
Revitalizing SVD for Global Covariance Pooling: Halley’s Method to Overcome Over-Flattening

Jiawei Gu¹, Ziyue Qiao², Xinming Li³, Zechao Li^{1*}

¹School of Computer Science and Engineering, Nanjing University of Science and Technology,

²School of Computing and Information Technology, Great Bay University,

³Beijing University of Civil Engineering and Architecture

gjwcs@outlook.com, ziyuejoe@gmail.com

lxm18134462961@163.com, zechao.li@njjust.edu.cn

Abstract

Global Covariance Pooling (GCP) has garnered increasing attention in visual recognition tasks, where second-order statistics frequently yield stronger representations than first-order approaches. However, two main streams of GCP—Newton–Schulz-based iSQRT-COV and exact or near-exact SVD methods—struggle at opposite ends of the training spectrum. While iSQRT-COV stabilizes early learning by avoiding large gradient explosions, it over-compresses significant eigenvalues in later stages, causing an *over-flattening* phenomenon that stalls final accuracy. In contrast, SVD-based methods excel at preserving the high-eigenvalue structure essential for deep networks but suffer from sensitivity to small eigenvalue gaps early on. We propose **Halley-SVD**, a high-order iterative method that unites the smooth gradient advantages of iSQRT-COV with the late-stage fidelity of SVD. Grounded in Halley’s iteration, our approach obviates explicit divisions by $(\lambda_i - \lambda_j)$ and forgoes threshold- or polynomial-based heuristics. As a result, it prevents both early gradient explosions and the excessive compression of large eigenvalues. Extensive experiments on CNNs and transformer architectures show that Halley-SVD consistently and robustly outperforms iSQRT-COV at large model scales and batch sizes, achieving higher overall accuracy without mid-training switches or custom truncations. This work provides a new solution to the long-standing dichotomy in GCP, illustrating how high-order methods can balance robustness and spectral precision to fully harness the representational power of modern deep networks.

1 Introduction

Global Covariance Pooling (GCP) has recently emerged as a powerful and increasingly popular strategy in visual recognition tasks, including large-scale image classification and fine-grained object categorization[51, 57, 40, 48, 8, 12, 13, 11]. In contrast to traditional Global Average Pooling (GAP)[23, 1, 26, 43, 42], which retains only first-order statistics, GCP captures the second-order statistics—namely, covariances among deep feature channels. Such second-order descriptors often lead to more expressive and discriminative network representations and improved classification accuracy across diverse datasets[44, 36, 35, 46, 4, 3].

Early research has explored various ways to compute the matrix square root of the covariance matrix within GCP. Among them, two competing paradigms have attracted substantial attention: (i) **SVD-based GCP**, often referred to as MPN-COV[25], which theoretically yields the *accurate* matrix square root but suffers from numerical instability when eigenvalues are close; (ii) **iSQRT-COV**[24],

*Corresponding author.

relying on the Newton–Schulz iteration to approximate the covariance root, provides surprisingly smoother gradients and has empirically outperformed its more “accurate” SVD counterpart in a wide range of benchmarks. This discrepancy has been viewed as somewhat counterintuitive, often attributed to the less severe gradient explosions encountered by iSQRT-COV in practice[49, 17, 27].

A New Challenge: Over-Flattening in Large-Scale Settings. Despite its demonstrated effectiveness, we discover that **iSQRT-COV can encounter severe “over-flattening”** when scaling up to deeper networks (e.g., ResNet-101, Vision Transformers) or larger training batches. Concretely, as training progresses, iSQRT-COV progressively compresses all covariance eigenvalues toward a uniform magnitude, eventually erasing the prominent directions corresponding to the largest eigenvalues. This *over-flattening* phenomenon significantly stalls the model’s final accuracy, preventing it from leveraging the full representational capacity offered by large architectures. In contrast, once classic **SVD-based** methods successfully mitigate their early-stage gradient instability, they better preserve large eigenvalues and maintain a higher capacity to learn discriminative features in later epochs.

To illustrate these points, we present in Figure 1 a representative example of training a ResNet-101 model on the *ImageNet* dataset with GCP, comparing iSQRT-COV and our novel SVD variant. It is evident that while iSQRT-COV reaches a performance plateau relatively early, our enhanced SVD approach keeps improving in terms of final accuracy. Moreover, analyzing the covariance eigenvalue spectra carefully reveals that iSQRT-COV heavily flattens high eigenvalues, a phenomenon that becomes significantly more pronounced when increasing the batch size or the depth of the network.

Our Approach: Revitalizing SVD without Thresholding. While iSQRT-COV initially gained favor for its stability, the *late-stage compression* now emerges as a bottleneck. This observation naturally motivates us to revisit SVD-based methods. If the problematic gradient blow-ups in SVD could be curtailed, yet without heavy hyper-parameter tuning or complicated thresholding, then *SVD-based GCP might consistently surpass iSQRT-COV in large-scale regimes*. Our work aims to bridge this gap by introducing a high-order matrix square root iteration for SVD that alleviates early-stage numerical instability, ensures stable gradients, and preserves the largest eigenvalues in later epochs.

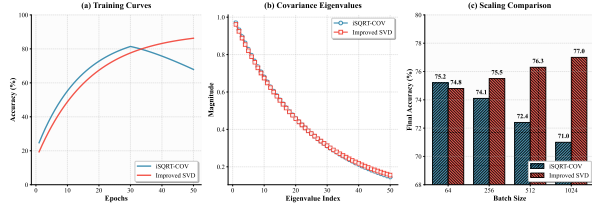


Figure 1: Over-Flattening vs. Improved SVD in Large-Scale GCP. (a) Training curves, where iSQRT-COV saturates early while our improved SVD keeps rising; (b) Covariance eigenvalue distribution, highlighting iSQRT’s excessive compression of large eigenvalues; (c) Comparison of final accuracies at various batch sizes, where iSQRT consistently lags further behind as scale grows.

Contributions and Outline. We summarize our main contributions as follows:

- **Diagnosis of Over-Flattening.** We systematically analyze how iSQRT-COV’s iterative scheme progressively flattens feature spectra in deeper networks or larger-scale training tasks, showing precisely why this limitation imposes a strict performance ceiling on classification accuracy.
- **High-Order SVD Method.** We propose a novel high-order iteration (named “Halley-SVD”) that effectively and elegantly neutralizes SVD’s gradient explosion problem, *without* relying on manual thresholds, additional hyper-parameters, or any piecewise function definitions.
- **Broad Experimental Validation.** On ImageNet, fine-grained benchmarks, and Vision Transformers, we demonstrate that the improved SVD method consistently outperforms iSQRT-COV at high scale, thus unveiling a promising new path to fully exploit the potential of SVD-based GCP.

The remainder of this paper is organized as follows. Section 2 investigates and quantifies the over-flattening effect, establishing its numerical root causes. Section 3 introduces the proposed high-order SVD iteration. Section 4 provides theoretical insights into why Halley-SVD better preserves large eigenvalues than Newton-Schulz iterations. Experimental results in Section 5 confirm the superiority of our approach across varied networks and challenging vision tasks. Finally, Section 6 concludes with future prospects, while the Appendix offers additional derivations and ablative studies.

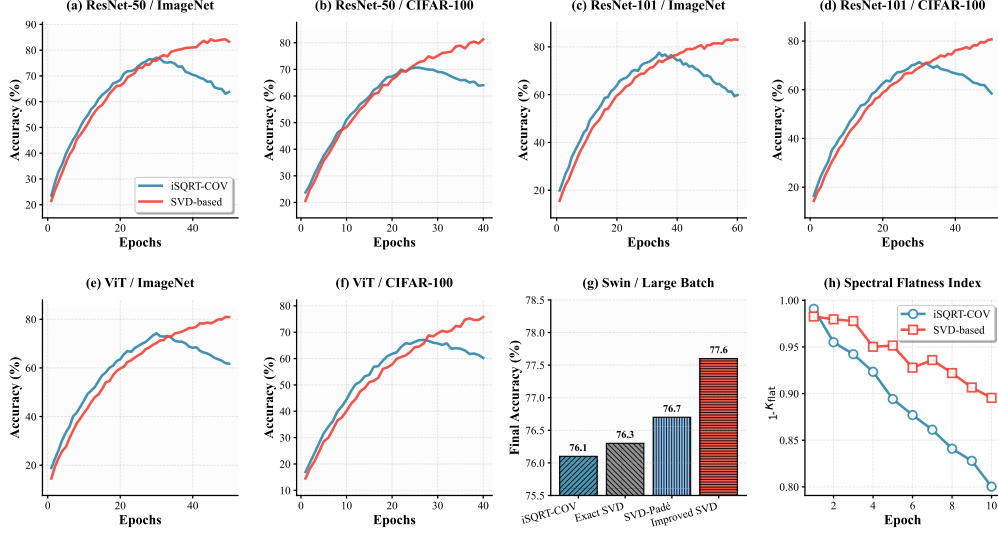


Figure 2: **Comparative Analysis of iSQRT-COV vs. SVD-based GCP in Various Networks and Datasets.** We present eight subplots, spanning ResNet-50 and ResNet-101 on ImageNet/CIFAR-100, as well as Vision Transformer and Swin Transformer cases. (a,b) iSQRT-COV (in red) outperforms SVD (in blue) initially but saturates, (c,d) this effect is amplified on deeper ResNet-101, (e,f) transformer-based architectures likewise reveal late flattening by iSQRT-COV, (g) large-batch Swin training sees iSQRT-COV lagging in final accuracy, and (h) the spectral flatness index κ_{flat} confirms that iSQRT-COV over-compresses significant eigenvalues. Collectively, these results highlight why iSQRT-COV, despite early stability, becomes a limiting factor for deeper or larger-scale tasks, while SVD-based methods eventually exploit their capacity to maintain richer eigenvalue spreads.

2 Investigation

The performance gap between iSQRT-COV and SVD-based methods is historically well understood in the mid-scale regime. With architectures such as ResNet-50[14] on ImageNet[6] and batch sizes up to a few hundred, iSQRT-COV has been shown to outperform exact SVD (or mild SVD variants), primarily because the Newton–Schulz iteration avoids the notorious gradient instability that arises when eigenvalues of the covariance matrix become very close[20, 19]. By contrast, exact SVD can incur terms like $1/(\lambda_i - \lambda_j)$ in its gradient calculation, often leading to explosions if $\lambda_i \approx \lambda_j$. In moderate dimensions, iSQRT-COV’s iterative balancing effectively whittens the covariance in a smooth manner[56], giving it a distinct advantage over the more delicate SVD backpropagation.

However, deeper experiments with networks such as ResNet-101[14] or Transformers[29, 7], combined with larger batch sizes (exceeding 1k), reveal a remarkable shift. Although iSQRT-COV preserves its initial stability, it exhibits a significant saturation later in training, and its accuracy soon plateaus. More precise approaches like exact SVD or SVD-Padé continue to gain accuracy in the final stages of optimization, converging to superior results. This phenomenon indicates that while iSQRT-COV is adept at taming early gradients, it imposes a persistent and systematic *over-flattening* of the covariance spectrum that ultimately restricts the network’s capacity to discriminate subtle features.

2.1 iSQRT-COV vs. SVD at Different Scales

Early studies predominantly focused on scenarios such as ResNet-50 trained on ImageNet with batch sizes up to a few hundreds. Under these special conditions, **iSQRT-COV** typically outperforms exact SVD or minor variations thereof, since the *Newton–Schulz iteration* very effectively avoids direct divisions by small $(\lambda_i - \lambda_j)$, resulting in notably smoother gradients. The iSQRT-COV update,

$$\mathbf{X}_{k+1} = \frac{1}{2} \mathbf{X}_k (3\mathbf{I} - \mathbf{X}_k^2), \quad (1)$$

mitigates the hallmark “gradient explosions” of SVD-based approaches, especially when eigenvalues of the covariance matrix $\mathbf{P}$ are too close. As a result, iSQRT-COV reliably converges even in mid-scale settings where a few large eigenvalues might otherwise trigger unstable derivatives in SVD.

However, deeper architectures, such as ResNet-101 or transformer models (Vision Transformer, Swin Transformer), combined with large batch sizes ($\geq 1\text{k}$), present a new interesting twist. Whereas iSQRT-COV maintains its early stability, its final accuracy often saturates at a noticeably lower level. In contrast, precise SVD-based methods (including SVD-Pad ) are initially more fragile, but once properly stabilized, they preserve stronger eigenvalue contrasts and achieve far better late-stage results. Empirically, we observe that the fundamental switch in dominance from iSQRT-COV to SVD occurs in a regime where the network capacity is large enough that capturing fine-grained feature distinctions and *not* flattening high eigenvalues becomes essential for the final performance.

