decay	0.1	0.1	0.1
LR schedule	cosine + warmup	cosine + warmup	cosine + warmup
$\mathrm{lr}_{\min}$	3e-5	3e-5	3e-5

Hyperparameter	**GPT-J 70M**	**GPT-J 125M**	**GPT-J 200M**
Base optimizer	AdamW	AdamW	AdamW
Subspace rank k	12	12	12
Gradient buffer m	32	16	16
`pca_every`	40	40	40
$\lambda_\perp$	0.25	0.25	0.20
`warmup_tagd`	0.05	0.05	0.05
`ramp_tagd`	0.15	0.15	0.15
`intervene_frac`	1.00	1.00	1.00
`fade_frac`	0.05	0.05	0.05
`use_topo`	✓	✓	✓
Epochs	3	3	3
Batch size	16	8	4
Sequence length	512	512	512
Learning rate	3e-4	3e-4	2e-4
Weight decay	0.1	0.1	0.1
LR schedule	cosine + warmup	cosine + warmup	cosine + warmup
$\mathrm{lr}_{\min}$	3e-5	3e-5	3e-5

Class	AdamW						SGD					
	PatchMLP		SmallCNN		MicroViT		PatchMLP		SmallCNN		MicroViT	
	Base	TAGD	Base	TAGD	Base	TAGD	Base	TAGD	Base	TAGD	Base	TAGD
▶ Airplane	82.6	82.0	80.3	**84.0**	81.8	**82.3**	80.5	**81.5**	87.0	85.7	80.3	**80.9**
▶ Auto	92.5	**92.7**	93.3	**94.5**	83.4	83.0	90.3	89.3	91.4	**92.9**	80.3	77.2
▶ Bird	72.4	69.7	75.4	73.6	56.6	56.2	66.7	**69.8**	77.8	77.0	55.8	**59.3**
▶ Cat	63.0	**63.9**	67.6	**68.8**	46.9	**48.5**	67.0	65.5	74.7	71.2	45.1	**48.6**
▶ Deer	77.2	**77.3**	82.6	**84.7**	63.6	63.5	66.9	64.9	82.4	**84.4**	57.7	57.2
▶ Dog	65.7	65.1	73.2	**74.6**	73.6	**73.9**	58.6	**64.1**	77.6	**79.6**	63.5	**65.2**
▶ Frog	88.4	**89.4**	92.7	90.1	81.8	**82.2**	89.2	**89.6**	87.8	**89.7**	82.1	**84.7**
▶ Horse	82.4	81.6	82.2	**84.1**	83.8	**84.3**	80.5	77.1	88.6	88.3	73.7	**74.2**
▶ Ship	88.1	**89.1**	91.9	**93.8**	79.8	79.7	85.1	80.7	90.1	**93.3**	81.5	79.2
▶ Truck	86.8	**87.3**	86.0	**86.4**	79.4	**79.8**	85.3	**89.8**	89.4	**91.4**	79.9	**80.6**

Table 3. Per-class CIFAR-10 accuracy (%) under AdamW and SGD. Baseline results are shown in gray, and improvements from TAGD are highlighted in blue. TAGD yields more frequent gains under SGD, consistent with its role in stabilizing noisier optimization dynamics.