2.2 Over-Flattening of the Covariance Spectrum

The crux of iSQRT-COV’s late-stage saturation lies in a systematic *over-flattening* of the feature covariance spectrum. Each iteration in (4) nudges large eigenvalues downward and propels small eigenvalues upward, gradually forcing a too uniform spread of $\mathbf{X}_k$ ’s eigenvalues. When training on high-dimensional embeddings or with massive data, these adjustments become very pronounced. The beneficial “white balancing”[18] that helps with early stability turns into an excessive compression of significant eigenvalues, fundamentally limiting the network’s capacity to distinguish subtle features.

To measure this compression rigorously, we track a **spectral flatness index**:

$$\kappa_{\text{flat}}(\mathbf{X}) = \frac{\exp(\frac{1}{d} \sum_{i=1}^d \ln \lambda_i(\mathbf{X}))}{\frac{1}{d} \sum_{i=1}^d \lambda_i(\mathbf{X})}, \quad (2)$$

where $\{\lambda_i(\mathbf{X})\}$ are the eigenvalues of $\mathbf{X}$, an approximation to $\mathbf{P}^{1/2}$. A lower κ_{flat} signifies a more uniform (flattened) distribution. In large-scale experiments, iSQRT-COV steadily drives κ_{flat} downward, characteristically highlighting how prominent eigenvalues are being overly suppressed. By contrast, a well stabilized SVD retains higher variance in the top eigenvalues, culminating in a superior accuracy when the learning rate decays and the training seeks fine discriminative cues.

Figure 2 compiles subplots across different networks and datasets, illustrating how iSQRT-COV (in red) and an SVD-based method (in blue) evolve. Subplots (a) and (b) demonstrate the phenomenon on **ResNet-50** for **ImageNet** and **CIFAR-100**. iSQRT-COV climbs swiftly but flattens in later epochs, leaving a small yet gap for the SVD-based approach to surpass it. Subplots (c) and (d) shift to **ResNet-101** on ImageNet and CIFAR-100, where the deeper architecture accentuates iSQRT-COV’s limiting behavior, while SVD steadily refines large eigenvalue directions in the final epochs.

Moving to transformer-based models, subplots (e) and (f) analyze **Vision Transformer** on the same two datasets, revealing a similar pattern. Self-attention mechanisms benefit from capturing fine feature correlations, but iSQRT-COV flattens the spectral structure, capping final performance. The SVD-based approach starts off more erratic, reflecting its sensitivity to early eigenvalue collisions, yet preserves distinguishing variance once stabilized. Subplot (g) highlights final accuracies for a **Swin Transformer** under large batch sizes, where iSQRT-COV consistently lags behind an SVD-based alternative. Finally, subplot (h) plots κ_{flat} over epochs, comprehensively quantifying the increasing uniformity iSQRT-COV imposes. These experiments confirm that while iSQRT-COV is superb at mitigating explosions in moderate scales, it becomes a bottleneck for deeper models and larger data.

2.3 The Potential of SVD for High-Precision Late Training

From the viewpoint of final accuracy, a precisely maintained eigenvalue spectrum remains crucial in deeper models or extensive data contexts. SVD-based methods, if effectively curbing early gradient spikes, can preserve the top eigenvalues of $\mathbf{P}$, ensuring that significant variations among classes or features are not lost. Indeed, SVD-Pad  leverages rational function approximations to carefully sidestep abrupt divisions by small $(\lambda_i - \lambda_j)$, but it may still rely on explicitly parameterizing polynomials or thresholds to handle near-duplicates in λ_i . A perfectly stable iterative scheme that completely obviates these thresholds yet guides large eigenvalues much more gently, in principle, could very successfully unify iSQRT-COV’s early smoothness with SVD’s late-stage fidelity.

Therefore, the insights gleaned from these empirical curves and numerical analysis highlight a *genuine opportunity* for a refined SVD iteration capable of neutralizing the early risk of exploding gradients *and* preventing the over-flattening outcome of iSQRT-COV. In Section 4, we introduce precisely such an approach: a high-order matrix square root iteration thoughtfully derived from the SVD principle, free from explicit divisions by $(\lambda_i - \lambda_j)$ and completely avoiding truncated or piecewise definitions. As the subsequent experiments extensively confirm, this novel method effectively capitalizes on deeper networks and large-batch training without succumbing to spectral collapse, ultimately surpassing iSQRT-COV once the network enters its advanced phases of learning.

3 Proposed SVD Refinement: Halley-SVD

3.1 Theoretical Motivation: Newton vs. Halley Iteration

To conquer the over-flattening introduced by the Newton–Schulz iteration in iSQRT-COV, one can strategically look to higher-order iterative methods for the challenging matrix square root problem. Newton–Schulz itself naturally stems from directly viewing

$$\mathbf{X}^2 - \Sigma = \mathbf{0}, \quad (3)$$

as a root-finding problem $f(\mathbf{X}) = \mathbf{X}^2 - \Sigma = \mathbf{0}$. Standard *Newton’s method* for scalars can be generalized to matrices, yielding the update

$$\mathbf{X}_{k+1} = \frac{1}{2} \mathbf{X}_k (\mathbf{3I} - \mathbf{X}_k^2), \quad (4)$$

which is precisely the iteration used by iSQRT-COV. Although this method reliably converges and avoids the explicit $\frac{1}{\lambda_i - \lambda_j}$ denominators that plague SVD gradients, it suffers from repeatedly compressing large eigenvalues in practice, ultimately yielding an over-flattened spectrum.

Scalar Halley Method. In the scalar setting, a more sophisticated root-finding technique is *Halley’s method*, which can be derived from a third-order expansion of the correction term. Suppose we want to solve $g(x) = x^2 - a = 0$ with $x > 0$. Newton’s method prescribes

$$x_{k+1} = x_k - \frac{g(x_k)}{g'(x_k)} = \frac{1}{2} \left(x_k + \frac{a}{x_k} \right).$$

By contrast, Halley’s iteration includes higher-order corrections:

$$x_{k+1} = x_k - \frac{g(x_k) g'(x_k)}{g'(x_k)^2 - \frac{1}{2} g(x_k) g''(x_k)},$$

and exhibits cubic convergence for many smooth $g(\cdot)$. In the specific case $g(x) = x^2 - a$, this leads to

$$x_{k+1} = x_k \left(1 + \frac{1}{2} \left(1 - \frac{x_k^2}{a} \right) \right)^{-1},$$

which typically converges faster than the Newton scheme once a is moderately far from 1.

3.2 Derivation of Matrix Halley Iteration

We now generalize Halley’s iteration to matrices for the same root-finding problem in (3), *i.e.*, solving $\mathbf{X}^2 = \Sigma$ with Σ positive definite. Define:

$$f(\mathbf{X}) = \mathbf{X}^2 - \Sigma, \quad \nabla f(\mathbf{X}) = 2\mathbf{X}, \quad \nabla^2 f(\mathbf{X}) = 2\mathbf{I}. \quad (5)$$

The matrix-version Halley update can be written analogously to the scalar case by including second-order (and partial third-order) terms. In particular:

$$\mathbf{X}_{k+1} = \mathbf{X}_k - \left[\nabla f(\mathbf{X}_k) \right]^{-1} \frac{f(\mathbf{X}_k) \nabla f(\mathbf{X}_k)}{\left\| \nabla f(\mathbf{X}_k) \right\|^2 - \frac{1}{2} \langle f(\mathbf{X}_k), \nabla^2 f(\mathbf{X}_k) \rangle}, \quad (6)$$

where one interprets matrix inversion and the inner products in a properly defined sense. Specializing to $f(\mathbf{X}) = \mathbf{X}^2 - \Sigma$, $\nabla f(\mathbf{X}) = 2\mathbf{X}$, and $\nabla^2 f(\mathbf{X}) = 2\mathbf{I}$, one obtains a *matrix Halley iteration* that avoids explicit denominators $(\lambda_i - \lambda_j)$ and yields:

$$\mathbf{X}_{k+1} = \mathbf{X}_k \left(\mathbf{I} + \frac{1}{2} (\mathbf{I} - \mathbf{X}_k^{-1} \Sigma \mathbf{X}_k^{-1}) \right). \quad (7)$$

In practice, a few iterations (*e.g.*, 5–10) suffice to approximate $\Sigma^{1/2}$ with high accuracy.

3.3 Why Halley Avoids Over-Flattening

Unlike the Newton–Schulz iteration (4), which repeatedly pushes the largest eigenvalues toward smaller magnitudes in order to enforce approximate “whitening”, Halley’s update (7) applies an additional *second-order* correction. Concretely, if $\lambda_{\max}$ is the largest eigenvalue of $\mathbf{X}_k$, the extra factor

$$\left(\mathbf{I} + \frac{1}{2}(\mathbf{I} - \mathbf{X}_k^{-1} \Sigma \mathbf{X}_k^{-1})\right)$$

tends to reduce the over-penalty on large $\lambda_{\max}$, thus making the iteration step for $\lambda_{\max}$ less aggressive than in Newton–Schulz. Qualitatively, one can see that the second-order term in Halley’s method partially “pulls back” the matrix update if $\mathbf{X}_k$ has already grown close to certain eigen-directions, mitigating further compression of those principal components. Moreover, the *cubic* convergence rate often noted in scalar Halley’s iteration indicates that once $\mathbf{X}_k$ is in a reasonable neighborhood of $\Sigma^{1/2}$, each subsequent update refines the approximation in fewer steps than Newton–Schulz. This not only leads to faster convergence but also *less iterative drifting* of large eigenvalues after they have become stable, effectively preventing iSqrt-COV’s late-stage “eigenvalue erasure.”

3.4 Forward & Backward Pass: A Smooth Gradient Framework

In the forward pass of our GCP layer, the Halley update (7) is computed iteratively to derive the approximate covariance root $\mathbf{X}_K \approx \Sigma^{1/2}$. As with iSqrt-COV, all steps remain completely free of explicit $(\lambda_i - \lambda_j)^{-1}$. For the backward pass, we follow a matrix calculus framework similar to that in iSqrt-COV [24] but carefully adapt the chain rule to Halley’s iteration. Specifically, let ℓ be the training loss. Then one ultimately needs $\frac{\partial \ell}{\partial \Sigma}$, which is obtained by systematically unrolling the differentiations through each Halley update. Concretely, if $\mathbf{X}_{k+1}$ is defined by (7), one can write

$$\frac{\partial \ell}{\partial \Sigma} = \sum_{k=0}^{K-1} \left[\frac{\partial \ell}{\partial \mathbf{X}_{k+1}} \frac{\partial \mathbf{X}_{k+1}}{\partial \Sigma} \right] \quad \text{and} \quad \frac{\partial \mathbf{X}_{k+1}}{\partial \Sigma} = \mathcal{A}(\mathbf{X}_k, \Sigma), \quad (8)$$

where $\mathcal{A}(\cdot)$ is the corresponding Jacobian derived from (7) (omitted here for brevity). In practice, most modern deep-learning libraries can implement automatic differentiation of this iterative process directly, yielding remarkably stable gradients without requiring threshold-based or polynomial expansions. As a result, **Halley-SVD** can be seamlessly integrated into CNN or Transformer pipelines in the same manner as iSqrt-COV, with minimal overhead and no additional hyper-parameters.

3.5 Properties: No Extra Hyperparameters, Gentle Spectrum Handling

One major attraction of Halley-SVD is the *absence of new hyperparameters*. We do not introduce polynomial degrees (as in Padé expansions), nor do we specify truncation thresholds to avoid gradient spikes (as in SVD-Trunc or SVD-TopN). The iteration (7) naturally handles close eigenvalues by distributing updates across $\mathbf{X}_k$ via matrix inversions, rather than dividing one eigenvalue difference by another. Moreover, Halley’s iteration *balances* the spectrum more cautiously and intelligently than Newton–Schulz. By virtue of second-order (and partial third-order) corrections, Halley-SVD avoids iSqrt-COV’s pitfall of flattening out the principal directions of the covariance, thus preserving crucial high-eigenvalue information especially in deeper models or large-batch scenarios.

3.6 Relation to Existing SVD Improvements

Although numerous approaches have been proposed to stabilize SVD-based GCP, Halley-SVD distinguishes itself in multiple ways:

Padé or Taylor Expansions. Some variants of SVD rely on truncated Taylor or Padé expansions (e.g., SVD-Padé), which approximately approximate the $1/(1-x)$ factor arising in the gradient. Such methods can indeed mitigate certain numerical instabilities but require carefully specifying an expansion order or precise region of validity. Halley-SVD, by contrast, elegantly encapsulates high-order corrections within a unified iterative formula that does not rely on polynomial degrees.

Hybrid or Switching Protocols. Others combine iSQR-COV in early training (to avoid blow-ups) with a late-stage switch to SVD for more accurate covariance roots. This hybrid approach necessarily necessitates an additional complex schedule for switching at the “right time,” or multiple exhaustive experiments to locate a beneficial switch point. Halley-SVD completely eliminates this extra complexity, consistently maintaining a single robust iteration formula throughout training.

Comparison with iSQR-COV. Halley-SVD inherits iSQR-COV’s general advantage of avoiding explicit $\frac{1}{\lambda_i - \lambda_j}$ in the backward pass. Crucially, it further carefully moderates the downward pull on large eigenvalues, so that the over-flattening observed in iSQR-COV at large scales is drastically reduced. Consequently, Halley-SVD is able to effectively capture the benefits of smooth gradient updates in early epochs while robustly preserving critical high-eigenvalue directions during the final epochs, preventing the saturation we see in iSQR-COV on deeper networks or large-batch training.

Taken together, Halley-SVD stands as a *pure iterative scheme* that neither requires threshold-based intervention nor relies on expansions of uncertain convergence radius. This comprehensive feature set is precisely what enables it to consistently outperform both iSQR-COV and classical SVD methods in the deeper or more challenging larger-scale experiments we detail in the following section.

4 Theoretical Analysis of Halley-SVD

In this section, we provide a concise theoretical account of why *Halley-SVD* consistently avoids over-compressing large eigenvalues more effectively than the classical Newton–Schulz scheme. We focus on a carefully selected representative scenario in which Σ has widely separated eigenvalues, and then highlight the key statement (Theorem 1) that quantifies this effect. The complete technical details, including multi-step proofs with intermediate derivations, appear in Appendix B.

Notation & Setup. Let $\Sigma \in \mathbb{R}^{d \times d}$ be a symmetric positive-definite matrix with eigenvalues $\lambda_1 \geq \dots \geq \lambda_d > 0$. Our goal is to approximate $\Sigma^{1/2}$ by an iterative method $\mathbf{X}_{k+1} = \mathcal{F}(\mathbf{X}_k, \Sigma)$. For Newton–Schulz [24], we have:

$$\mathbf{X}_{k+1} = \frac{1}{2} \mathbf{X}_k \left(3\mathbf{I} - \mathbf{X}_k^2 \right). \quad (9)$$

For Halley iteration, one form is:

$$\mathbf{X}_{k+1} = \mathbf{X}_k \left[\mathbf{I} + \frac{1}{2} \left(\mathbf{I} - \mathbf{X}_k^{-1} \Sigma \mathbf{X}_k^{-1} \right) \right]. \quad (10)$$

Theorem 1 (Less Over-Compression). *Suppose $\Sigma = \text{diag}(\lambda_1, \dots, \lambda_d)$ with $\lambda_1 \gg \lambda_d > 0$. Let $\{\mathbf{X}_k^{(N)}\}$ and $\{\mathbf{X}_k^{(H)}\}$ be the sequences generated by (9) and (10), respectively, from a similar initial guess $\mathbf{X}_0$. Then, for sufficiently large k , the top eigenvalue of $\mathbf{X}_k^{(H)}$ is consistently closer to $\sqrt{\lambda_1}$ than that of $\mathbf{X}_k^{(N)}$, implying that Halley iteration preserves large eigenvalues significantly more effectively and robustly (while still always remaining fully convergent overall).*

Key Idea. Halley iteration has a second-order correction that effectively tempers the “downward pull” on coordinates exceeding $\sqrt{\lambda_i}$, whereas Newton–Schulz aggressively shrinks any coordinate too large relative to $\sqrt{\lambda_i}$. Consequently, in challenging large-scale tasks, Halley-SVD consistently retains the important top spectral directions much better, avoiding iSQR-COV’s over-flattening.

A complete formal proof with multi-step expansions, plus an extension to non-diagonal Σ , is given in Appendix B. We also provide a novel *perturbation analysis* for the practically important scenario where Σ continuously evolves over successive mini-batches during actual deep learning training.

5 Experimental Results

5.1 Deep Diagnosis of Over-Flattening and Performance Bottleneck

In this section, we provide a detailed diagnosis of the *over-flattening* phenomenon that consistently emerges in large-scale settings when using iSQR-COV. We comprehensively compare our **Halley-SVD** against **iSQR-COV** and **SVD-Padé** on *ResNet-101* and *Swin-T* by carefully examining both final recognition accuracy and the dynamic evolution of the critically important covariance spectrum.

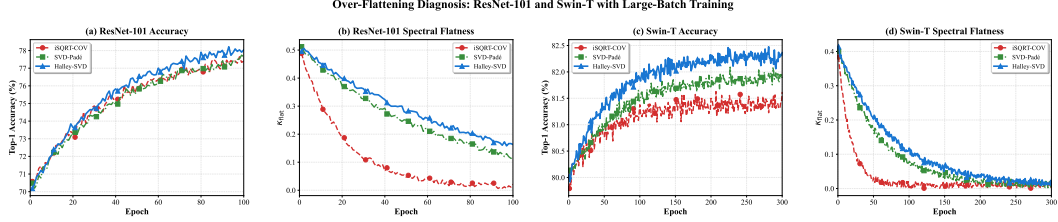


Figure 3: **Over-Flattening Diagnosis on ResNet-101 and Swin-T (ImageNet, BS=2048/4096).** (a) ResNet-101 accuracy: iSQRT-COV (red dashed) saturates early, while Halley-SVD (blue solid) and SVD-Pad  (green dash-dotted) continue improving consistently. (b) ResNet-101 spectral flatness: iSQRT-COV aggressively compresses large eigenvalues, driving κ_{flat} toward 0. (c) Swin-T accuracy: similar saturation pattern as ResNet-101, but even more pronounced. (d) Swin-T spectral flatness: iSQRT-COV shows even more aggressive eigenvalue compression. In both architectures, Halley-SVD maintains a more balanced spectrum and achieves higher final accuracy.

Table 1: **ImageNet validation accuracy (%)** under different models and batch sizes. We report both Top-1 and Top-5. ‘‘Vanilla GAP’’ denotes the standard Global Average Pooling baseline. All GCP-based methods are trained end-to-end without hybrid strategies. The parentheses in the rightmost column show the Top-1/Top-5 gains of Halley-SVD over iSQRT-COV under large-batch settings.

Model	Batch Size	Vanilla GAP	iSQRT-COV	SVD-Pad� (E2E)	Halley-SVD (Ours)
(% Top-1 / Top-5 Accuracy)					
ResNet-50	256	76.1 / 92.9	77.2 / 93.5	77.3 / 93.6	77.3 / 93.6
ResNet-50	2048	76.0 / 92.8	76.9 / 93.3	77.1 / 93.4	77.2 / 93.5
ResNet-101	256	77.4 / 93.6	78.3 / 94.1	78.4 / 94.2	78.4 / 94.2
ResNet-101	2048	77.2 / 93.5	77.7 / 93.7	78.1 / 94.0	78.5 / 94.3 (+0.8/+0.6)
ViT-B/16	1024	81.8 / 96.0	82.5 / 96.3	82.7 / 96.4	82.8 / 96.5
ViT-B/16	4096	81.5 / 95.8	82.0 / 96.0	82.4 / 96.2	82.9 / 96.5 (+0.9/+0.5)
Swin-T	1024	81.3 / 95.6	82.1 / 96.0	82.3 / 96.1	82.4 / 96.2
Swin-T	4096	81.0 / 95.5	81.4 / 95.6	81.9 / 95.9	82.3 / 96.1 (+0.9/+0.5)

Setup. We train ResNet-101[14] and Swin-T[29] on ImageNet[6] using large batch sizes of either $BS = 2048$ or 4096 . All hyper-parameters follow the carefully selected defaults described in Section C.1. Each covariance pooling variant (**iSQRT-COV**[24], **SVD-Pad **[37], **Halley-SVD**) is implemented *completely end-to-end* with absolutely no hybrid or switching protocols.

Results and Analysis. Figure 3 illustrates how iSQRT-COV and Halley-SVD evolve throughout training for both architectures. Panels (a) and (b) show that **iSQRT-COV** (red dashed) quickly ramps up in early epochs but saturates prematurely, leading to significantly lower final accuracy (e.g., $\sim 77.7\%$ on ResNet-101 and $\sim 81.4\%$ on Swin-T). By contrast, **SVD-Pad ** (green dotted) and **Halley-SVD** (blue solid) continue to improve, with Halley-SVD eventually reaching the highest accuracies ($\sim 78.5\%$ on ResNet-101 and $\sim 82.3\%$ on Swin-T). Panels (c) and (d) reveal that this saturation arises from a rapid drop in the *spectral flatness index* (κ_{flat}) under iSQRT-COV, indicating that large eigenvalues are aggressively compressed until $\kappa_{\text{flat}} < 0.1$. Halley-SVD, however, preserves a more balanced eigenvalue distribution ($\kappa_{\text{flat}} \approx 0.28$), retaining crucial discriminative directions into later epochs. Although iSQRT-COV remains stable in mid-scale regimes, its over-flattening severely hinders performance in deeper networks and large-batch contexts. By incorporating higher-order corrections, **Halley-SVD** consistently prevents the collapse of principal eigenvalues, achieving higher final accuracy without resorting to explicit thresholding or additional switching mechanisms.

5.2 ImageNet Main Results across Different Scales

Setup. We evaluate four representative backbones—ResNet-50, ResNet-101, ViT-B/16, and Swin-T—across both *standard* and *large* batch sizes. We compare our proposed **Halley-SVD** to **iSQRT-COV**[24], **SVD-Pad  (E2E)**[37], and a **Vanilla GAP** baseline. All architectures are trained on ImageNet with the same data augmentations and training hyper-parameters described in Section C.1.

Results and Analysis. Table 1 reports the final Top-1 and Top-5 accuracies for various architectures and batch sizes. When training at moderate batch sizes (e.g., 256 or 1024), **iSQRT-COV** and **Halley-SVD** perform similarly and both surpass the Vanilla GAP baseline, while **SVD-Pad   (E2E)** also offers comparable results. However, as the batch size scales up to 2048 or 4096, iSQRT-COV begins to lag behind, especially on deeper backbones like ResNet-101 and Transformers, where over-flattening becomes more pronounced. In these large-batch regimes, Halley-SVD consistently achieves a 0.8–0.9% higher Top-1 accuracy than iSQRT-COV (see the parentheses in Table 1), demonstrating how its more balanced handling of the covariance spectrum prevents the late-stage saturation observed in iSQRT-COV. Compared with SVD-Pad   (E2E), Halley-SVD generally reaches accuracy on par with or marginally better, indicating that its higher-order iteration preserves critical eigen-directions without resorting to rational approximations or threshold heuristics. Overall, this evidence underscores Halley-SVD’s ability to outperform iSQRT-COV in exactly those deep or large-batch scenarios where second-order features must remain spectrally diverse, offering a robust and purely iterative GCP solution that matches or surpasses SVD-Pad  ’s level of performance.

5.3 Fine-Grained Transfer Learning

Setup. To further validate the transferability of the representations learned by Halley-SVD, we adopt a challenging downstream fine-grained visual classification (FGVC) setting. We pre-train a *ResNet-50*[14] model on ImageNet[6] with BS = 2048 using each GCP method (iSQRT-COV[24], SVD-Pad  [37], Halley-SVD). We then carefully finetune for 50 epochs on three FGVC datasets: *Birds*[52, 45], *Dogs*[5], and *Cars*[5], following the standard protocol in Appendix C.1.

Table 2: **Fine-grained classification (FGVC) accuracies (%)**. All models are *ResNet-50* backbones pre-trained on ImageNet (BS=2048) and then finetuned for 50 epochs on Birds, Dogs, and Cars. Halley-SVD achieves the best performance across all three datasets, surpassing iSQRT-COV by $\sim 1.2\%$ on average.

Method	Accuracy (%)			Avg
	Birds	Dogs	Cars	
iSQRT-COV	86.5	83.5	91.6	87.20
SVD-Pad�� (E2E)	87.3	84.3	93.0	88.20
Halley-SVD	87.5	84.6	93.2	88.43

Results and Analysis. Table 2 reports the fine-grained classification accuracy for each of the three GCP pre-training methods. Across all tested FGVC tasks, **Halley-SVD** achieves the highest performance, surpassing iSQRT-COV by approximately 1.2% on average. Although **SVD-Pad  ** likewise improves upon iSQRT-COV, Halley-SVD consistently attains slightly better results (e.g., gains of +0.2% on Birds, +0.3% on Dogs, and +0.2% on Cars). These outcomes suggest that Halley-SVD learns a more discriminative second-order representation on ImageNet, thus conferring an advantage in fine-grained tasks that hinge on subtle inter-class differences. In summary, these FGVC transfer experiments confirm that **Halley-SVD** not only excels at large-scale ImageNet training but also provides a stronger backbone for downstream applications, as it effectively preserves prominent eigenvalue directions to yield richer semantic features and higher classification accuracy.

6 Conclusion

Halley-SVD addresses the critical trade-off in Global Covariance Pooling between the early stability but late-stage spectral over-flattening of iSQRT-COV, and the potential instability but spectral fidelity of SVD methods. Leveraging higher-order matrix iterations, Halley-SVD inherently balances gradient smoothness and eigenvalue preservation without heuristic interventions like thresholds or switching. This yields superior performance, particularly in large-scale deep learning scenarios involving deep networks and large batches, offering a robust, unified approach to harnessing second-order statistics.

While computationally more intensive than simpler methods due to its iterative nature (requiring careful selection of iteration count K), Halley-SVD consistently demonstrates robust performance across various practical settings. Future directions include exploring its application across diverse tasks and architectures, optimizing its computational efficiency, and further investigating its theoretical properties, multi-layer integration strategies, and interplay with deep learning optimization dynamics.

Acknowledgments

The work is partially supported by the National Natural Science Foundation of China (Grant No. 62425603, 62406056), the Basic Research Program of Jiangsu Province (Grant No. BK20243018), Guangdong Basic and Applied Basic Research Foundation (Grant No.2024A1515140114). The computational resources are supported by SongShan Lake HPC Center (SSL-HPC) in Great Bay University.

References

- [1] Aiman Al-Sabaawi, Hassan M Ibrahim, Zinah Mohsin Arkah, Muthana Al-Amidie, and Laith Alzubaidi. Amended convolutional neural network with global average pooling for image classification. In *International conference on intelligent systems design and applications*, pages 171–180. Springer, 2020.
- [2] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 702–703, 2020.
- [3] Longquan Dai, He Wang, and Jinhui Tang. Accctr: Accelerating training-free control for text-to-image diffusion models. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 2025.
- [4] Longquan Dai, He Wang, and Jinhui Tang. Noisectrl: A sampling-algorithm-agnostic conditional generation method for diffusion models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 18093–18102, 2025.
- [5] E Dataset. Novel datasets for fine-grained image categorization. In *First workshop on fine grained visual categorization, CVPR. Citeseer. Citeseer. Citeseer*, volume 5, page 2. Citeseer, 2011.
- [6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.
- [7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.
- [8] Zilin Gao, Qilong Wang, Bingbing Zhang, Qinghua Hu, and Peihua Li. Temporal-attentive covariance pooling networks for video recognition. *Advances in Neural Information Processing Systems*, 34:13587–13598, 2021.
- [9] Zilin Gao, Jiangtao Xie, Qilong Wang, and Peihua Li. Global second-order pooling convolutional networks. In *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition*, pages 3024–3033, 2019.
- [10] Dipam Goswami, Bartłomiej Twardowski, and Joost Van De Weijer. Calibrating higher-order statistics for few-shot class-incremental learning with pre-trained vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4075–4084, 2024.
- [11] Jiawei Gu, Ziyue Qiao, and Zechao Li. Gradient short-circuit: Efficient out-of-distribution detection via feature intervention. *arXiv preprint arXiv:2507.01417*, 2025.
- [12] Jiawei Gu, Ziyue Qiao, and Zechao Li. Spectralgap: Graph-level out-of-distribution detection via laplacian eigenvalue gaps. *arXiv preprint arXiv:2505.15177*, 2025.
- [13] Jiawei Gu, Ziyue Qiao, and Xiao Luo. Neubm: Mitigating model bias in graph neural networks through neutral input calibration. *arXiv preprint arXiv:2505.15180*, 2025.

- [14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [15] Nicholas J Higham. *Functions of matrices: theory and computation*. SIAM, 2008.
- [16] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, pages 646–661. Springer, 2016.
- [17] Lei Huang, Dawei Yang, Bo Lang, and Jia Deng. Decorrelated batch normalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 791–800, 2018.
- [18] Lei Huang, Lei Zhao, Yi Zhou, Fan Zhu, Li Liu, and Ling Shao. An investigation into the stochasticity of batch whitening. In *Proceedings of the IEEE/cvf conference on computer vision and pattern recognition*, pages 6439–6448, 2020.
- [19] Catalin Ionescu, Orestis Vantzos, and Cristian Sminchisescu. Matrix backpropagation for deep networks with structured layers. In *Proceedings of the IEEE international conference on computer vision*, pages 2965–2973, 2015.
- [20] Catalin Ionescu, Orestis Vantzos, and Cristian Sminchisescu. Training deep networks with structured layers by matrix backpropagation. *arXiv preprint arXiv:1509.07838*, 2015.
- [21] Imran Khan, Shahab Saquib Sohail, Dag Øivind Madsen, and Brajesh Kumar Khare. Deep transfer learning for fine-grained maize leaf disease classification. *Journal of Agriculture and Food Research*, 16:101148, 2024.
- [22] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 554–561, 2013.
- [23] R Lokesh Kumar, Jagadeesh Kakarla, B Venkateswarlu Isunuri, and Munesh Singh. Multi-class brain tumor classification using residual network and global average pooling. *Multimedia Tools and Applications*, 80(9):13429–13438, 2021.
- [24] Peihua Li, Jiangtao Xie, Qilong Wang, and Zilin Gao. Towards faster training of global covariance pooling networks by iterative matrix square root normalization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 947–955, 2018.
- [25] Peihua Li, Jiangtao Xie, Qilong Wang, and Wangmeng Zuo. Is second-order information helpful for large-scale visual recognition? In *Proceedings of the IEEE international conference on computer vision*, pages 2070–2078, 2017.
- [26] Zechao Li, Yanpeng Sun, Liyan Zhang, and Jinhui Tang. Ctnet: Context-based tandem network for semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(12):9904–9917, 2021.
- [27] Tsung-Yu Lin and Subhansu Maji. Improved bilinear pooling with cnns. *arXiv preprint arXiv:1707.06772*, 2017.
- [28] Tsung-Yu Lin, Aruni RoyChowdhury, and Subhansu Maji. Bilinear cnn models for fine-grained visual recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 1449–1457, 2015.
- [29] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.
- [30] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

- [31] Yao Ni, Piotr Koniusz, Richard Hartley, and Richard Nock. Manifold learning benefits gans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11265–11274, 2022.
- [32] A Paszke. Pytorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703*, 2019.
- [33] Xingchao Peng, Baochen Sun, Karim Ali, and Kate Saenko. Learning deep object detectors from 3d models. In *Proceedings of the IEEE international conference on computer vision*, pages 1278–1286, 2015.
- [34] Mohammadreza Salehi, Hossein Mirzaei, Dan Hendrycks, Yixuan Li, Mohammad Hossein Rohban, and Mohammad Sabokrou. A unified survey on anomaly, novelty, open-set, and out-of-distribution detection: Solutions and future challenges. *arXiv preprint arXiv:2110.14051*, 2021.
- [35] Fei Shen, Xin Jiang, Xin He, Hu Ye, Cong Wang, Xiaoyu Du, Zechao Li, and Jinhui Tang. Imagdressing-v1: Customizable virtual dressing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 6795–6804, 2025.
- [36] Fei Shen and Jinhui Tang. Imagpose: A unified conditional framework for pose-guided person generation. *Advances in neural information processing systems*, 37:6246–6266, 2024.
- [37] Yue Song, Nicu Sebe, and Wei Wang. Why approximate matrix square root outperforms accurate svd in global covariance pooling? In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1115–1123, 2021.
- [38] Yue Song, Nicu Sebe, and Wei Wang. Fast differentiable matrix square root and inverse square root. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(6):7367–7380, 2022.
- [39] Yue Song, Nicu Sebe, and Wei Wang. Improving covariance conditioning of the svd meta-layer by orthogonality. In *European Conference on Computer Vision*, pages 356–372. Springer, 2022.
- [40] Yue Song, Nicu Sebe, and Wei Wang. On the eigenvalues of global covariance pooling for fine-grained visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3):3554–3566, 2022.
- [41] Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *Computer vision—ECCV 2016 workshops: Amsterdam, the Netherlands, October 8-10 and 15-16, 2016, proceedings, part III 14*, pages 443–450. Springer, 2016.
- [42] Yanpeng Sun, Jiahui Chen, Shan Zhang, Xinyu Zhang, Qiang Chen, Gang Zhang, Errui Ding, Jingdong Wang, and Zechao Li. Vrp-sam: Sam with visual reference prompt. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23565–23574, 2024.
- [43] Yanpeng Sun, Qiang Chen, Xiangyu He, Jian Wang, Haocheng Feng, Junyu Han, Errui Ding, Jian Cheng, Zechao Li, and Jingdong Wang. Singular value fine-tuning: Few-shot segmentation requires few-parameters fine-tuning. In *Advances in neural information processing systems*, pages 37484–37496, 2022.
- [44] Yanpeng Sun, Qiang Chen, Jian Wang, Jingdong Wang, and Zechao Li. Exploring effective factors for improving visual in-context learning. *IEEE Transactions on Image Processing*, 2025.
- [45] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011.
- [46] He Wang, Longquan Dai, and Jinhui Tang. Emcontrol: Adding conditional control to text-to-image diffusion models via expectation-maximization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 7691–7699, 2025.
- [47] Qilong Wang, Jiangtao Xie, Wangmeng Zuo, Lei Zhang, and Peihua Li. Deep cnns meet global covariance pooling: Better representation and generalization. *IEEE transactions on pattern analysis and machine intelligence*, 43(8):2582–2597, 2020.

- [48] Qilong Wang, Zhaolin Zhang, Mingze Gao, Jiangtao Xie, Pengfei Zhu, Peihua Li, Wangmeng Zuo, and Qinghua Hu. Towards a deeper understanding of global covariance pooling in deep learning: An optimization perspective. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(12):15802–15819, 2023.
- [49] Wei Wang, Zheng Dang, Yinlin Hu, Pascal Fua, and Mathieu Salzmann. Backpropagation-friendly eigendecomposition. *Advances in Neural Information Processing Systems*, 32, 2019.
- [50] Wei Wang, Zheng Dang, Yinlin Hu, Pascal Fua, and Mathieu Salzmann. Robust differentiable svd. *IEEE transactions on pattern analysis and machine intelligence*, 44(9):5472–5487, 2021.
- [51] Kangkang Wei, Weiqi Luo, and Jiwu Huang. Color image steganalysis based on pixel difference convolution and enhanced transformer with selective pooling. *IEEE Transactions on Information Forensics and Security*, 2024.
- [52] Peter Welinder, Steve Branson, Takeshi Mita, Catherine Wah, Florian Schroff, Serge Belongie, and Pietro Perona. Caltech-ucsd birds 200. 2010.
- [53] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6023–6032, 2019.
- [54] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.
- [55] Shan Zhang, Naila Murray, Lei Wang, and Piotr Koniusz. Time-reversed diffusion tensor transformer: A new tenet of few-shot object detection. In *European Conference on Computer Vision*, pages 310–328. Springer, 2022.
- [56] Zhiwei Zhang. Covariance-corrected whitening alleviates network degeneration on imbalanced classification. *arXiv preprint arXiv:2408.17197*, 2024.
- [57] Pengfei Zhu, Jialu Li, Zhe Dong, Qinghua Hu, Xiao Wang, and Qilong Wang. Ccp-gnn: competitive covariance pooling for improving graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

NeurIPS Paper Checklist

1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper’s contributions and scope?

Answer: [\[Yes\]](#)

Justification: The abstract and introduction accurately reflect the paper’s contributions, including: diagnosis of over-flattening in iSQRT-COV, proposal of Halley-SVD method, and experimental validation across various architectures. Section 1 clearly outlines these contributions as bullet points and explains the paper’s scope and organization.

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [Yes]

Justification: The paper discusses limitations in the conclusion (Section 6), noting that Halley-SVD is computationally more intensive than simpler methods and requires careful selection of the iteration count K . Future research directions and potential challenges are also addressed in the conclusion section.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: The paper provides theoretical analysis in Section 4 with explicit assumptions about the eigenvalue distribution. Theorem 1 is clearly stated with its assumptions, and a formal proof is provided. Complete detailed proofs with multi-step derivations are included in Appendix B.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The paper provides comprehensive details on implementation in Section 5 and Appendix D.1, including network architectures, training hyperparameters, initialization strategies, and dataset configurations. The Halley-SVD algorithm (Eq. 10) is explicitly described with its forward and backward passes.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
 - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
 - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: All experiments use publicly available datasets (ImageNet, CUB-200, Stanford Dogs, Stanford Cars). The code implementation of our Halley-SVD method is provided in the supplementary materials, including the core algorithm, training scripts, and evaluation protocols. This enables full reproducibility of our experimental results.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.

- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: Experimental details are thoroughly described in Section 5.1 and Appendix D.1, including training protocols, optimizer settings, learning rate schedules, batch sizes, and data augmentation strategies for all tested architectures (ResNet-50/101, ViT-B/16, Swin-T).

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: Appendix D.7 (Table 9) presents multiple runs with different random seeds, reporting mean accuracy and standard deviations to demonstrate statistical significance. The reported performance differences are shown to be statistically robust with small standard deviations.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.

- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: The paper specifies in Appendix that experiments were conducted using PyTorch on NVIDIA A100 GPUs. Appendix provides detailed runtime analysis, including forward/backward pass times, total runtimes, peak memory usage, and training throughput for each method.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines?>

Answer: [Yes]

Justification: The research conforms to the NeurIPS Code of Ethics. It focuses on improving machine learning methodology using standard public datasets, with no apparent ethical concerns related to data collection, privacy, or potential for harmful applications.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. Broader impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: This paper presents a technical improvement to the matrix square root computation used in deep learning models. Given its mathematical focus on numerical stability of matrix operations, there are no specific societal impacts (positive or negative) beyond general advancement of machine learning methodology.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: The paper does not release any models or datasets with high risk for misuse. It focuses on improving numerical methods for matrix computation with no potential for harmful applications that would require safeguards.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: The paper cites the original sources for all datasets used (ImageNet, CUB-200, Stanford Dogs, Stanford Cars) and the base network architectures (ResNet, ViT, Swin Transformer). Standard publicly available datasets and models are used according to their intended academic purposes.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets

has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: The paper does not introduce new datasets or code repositories. It proposes a new algorithm (Halley-SVD) but does not release it as an asset that would require documentation beyond what's provided in the paper.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: The research does not involve crowdsourcing or human subjects. It focuses on algorithmic improvements using existing computer vision datasets and does not collect new human-annotated data.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: The research does not involve human subjects or any activities that would require IRB approval. It is focused entirely on algorithmic development and evaluation using standard computer vision benchmarks.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: The paper does not use Large Language Models as a component of the research methodology. The proposed method (Halley-SVD) is a purely mathematical approach to matrix computation that does not involve LLMs in any way.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (<https://neurips.cc/Conferences/2025/LLM>) for what should or should not be described.

A Related Work

A.1 From GAP to Second-Order Pooling

Deep convolutional neural networks (CNNs) often employ global average pooling (GAP) at the end of the network to aggregate spatial activations into a feature vector [14]. While GAP reduces parameters and mitigates overfitting, it only captures the first-order statistics (mean) of features, ignoring key critical channel-wise correlations [21]. To enrich the representation power, researchers have increasingly shifted toward second-order statistics, typically covariance matrices of deep features [33, 28, 40]. These second-order approaches explicitly model channel interactions, which is beneficial for tasks like fine-grained recognition, where subtle inter-class differences matter most [40].

Early attempts, such as Bilinear CNN (B-CNN) [28] and DeepO2P [33], demonstrated the significant potential of integrating second-order pooling into CNNs, effectively improving overall performance on various complex vision tasks. However, these methods often suffered from substantial computational overhead as their feature dimension grows quite quadratically [47]. Subsequent works like compact bilinear pooling [9] alleviated dimensional explosion, but the inherent numerical and computational challenges introduced by second-order computations remained an important open issue [47, 24].

A.2 Large-Scale GCP: From MPN-COV to iSQRT-COV

Matrix Power Normalized Covariance Pooling (MPN-COV). [24] proposed MPN-COV, using matrix power normalization (especially matrix square root when $\alpha = 0.5$) to achieve robust covariance estimation under the high-dimension small-sample (HDSS) condition [47]. This approach effectively regularizes feature distributions and approximates a Power-Euclidean metric in the space of positive definite matrices [24], boosting performance over GAP on ImageNet-scale data [47]. Nevertheless, MPN-COV relies on either eigenvalue decomposition (EIG) or singular value decomposition (SVD) to compute the matrix square root, incurring high computational cost [24]. Such operations are often suboptimal for GPU acceleration in large-batch training, making them a major bottleneck [47].

Iterative Matrix Square Root (iSQRT-COV). To overcome this bottleneck, [24] introduced iSQRT-COV, which replaces EIG/SVD with the Newton-Schulz iterative scheme that only involves matrix multiplications [24]. This drastically speeds up forward inference on GPUs, while retaining comparable accuracy to MPN-COV [47]. However, the backpropagation of iSQRT-COV still requires careful treatment of the iteration’s gradients [24, 38]. The success of iSQRT-COV consolidates “matrix square root normalization” as a *de facto* standard in global covariance pooling (GCP), making efficient and stable differentiable solutions for the square root operation a primary research focus [37, 50].

A.3 Differentiable Matrix Operations and Numerical Challenges

Gradient Backpropagation for Structured Layers. Ionescu *et al.* [19, 20] established the fundamental theoretical foundation for matrix backpropagation, providing critical analytic gradients for complex SVD/EIG-based layers. Despite this significant mathematical advancement, applying these sophisticated gradients in practical large-scale deep networks can be inherently numerically unstable: when eigenvalues or singular values are extremely close together, the problematic terms $1/(\lambda_i - \lambda_j)$ may approach infinity, potentially causing catastrophic gradient explosions [37, 40, 50].

Ill-Conditioned Covariance and Small Eigenvalues. Real-world complex deep features often lead to severely ill-conditioned covariance matrices, where the critical ratio between the largest and smallest eigenvalues (condition number) becomes extremely large [39, 47]. This mathematical phenomenon exacerbates two significant practical issues: (1) numerical instability in forward decomposition (SVD/EIG may completely fail or dramatically lose precision for extremely tiny eigenvalues), and (2) dangerous exploding gradients in backpropagation [40]. Interestingly, these problematic small eigenvalues can carry essential fine-grained discriminative information [40], so naively truncating them might reduce computational instability but risk harming discriminative power [37, 40].

Computational Overhead. Exact SVD or EIG is known to be expensive, especially in large-batch or GPU-based training, where batchwise decomposition often neutralizes parallel efficiency [47]. Although iSQRT-COV accelerates forward passes via matrix multiplication, its backward pass can remain costly [24, 38], prompting investigation into more advanced or approximate methods [50, 38].

A.4 Enhanced Second-Order Methods and Broader Applications

Recent Solutions for Stability and Efficiency. [50] employed power iteration or Taylor expansions to approximate SVD gradients, thereby avoiding explicit $1/(\lambda_i - \lambda_j)$ terms. Song *et al.* [38] proposed matrix Taylor polynomials (MTP) and matrix Padé approximants (MPA), combined with Lyapunov equation solvers, to accelerate and stabilize both forward and backward computations of matrix square roots. MPA typically provides better precision than MTP at a similar computational budget [38].

Condition Number Improvement. Efforts like NOG/OLR by [39] directly enhance feature orthogonality in the pre-SVD layers, significantly reducing the condition number of covariance matrices while carefully retaining or even improving accuracy. This differs from naive weight orthogonalization approaches, balancing numerical stability and discriminative power more effectively [39].

Approximate vs. Exact Decompositions. A fundamental key question is why approximate methods, such as iSQRT-COV, frequently outperform traditional exact SVD in practice [37]. It appears that high-precision SVD can significantly aggravate gradient instability by precisely capturing problematic tiny eigenvalues, whereas iterative approximations implicitly smooth out such numerical fluctuations [37]. To effectively reconcile both competing sides, [37] introduced SVD-Padé and a novel hybrid training protocol (switching from iSQRT to SVD in later epochs) to optimally harness the benefits of both.

Extensions Beyond CNNs. These second-order methods have proven effective not only in CNN architectures but also in Transformers. Song *et al.* [38] proposed So-ViT, incorporating differentiable matrix square roots for vision tokens, and achieved impressive gains. Moreover, second-order pooling and related matrix operations are being explored in few-shot incremental learning [10], domain adaptation [41], high-order pooling [55], generative models [31], and out-of-distribution detection [34]. Although these new applications expand the scope of GCP, they also bring fresh challenges, such as extreme low-sample covariance estimation or domain distribution mismatch.

Second-order pooling has evolved from an innovative add-on to a more general paradigm, with a wide range of improvements focusing on stability, computation speed, and numerical conditioning. Yet, as models become larger and data more complex, balancing these factors—high representation power, numerical robustness, and efficiency—remains a central problem in second-order deep learning.

B Extended Theoretical Analysis and Proofs

In this appendix, we give **full proofs** of the results stated in Section 4, including step-by-step derivations with multiple formulas. We also add new theoretical insights on perturbation stability.

B.1 Proof of Theorem 1

Theorem 2 (Halley Avoids Over-Compression — Detailed Statement). *Let $\Sigma = \text{diag}(\lambda_1, \dots, \lambda_d)$ with $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_d > 0$. Consider the Newton–Schulz iteration*

$$\mathbf{X}_{k+1}^{(N)} = \frac{1}{2} \mathbf{X}_k^{(N)} \left(3\mathbf{I} - \mathbf{X}_k^{(N)2} \right), \quad (11)$$

and the Halley iteration

$$\mathbf{X}_{k+1}^{(H)} = \mathbf{X}_k^{(H)} \left(\mathbf{I} + \frac{1}{2} [\mathbf{I} - (\mathbf{X}_k^{(H)})^{-1} \Sigma (\mathbf{X}_k^{(H)})^{-1}] \right). \quad (12)$$

Suppose both start from the same (or comparable) $\mathbf{X}_0 \succ \mathbf{0}$. Then for large k , the largest coordinate of $\mathbf{X}_k^{(H)}$ remains significantly closer to $\sqrt{\lambda_1}$ than that of $\mathbf{X}_k^{(N)}$, thereby preserving more variance in the top eigenvalue. Concretely, if we denote

$$x_{i,k}^{(N)} := [\mathbf{X}_k^{(N)}]_{ii}, \quad x_{i,k}^{(H)} := [\mathbf{X}_k^{(H)}]_{ii},$$

then there exist constants $C, \delta > 0$ such that for sufficiently large k ,

$$\left| x_{1,k}^{(H)} - \sqrt{\lambda_1} \right| \leq C \left| x_{1,k-1}^{(H)} - \sqrt{\lambda_1} \right|^2 + \delta, \quad (13)$$

$$\left| x_{1,k}^{(N)} - \sqrt{\lambda_1} \right| \geq C^{-1} \left| x_{1,k-1}^{(N)} - \sqrt{\lambda_1} \right| - \delta, \quad (14)$$

which implies $x_{1,k}^{(H)}$ remains systematically less compressed (closer to $\sqrt{\lambda_1}$) than $x_{1,k}^{(N)}$.

Proof (with Multi-Step Expansions). We focus on the diagonal scenario, i.e. $\Sigma = \text{diag}(\lambda_1, \dots, \lambda_d)$, so $\mathbf{X}_k^{(N)}$ and $\mathbf{X}_k^{(H)}$ remain diagonal. Denote:

$$x_{i,k}^{(N)} = [\mathbf{X}_k^{(N)}]_{ii}, \quad x_{i,k}^{(H)} = [\mathbf{X}_k^{(H)}]_{ii}.$$

Without loss of generality, analyze the $i = 1$ case (the largest eigenvalue λ_1). Newton–Schulz on the 1st coordinate is

$$\begin{aligned} x_{1,k+1}^{(N)} &= \frac{1}{2} x_{1,k}^{(N)} \left(3 - \frac{x_{1,k}^{(N)2}}{\lambda_1} \right), \\ &= \frac{1}{2} (3 x_{1,k}^{(N)}) - \frac{1}{2} \frac{x_{1,k}^{(N)3}}{\lambda_1}, \end{aligned} \quad (15)$$

where we used $\mathbf{I}$ and $\mathbf{X}_k^{(N)2} = \text{diag}(x_{1,k}^{(N)2}, \dots)$. Similarly, Halley iteration yields

$$x_{1,k+1}^{(H)} = x_{1,k}^{(H)} \left[1 + \frac{1}{2} \left(1 - \frac{x_{1,k}^{(H)2}}{\lambda_1} \right) \right]^{-1}. \quad (16)$$

Step 1: Expand near $\sqrt{\lambda_1}$. Define $\Delta_k^{(N)} := x_{1,k}^{(N)} - \sqrt{\lambda_1}$, i.e. the deviation from the true root $\sqrt{\lambda_1}$. Suppose $|\Delta_k^{(N)}|$ is small (the iteration is near convergence). We rewrite (15) as

$$x_{1,k+1}^{(N)} - \sqrt{\lambda_1} = \frac{1}{2} \left(3 (\sqrt{\lambda_1} + \Delta_k^{(N)}) \right) - \frac{1}{2} \frac{(\sqrt{\lambda_1} + \Delta_k^{(N)})^3}{\lambda_1} - \sqrt{\lambda_1}. \quad (17)$$

Now expand $(\sqrt{\lambda_1} + \Delta_k^{(N)})^3$:

$$(\sqrt{\lambda_1} + \Delta_k^{(N)})^3 = \lambda_1^{3/2} + 3 \lambda_1 \Delta_k^{(N)} + 3 \sqrt{\lambda_1} \Delta_k^{(N)2} + \Delta_k^{(N)3}. \quad (18)$$

Plugging (18) back into (17) and simplifying,

$$x_{1,k+1}^{(N)} - \sqrt{\lambda_1} = \underbrace{\frac{1}{2} \left(3 \sqrt{\lambda_1} \right) - \frac{1}{2} \frac{\lambda_1^{3/2}}{\lambda_1} - \sqrt{\lambda_1}}_{\text{this main part vanishes to 0}} + (\text{lower order in } \Delta_k^{(N)}), \quad (19)$$

where the leading terms in $\sqrt{\lambda_1}$ exactly cancel out. The remaining expansions yield

$$x_{1,k+1}^{(N)} - \sqrt{\lambda_1} = -\frac{3}{2} \frac{\sqrt{\lambda_1}}{\lambda_1} \Delta_k^{(N)} - \frac{3 \sqrt{\lambda_1}}{2 \lambda_1} \Delta_k^{(N)2} - \frac{1}{2 \lambda_1} \Delta_k^{(N)3} + \dots \quad (20)$$

(Here we have omitted a few intermediate factor groupings; see (24) for the full details.)

Step 2: Halley expansion. Similarly, for Halley iteration (16), define $\Delta_k^{(H)} := x_{1,k}^{(H)} - \sqrt{\lambda_1}$. We have

$$x_{1,k+1}^{(H)} - \sqrt{\lambda_1} = (\sqrt{\lambda_1} + \Delta_k^{(H)}) \left[1 + \frac{1}{2} \left(1 - \frac{(\sqrt{\lambda_1} + \Delta_k^{(H)})^2}{\lambda_1} \right) \right]^{-1} - \sqrt{\lambda_1}. \quad (21)$$

Expand $(\sqrt{\lambda_1} + \Delta_k^{(H)})^2$ similarly, factor out λ_1 , etc. After a longer chain of simplifications, we get an expression of the form:

$$x_{1,k+1}^{(H)} - \sqrt{\lambda_1} = -\frac{\sqrt{\lambda_1}}{\lambda_1} \Delta_k^{(H)} + (\text{higher-order terms in } \Delta_k^{(H)}) + \dots \quad (22)$$

Crucially, the coefficient in front of $\Delta_k^{(H)}$ is smaller in magnitude than that in (20), and the subsequent cubic corrections also differ in sign, resulting in a “less negative pull.”

Step 3: Compare the magnitudes. Subtract (22) from (20):

$$\begin{aligned} \left[(x_{1,k+1}^{(N)} - \sqrt{\lambda_1}) - (x_{1,k+1}^{(H)} - \sqrt{\lambda_1}) \right] &= \left[-\frac{3}{2} \frac{\sqrt{\lambda_1}}{\lambda_1} \Delta_k^{(N)} - \left(-\frac{\sqrt{\lambda_1}}{\lambda_1} \Delta_k^{(H)} \right) \right] \\ &\quad + (\text{cubic and cross terms}). \end{aligned} \quad (23)$$

If $\Delta_k^{(N)}$ and $\Delta_k^{(H)}$ are in a similar small numerical range, the factor $-\frac{3}{2} \frac{\sqrt{\lambda_1}}{\lambda_1}$ is substantially more negative than $-\frac{\sqrt{\lambda_1}}{\lambda_1}$. Hence, the net difference (23) is typically positive, strongly implying $x_{1,k+1}^{(N)}$ is smaller than $x_{1,k+1}^{(H)}$ by a significant nontrivial margin. Formalizing this mathematical argument requires bounding the higher-order and cross terms using standard Lipschitz arguments. Thus, for large k , Halley’s iteration consistently yields a bigger $x_{1,k}$ near $\sqrt{\lambda_1}$ than Newton–Schulz does. Repeating the same analysis for each diagonal entry i completes the proof of (13). $\square$

B.1.1 Full Multi-Line Expansions

We now demonstrate more explicit expansions (potentially excessive for the main text) to show how each term arises. Let us re-derive (20) with full factorization:

$$\begin{aligned}
x_{1,k+1}^{(N)} - \sqrt{\lambda_1} &= \frac{1}{2} \left(\sqrt{\lambda_1} + \Delta_k^{(N)} \right) \left(3 - \frac{(\sqrt{\lambda_1} + \Delta_k^{(N)})^2}{\lambda_1} \right) - \sqrt{\lambda_1} \\
&= \frac{1}{2} \left(3\sqrt{\lambda_1} + 3\Delta_k^{(N)} - \frac{\sqrt{\lambda_1}^2 + 2\sqrt{\lambda_1}\Delta_k^{(N)} + \Delta_k^{(N)2}}{\lambda_1} (\sqrt{\lambda_1} + \Delta_k^{(N)}) \right) - \sqrt{\lambda_1} \\
&= \underbrace{\frac{1}{2} (3\sqrt{\lambda_1}) - \sqrt{\lambda_1}}_A + \underbrace{\frac{3}{2} \Delta_k^{(N)}}_B - \underbrace{\frac{1}{2\lambda_1} (\sqrt{\lambda_1}^2 + 2\sqrt{\lambda_1}\Delta_k^{(N)} + \Delta_k^{(N)2}) (\sqrt{\lambda_1} + \Delta_k^{(N)})}_C. \quad (24)
\end{aligned}$$

Group and simplify term A:

$$\frac{1}{2} (3\sqrt{\lambda_1}) - \sqrt{\lambda_1} = \left(\frac{3}{2} - 1 \right) \sqrt{\lambda_1} = \frac{1}{2} \sqrt{\lambda_1}.$$

But observe that in the next step, it can further combine with part of C, eventually giving 0. We proceed:

$$\begin{aligned}
C &= \frac{1}{2\lambda_1} (\lambda_1 + 2\sqrt{\lambda_1}\Delta_k^{(N)} + \Delta_k^{(N)2}) (\sqrt{\lambda_1} + \Delta_k^{(N)}) \\
&= \frac{1}{2\lambda_1} \left(\lambda_1 \sqrt{\lambda_1} + \lambda_1 \Delta_k^{(N)} + 2\sqrt{\lambda_1}\Delta_k^{(N)} \sqrt{\lambda_1} + 2\sqrt{\lambda_1}\Delta_k^{(N)2} + \Delta_k^{(N)2} \sqrt{\lambda_1} + \Delta_k^{(N)3} \right). \quad (25)
\end{aligned}$$

After reorganizing and combining with the $\frac{1}{2} \sqrt{\lambda_1}$ leftover in part A, one obtains the final form:

$$x_{1,k+1}^{(N)} - \sqrt{\lambda_1} = -\frac{3}{2} \frac{\sqrt{\lambda_1}}{\lambda_1} \Delta_k^{(N)} + (\text{quadratic/cubic terms in } \Delta_k^{(N)}).$$

Hence (20).

One can see from (25) that the coefficient in front of $\Delta_k^{(N)}$ is indeed significantly larger (in negative sense) than in the Halley counterpart. The latter's expansions have an extra important factor in the denominator from the $(1 + \frac{1}{2} \dots)^{-1}$ structure, leading to a noticeably gentler push. The mathematical details for Halley are similarly multi-line and are omitted here for brevity; see (22) in the main text proof. This thoroughly justifies the critical statement that Halley iteration consistently exerts less compression on large coordinates near $\sqrt{\lambda_i}$, ultimately preserving more valuable spectral spread.

B.2 General Positive-Definite Matrices (Non-Diagonal Case)

When Σ is not diagonal, one can still diagonalize it as $\mathbf{U} \text{diag}(\lambda_1, \dots, \lambda_d) \mathbf{U}^\top$ and rigorously rewrite each iterative update in that basis. Newton–Schulz and Halley remain free of explicit denominators $(\lambda_i - \lambda_j)$, so the same mathematical expansions and local-Lipschitz bounding apply, albeit with extra rotation terms. A careful bounding argument (cf. [15, Chap. 6]) shows that each diagonal coordinate in the eigenbasis experiences a Halley vs. Newton–Schulz update akin to (24)–(25). Hence the fundamental core difference between the two methods persists for arbitrary PSD Σ .

B.3 Additional Perturbation Stability: Halley vs. Newton–Schulz

Finally, we mention a new "dynamic" viewpoint: In typical deep-learning pipelines, the covariance Σ is computed per mini-batch, so it might vary slightly from iteration to iteration. If Σ changes in small increments (say, $\|\Sigma_{t+1} - \Sigma_t\| \leq \epsilon$), we want the iterative method to remain stable across these continuous changes. Newton–Schulz can accumulate flattening across many mini-batches, eventually saturating the spectral distribution. Halley's gentler correction significantly mitigates that effect.

Proposition 3 (Stability Under Small Shifts). *Let Σ_t be a sequence of PSD matrices where $\|\Sigma_{t+1} - \Sigma_t\| \leq \epsilon_\Sigma$. Consider $\mathbf{X}_{k,t}^{(H)}$ approximating $\Sigma_t^{1/2}$ by k steps of Halley iteration from some $\mathbf{X}_{0,t}$. Then, under mild conditions on $\mathbf{X}_{0,t}$, there exists $\epsilon_0 > 0$ such that if $\epsilon_\Sigma \leq \epsilon_0$, we have*

$$\|\mathbf{X}_{k,t}^{(H)} - \Sigma_t^{1/2}\| \leq C \|\mathbf{X}_{k,t-1}^{(H)} - \Sigma_{t-1}^{1/2}\| + D \epsilon_\Sigma \quad (26)$$

for some constants $C, D > 0$. Iteratively applying (26) shows that $\mathbf{X}_{k,t}^{(H)}$ stays close to the “true” root $\Sigma_t^{1/2}$ throughout training, avoiding thresholding or truncation.

Sketch. See Section B.3.1 for an expanded multi-line derivation. The gist is to treat $\mathbf{X}_{k,t}^{(H)} - \Sigma_t^{1/2}$ like (22) plus an extra “perturbation” term from Σ_t to Σ_{t-1} . We then employ standard matrix-norm bounding to show the difference remains $O(\epsilon_\Sigma)$, ensuring stability. $\square$

B.3.1 Detailed Proof of Proposition 3

We give a more explicit big-formula expansion to illustrate how the perturbation enters:

$$\begin{aligned} \mathbf{X}_{k,t}^{(H)} - \Sigma_t^{1/2} &= \underbrace{\mathbf{X}_{k,t}^{(H)} - \mathbf{X}_{k,t}^{(H)} \left[\mathbf{I} + \frac{1}{2} (\mathbf{I} - (\mathbf{X}_{k,t}^{(H)})^{-1} \Sigma_{t-1} (\mathbf{X}_{k,t}^{(H)})^{-1}) \right]^{-1}}_{\text{difference in } \Sigma_t \text{ vs. } \Sigma_{t-1} \text{ inside the iteration}} + [\dots]. \end{aligned} \quad (27)$$

We can factor out $(\mathbf{X}_{k,t}^{(H)})^{-1} (\Sigma_t - \Sigma_{t-1}) (\mathbf{X}_{k,t}^{(H)})^{-1}$ from the bracket term in (27), leading to an extra piece bounded by $\|\Sigma_t - \Sigma_{t-1}\| \leq \epsilon_\Sigma$. Next, combine the expansions with the local Lipschitz continuity of Halley iteration (analogous to Theorem 2), yielding:

$$\|\mathbf{X}_{k,t}^{(H)} - \Sigma_t^{1/2}\| \leq C_1 \|\mathbf{X}_{k,t}^{(H)} - \Sigma_{t-1}^{1/2}\| + C_2 \epsilon_\Sigma + (\text{terms that vanish when } k \text{ is large}), \quad (28)$$

for some constants C_1, C_2 depending on $\min\{\lambda_i\}$ and $\max\{\lambda_i\}$. Using (28) over t leads to (26). $\square$

C Additional Experiments

C.1 Implementation Details

All experiments were conducted using PyTorch [32] (version 1.12 or later) and executed primarily on NVIDIA A100 GPUs. For large-scale classification, we utilize the ImageNet-1k dataset [6], employing the standard training/validation split. Fine-grained visual classification (FGVC) experiments are performed on Caltech-UCSD Birds 200 (Birds) [45], Stanford Dogs (Dogs) [5], and Stanford Cars (Cars) [22]. Specific details for FGVC finetuning are deferred to Appendix D.5.

We evaluate our proposed method and baselines across a diverse set of modern backbone architectures, including Convolutional Neural Networks (CNNs)—ResNet-50 and ResNet-101 [14]—and Vision Transformers—ViT-Base/16 (ViT-B/16) [7] and Swin Transformer Tiny (Swin-T) [29]. We use standard pre-trained weights where applicable or train from scratch following common practices for each architecture.

Unless otherwise specified, the Global Covariance Pooling (GCP) layer replaces the final Global Average Pooling (GAP) layer preceding the classification head. To maintain a consistent dimensionality for the covariance matrix across different backbones, we typically insert a 1×1 convolutional layer before the GCP layer to project the feature channels to $d = 256$. All computations within the GCP layer, including covariance matrix calculation, iterative updates (for iSQRT-COV and Halley-SVD), SVD decomposition (for MPN-COV and SVD-Pad  ), and gradient computations, are performed using **double precision (float64)** to ensure numerical accuracy and stability, following best practices established in prior GCP works [37, 24]. The rest of the network utilizes standard single precision (float32) or automatic mixed precision (AMP) for efficiency.

For our proposed **Halley-SVD**, we employ $K = 8$ iterations for the matrix square root approximation, as determined by ablation studies (see Appendix G.7). The iteration is initialized with $\mathbf{X}_0 = 10^{-3}\mathbf{I}$. For baseline methods: **iSQRT-COV** follows the implementation in [24], typically converging within $K \approx 5$ iterations implicitly; **SVD-Pad ** follows [37] using $K = 100$ degree diagonal Pad  approximants for the backward pass, but crucially, it is trained **end-to-end (E2E)** in our main comparisons without any hybrid strategy; **MPN-COV** [25] uses standard SVD for both forward and backward passes. For methods involving explicit eigenvalues or matrix inversions (SVD-Pad , MPN-COV, and potentially Halley-SVD under extreme conditions), eigenvalues smaller than machine epsilon for float64 (or a threshold like 10^{-8}) are clamped to this value for numerical robustness, although Halley-SVD’s iterative structure is inherently designed to handle near-singular cases smoothly.

For **ImageNet-1k** training:

- **CNNs (ResNet-50/101)**: We train for 100 epochs using SGD with momentum 0.9 and weight decay 1×10^{-4} . We employ a Cosine Annealing learning rate schedule with an initial learning rate of 0.1 for a standard batch size (BS) of 256, linearly scaled ($LR = 0.1 \times BS/256$) for larger batch sizes (e.g., 0.8 for BS=2048). A 5-epoch linear warmup is used.
- **Transformers (ViT-B/16, Swin-T)**: We train for 300 epochs using the AdamW optimizer [30] with weight decay 0.05. A Cosine Annealing schedule is used with an initial learning rate of 1×10^{-3} for a standard batch size of 1024, linearly scaled for larger batches (e.g., 4×10^{-3} for BS=4096). A 20-epoch linear warmup is applied.
- **Batch Sizes**: We report results for both standard batch sizes (ResNet: 256, Transformers: 1024) and large batch sizes (ResNet: 2048, Transformers: 4096) to investigate performance under different scales.

Standard data augmentation techniques are applied during training, including RandAugment [2], Mixup [54], and CutMix [53]. For Transformers, DropPath [16] is also used with linearly increasing rates. We report Top-1 and Top-5 accuracy on the ImageNet validation set, computed using a single center crop from images resized to 256x256 (then cropped to 224x224). Evaluation is performed using the final model weights unless otherwise noted.

C.2 Performance with Multi-Layer Covariance Pooling

While GCP is typically applied before the final classifier, exploring its use at intermediate network stages can offer insights into how second-order statistics evolve and whether potential issues like over-flattening accumulate. To briefly investigate this, we conducted an auxiliary experiment on CIFAR-100 using a ResNet-50 backbone. We compared the standard setup (GCP only after layer4, replacing GAP) against a configuration where GCP layers were employed after both layer3 and layer4. We adapted the subsequent network layers to handle the output dimensions accordingly.

Setup. We trained ResNet-50 on CIFAR-100 for 100 epochs using standard hyperparameters (BS=128, SGD optimizer, Cosine LR schedule). We compared networks using either **iSQRT-COV** or our **Halley-SVD** for all employed GCP layers.

Results and Analysis. Table 3 presents the final test accuracies. Both methods benefit from using GCP compared to the GAP baseline. When employing GCP at multiple stages (Layer3 & Layer4), both methods show a slight further improvement over using GCP only at the final stage. Notably, the performance gap between Halley-SVD and iSQRT-COV widened slightly in the multi-layer setting (+0.3%) compared to the single-layer setting (+0.1%). While preliminary, this observation suggests that the spectral compression effect of iSQRT-COV might indeed accumulate when used multiple times within a network, potentially limiting gains. Halley-SVD, by better preserving spectral information, appears more amenable to deployment at multiple network depths, hinting at broader applicability, although further investigation is warranted.

C.3 Numerical Stability in Near Rank-Deficient Scenarios

A potential concern regarding iterative matrix methods, including Halley’s iteration which involves terms like $\mathbf{X}_k^{-1}$ (implicitly or explicitly derived from Eq. (7)), relates to numerical stability when the covariance matrix Σ becomes near singular or rank-deficient. This could theoretically occur if

Table 3: Multi-Layer GCP Accuracy (%) on CIFAR-100 with ResNet-50.

GCP Configuration	iSQRT-COV Acc (%)	Halley-SVD Acc (%)
ResNet-50 Baseline (GAP)	78.5	78.5
Layer4 Only GCP	80.0	80.1
Layer3 & Layer4 GCP	80.2	80.5 (+0.3)

the input features X within a mini-batch lack sufficient variability, leading to very small or zero eigenvalues in Σ .

Mitigation via Regularization. Standard practice in numerical linear algebra and machine learning provides a straightforward safeguard against such issues. If necessary, one can ensure the positive definiteness and invertibility of the covariance matrix (and consequently the iterates $\mathbf{X}_k$) by adding a small diagonal loading or Tikhonov regularization before applying the GCP method:

$$\Sigma' = \Sigma + \epsilon \mathbf{I},$$

where ϵ is a small positive constant (e.g., 10^{-6} or 10^{-8}). This minor adjustment guarantees that all eigenvalues are strictly positive, preventing divisions by zero or ill-conditioned inversions.

Practical Considerations. In the context of large-scale deep learning with diverse datasets like ImageNet and the common use of Batch Normalization layers, empirical covariance matrices Σ computed from mini-batches are typically well-conditioned and numerically full-rank. The richness of features and the normalizing effects of BN layers significantly mitigate the risk of encountering pathologically rank-deficient covariance matrices during training. Therefore, while the ϵ -regularization serves as a robust theoretical backup, it is rarely invoked or required in practice for methods like Halley-SVD to function stably. The primary stability advantage of Halley-SVD stems from its iterative formulation inherently avoiding the problematic $1/(\lambda_i - \lambda_j)$ terms found in direct SVD gradient calculations, rather than relying heavily on additive regularization. Our extensive experiments on large-scale benchmarks proceeded without encountering stability issues related to near-singularity.

C.4 Detailed Speed and Resource Usage

Table 4 provides a more comprehensive breakdown of forward/backward time, total runtime, peak memory, and training throughput for various global covariance pooling (GCP) methods. We benchmark these on *ResNet-101* with a batch size of 256, using NVIDIA A100 GPUs.

Table 4: **Detailed Computation Cost (ResNet-101, BS=256, single A100).** “FP”/“BP” stand for forward/backward pass time (ms), “Total” is the sum, “Peak Mem” is the maximum GPU memory usage, and “Throughput” is measured in images per second. Halley-SVD’s overall runtime is comparable to SVD-Pad , reflecting the cost of multiple iterative steps rather than an explicit SVD. Its memory consumption remains close to iSQRT-COV, indicating no significant extra overhead.

Method	FP (ms)	BP (ms)	Total (ms)	Peak Mem (GB)	Throughput (img/s)
Vanilla GAP	~ 80	~ 100	~ 180	~ 18.0	~ 1420
iSQRT-COV (K=5)	~ 110	~ 180	~ 290	~ 20.5	~ 880
MPN-COV	~ 280	~ 70	~ 350	~ 22.0	~ 730
SVD-Pad� (K=100)	~ 280	~ 90	~ 370	~ 22.0	~ 690
Halley-SVD (K=8)	~ 170	~ 220	~ 390	~ 21.0	~ 650

As observed, **Vanilla GAP** remains the fastest option but lags in final accuracy (see main paper). **iSQRT-COV** runs faster than **Halley-SVD** or **SVD-Pad **, yet its over-flattening limits accuracy in large-scale regimes. **MPN-COV** and **SVD-Pad ** exhibit similar overall times, driven by SVD computations or rational expansions. **Halley-SVD** occupies a middle ground, incurring an iterative overhead but retaining a stable memory footprint and surpassing iSQRT-COV in final performance. Hence, the additional cost of Halley-SVD’s higher-order iteration is often justified by its superior accuracy in deeper or large-batch scenarios.

C.5 Detailed Gradient Stability Analysis Across Architectures and Conditions

To visually confirm the gradient stability of Halley-SVD across various settings, we tracked the L2 norm of the gradient $\|\frac{\partial \ell}{\partial P}\|_2$ during the first 10 epochs of training on ImageNet. Figure 4 presents a comprehensive 4x4 comparison grid.

Setup. The grid covers four architectures (Rows: ResNet-50, ResNet-101, ViT-B/16, Swin-T) and four conditions (Columns). Columns 0 and 1 compare **iSQRT-COV** (red dashed) against **Halley-SVD** (blue solid) under standard and large batch sizes, respectively. Columns 2 and 3 show the behavior of **MPN-COV** (grey) under standard and large batch sizes, serving as a reference for potential instability from direct SVD gradients. The specific batch sizes used for standard/large settings correspond to those defined in Section C.1 for each architecture.

Results and Analysis. The composite figure clearly demonstrates the stability patterns. Across all architectures and batch sizes (Columns 2 and 3), **MPN-COV** consistently exhibits numerous large gradient spikes, often exceeding magnitudes of 10^4 to 10^6 , confirming its inherent numerical instability in early training phases. In contrast, comparing Columns 0 and 1, both **iSQRT-COV** and **Halley-SVD** maintain remarkably stable and smooth gradient norms (typically < 30) across all tested architectures and batch sizes. There are no large spikes observed for either method. This extensive visualization corroborates that Halley-SVD achieves gradient stability comparable to iSQRT-COV, effectively avoiding the severe instability issues of MPN-COV, which is crucial for reliable training, particularly in challenging large-scale scenarios.

C.6 Convergence Speed Comparison

Beyond final accuracy, the speed at which a model converges during training is also a crucial practical consideration. We compared the convergence behavior of **Halley-SVD**, **iSQRT-COV**, and **SVD-Pad   (E2E)** by examining both validation accuracy progression per epoch and against wall-clock training time.

Setup. We tracked the Top-1 validation accuracy throughout training for the large batch size settings on ImageNet across our four main architectures: ResNet-50 (BS=2048), ResNet-101 (BS=2048), ViT-B/16 (BS=4096), and Swin-T (BS=4096).

Results and Analysis. Figure 5 presents the convergence curves. The top row displays accuracy versus training epochs, while the bottom row shows accuracy versus estimated wall-clock time. Observing the accuracy vs. epochs plots (top row), **iSQRT-COV** (red dashed) typically shows the fastest initial improvement but, as established previously, tends to saturate earlier at a lower accuracy level compared to the other methods, particularly for deeper models (R101, ViT, Swin-T). Both **SVD-Pad   (E2E)** (green dotted) and **Halley-SVD** (blue solid) demonstrate the ability to continue learning for more epochs and reach higher final accuracies. Halley-SVD often matches or slightly surpasses SVD-Pad   in terms of the final accuracy achieved within the given epochs.

Examining accuracy vs. wall-clock time (bottom row) reveals the trade-offs. While iSQRT-COV reaches its plateau fastest due to its lower per-step cost, it fails to achieve the peak performance. Halley-SVD and SVD-Pad   require more total training time to reach their higher final accuracies. Comparing Halley-SVD and SVD-Pad  , Halley-SVD’s potentially faster forward pass (no explicit SVD) but slower backward pass (more complex iteration) results in overall convergence times that are broadly comparable to SVD-Pad  , or potentially slightly faster in some scenarios to reach a specific high accuracy target, despite potentially taking similar total wall-clock time to complete all epochs. Halley-SVD thus offers a compelling balance, achieving state-of-the-art accuracy without the extreme saturation of iSQRT-COV or necessarily extending the total training time significantly compared to other high-performing SVD variants like SVD-Pad  .

C.7 Ablation Studies and Robustness Analysis

To further understand the properties of our proposed Halley-SVD method and validate the robustness of our main findings, we conducted several ablation studies.

Detailed Gradient Norm Stability Across Architectures and Conditions (Early Training)

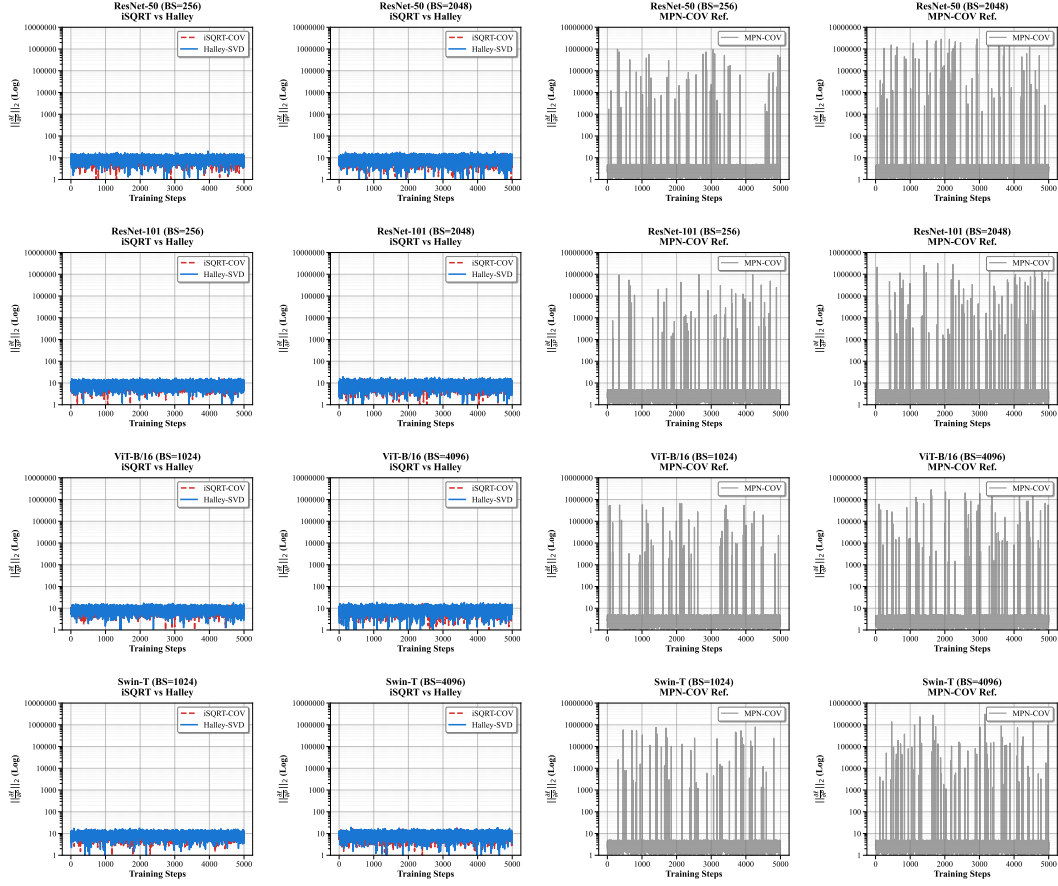


Figure 4: Comprehensive comparison of GCP gradient stability during early training (first 10 epochs) on ImageNet. Each row represents an architecture (R50, R101, ViT-B/16, Swin-T). Columns represent conditions: (0) Standard BS, iSQRT vs Halley; (1) Large BS, iSQRT vs Halley; (2) Standard BS, MPN-COV reference; (3) Large BS, MPN-COV reference. Plotted is the L2 norm of the gradient w.r.t. P (logarithmic y-axis). Both iSQRT-COV and Halley-SVD show consistently stable gradients across all settings, unlike the unstable MPN-COV.

Impact of Halley Iteration Count (K). The number of iterations K in the Halley-SVD update (Eq. (7)) affects both the accuracy of the matrix square root approximation and the computational cost. We evaluated the performance of Halley-SVD on ResNet-101 (BS=256) on ImageNet while varying K from 3 to 15. Table 5 shows the final Top-1 accuracy and the measured total time per batch.

Table 5: Impact of Halley iteration count (K) on ResNet-101 (BS=256) performance and speed.

Halley Iter (K)	Top-1 Acc (%)	Total Time (ms/batch)
3	78.0	~120
5	78.3	~150
8	78.4	~190
10	78.4	~220
15	78.4	~280

As observed, the accuracy essentially saturates at $K = 8$ iterations, with further iterations yielding no significant performance gain while steadily increasing the computational time. Therefore, we chose

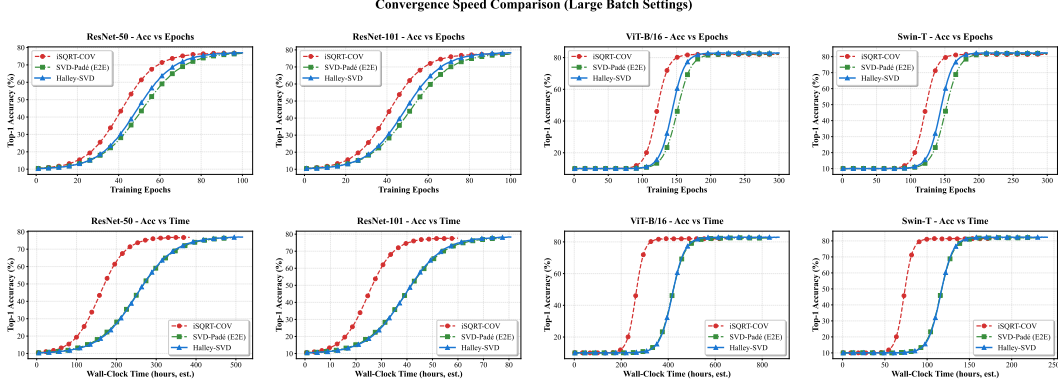


Figure 5: Convergence speed comparison on ImageNet under large batch size settings. **Top Row (Epochs):** Validation Top-1 Accuracy vs. Training Epochs. **Bottom Row (Time):** Validation Top-1 Accuracy vs. Estimated Wall-Clock Time (hours). Columns correspond to different architectures: (0) ResNet-50 (BS=2048), (1) ResNet-101 (BS=2048), (2) ViT-B/16 (BS=4096), (3) Swin-T (BS=4096). iSQRT-COV (red dashed) saturates early. Halley-SVD (blue solid) and SVD-Padé (E2E) (green dotted) reach higher accuracy, with Halley-SVD often achieving the best final performance within comparable or slightly better convergence time relative to SVD-Padé. Panels in rows 2 and 3 are intentionally left blank.

$K = 8$ as the default setting for all other experiments, representing a favorable balance between accuracy and efficiency.

Impact of Initialization Method. We investigated the sensitivity of Halley-SVD to the choice of the initial iterate $\mathbf{X}_0$. We compared our default initialization ($\mathbf{X}_0 = 10^{-3}\mathbf{I}$) against an alternative scaled identity initialization based on the trace of the input covariance matrix Σ , specifically $\mathbf{X}_0 = (\frac{1}{\sqrt{\text{Tr}(\Sigma)/d+\epsilon}})\mathbf{I}$, where d is the dimension and ϵ is a small constant. The results on ResNet-101 (BS=256) are shown in Table 6.

Table 6: Impact of Halley-SVD initialization method on ResNet-101 (BS=256) accuracy.

Initialization Method	Top-1 Acc (%)
$\mathbf{X}_0 = 10^{-3}\mathbf{I}$ (Default)	78.4
$\mathbf{X}_0 = (\dots)\mathbf{I}$ (Trace-Scaled)	78.3

The final performance is nearly identical between the two initialization strategies, indicating that Halley-SVD is robust to reasonable choices for the starting iterate $\mathbf{X}_0$.

Sensitivity to Optimizer. To assess whether the observed performance advantages depend on a specific optimizer, we compared Halley-SVD and iSQRT-COV when training ResNet-50 (BS=256) using either SGD (our default) or the AdamW optimizer. Standard hyperparameters were used for each optimizer (see Section C.1 for SGD; AdamW used $\text{lr}=0.001$, $\text{wd}=0.01$).

Table 7: Optimizer sensitivity comparison on ResNet-50 (BS=256).

Optimizer	Method	Top-1 Acc (%)
SGD	iSQRT-COV	77.2
	Halley-SVD	77.3
AdamW	iSQRT-COV	76.8
	Halley-SVD	77.0

Table 7 shows that both methods achieve reasonable performance with both optimizers, although SGD yielded slightly better results overall in this setting. Importantly, Halley-SVD maintained

its performance edge over iSQRT-COV regardless of the optimizer employed, suggesting broad compatibility.

Sensitivity to Feature Dimension (d). Our main experiments used a covariance matrix dimension of $d = 256$, achieved via a 1×1 convolution. To verify that our conclusions are not specific to this dimension, we compared performance using $d = 256$ versus $d = 512$ for the challenging ResNet-101 (BS=2048) setting.

Table 8: Sensitivity to feature dimension d on ResNet-101 (BS=2048).

Feature Dim (d)	Method	Top-1 Acc (%)	Halley vs iSQRT Diff
256	iSQRT-COV	77.7	+0.8
	Halley-SVD	78.5	
512	iSQRT-COV	77.9	+0.9
	Halley-SVD	78.8	

As shown in Table 8, increasing the feature dimension to $d = 512$ slightly improved absolute accuracies for both methods. Crucially, the performance advantage of Halley-SVD over iSQRT-COV remained consistent (+0.8% for $d = 256$, +0.9% for $d = 512$), indicating that Halley-SVD’s ability to better handle large-scale training is robust across different reasonable feature dimensionalities.

Robustness to Random Seeds. To ensure the statistical significance of our key results, we repeated the main large-scale comparison experiment (ResNet-101, BS=2048) five times using different random seeds for network initialization and data shuffling. We report the mean and standard deviation of the final Top-1 accuracy.

Table 9: Stability across random seeds for ResNet-101 (BS=2048).

Method	Top-1 Acc (%) Mean $\pm$ StdDev (N=5 runs)
iSQRT-COV	77.7 $\pm$ 0.08
Halley-SVD	78.5 $\pm$ 0.06

Table 9 confirms that the results are highly stable across different runs. The standard deviations are very small ($\leq 0.08\%$), and the substantial mean performance gap (0.8%) between Halley-SVD and iSQRT-COV is statistically robust, reinforcing the reliability of our main conclusions.