Spherical Perspective on Learning with Normalization Layers

Simon Roburin
LIGM, Ecole des Ponts, Univ Gustave Eiffel, CNRS, Marne-la-Vallée, France
valeo.ai, Paris, France

Yann de Mont-Marin
Département d'informatique de l'ENS, PSL, Inria, Paris, France

Andrei Bursuc
valeo.ai, Paris, France

Renaud Marlet
LIGM, Ecole des Ponts, Univ Gustave Eiffel, CNRS, Marne-la-Vallée, France
valeo.ai, Paris, France

Patrick Pérez
valeo.ai, Paris, France

Mathieu Aubry
LIGM, Ecole des Ponts, Univ Gustave Eiffel, CNRS, Marne-la-Vallée, France

Abstract

Normalization Layers (NL) are widely used in modern deep-learning architectures. Despite their apparent simplicity, their effect on optimization is not yet fully understood. We introduce a spherical framework to study the optimization of neural networks with NL from a geometric perspective. Concretely, we leverage the radial invariance of groups of parameters to translate the optimization steps on the L_2 unit hypersphere. This formulation and the associated geometric interpretation shed new light on the training dynamics. We use it to derive the first effective learning rate expression of Adam. We then show theoretically and empirically that, in the presence of NL, performing SGD alone is actually equivalent to a variant of Adam constrained to the unit hypersphere.

1. Introduction

The optimization process of deep neural networks is still poorly understood. Their training involves minimizing a high-dimensional non-convex function, which has been proved to be a NP-hard problem [4]. Yet, elementary gradient-based methods show good results in practice. To improve the quality of reached minima, numerous Normalization Layers (NL) have stemmed in the last years and become common practices. One of the most prominent is Batch Normalization (BN) [11], which improves significantly both the the training speed and the prediction performance; it has however a notable shortcoming: BN relies heavily on the batch size. To avoid the dependency on the batch size, normalization layers such as WeightNorm (WN) [18], LayerNorm (LN) [2], InstanceNorm (IN) [23], or GroupNorm (GN) [25] were designed. Yet, the interaction of NLs with optimization remains an open research topic. Previous studies highlighted some of the mechanisms of the interaction between BN and SGD, both empirically [19] and theoretically [1, 3, 10]. But, none of them provided a generic framework and studied the interaction between NL and the most common adaptive schemes for Neural Networks (NNs): Adam [12]. In this work, we provide an extensive analysis of the relation between NL and any order 1 optimization scheme and we theoretically relate SGD with NLs to a variant of Adam (AdaGradG).

A shared effect of all mentioned NL is to make NNs invariant to positive scalings of groups of parameters which may differ from one NL method to another. The core idea of this paper is precisely to focus on these groups of radially-invariant parameters and analyze their optimization projected on the L_2 unit hypersphere (see Figure 1), which is topologically equivalent to the quotient manifold of the parameter space by the scaling action. One could directly optimize parameters on the hypersphere as [6], yet, most optimization methods are still performed successfully in the original parameter space. Here we propose to study an optimization scheme for a given group of radially-invariant parameters through its image scheme on the unit hypersphere. This geometric perspective sheds light on the interaction between NL and Adam, and also outlines an interesting theoretical link between standard SGD and a variant of Adam adapted and constrained to the unit hypersphere: AdamG [6].

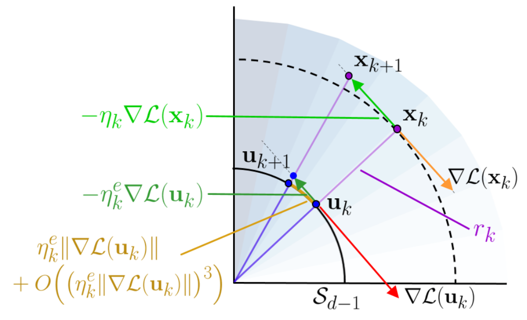
$\mathbf{x}_{k+1}$
$-\eta_k \nabla \mathcal{L}(\mathbf{x}_k)$
$\mathbf{x}_k$
$\mathbf{u}_{k+1}$
$-\eta_k^e \nabla \mathcal{L}(\mathbf{u}_k)$
$\nabla \mathcal{L}(\mathbf{x}_k)$
$\mathbf{u}_k$
r_k
$\eta_k^e \|\nabla \mathcal{L}(\mathbf{u}_k)\|$
$+ O\left(\left(\eta_k^e \|\nabla \mathcal{L}(\mathbf{u}_k)\|\right)^3\right)$
$\mathcal{S}_{d-1}$
$\nabla \mathcal{L}(\mathbf{u}_k)$

Figure 1: **Illustration of the spherical perspective for SGD.** The loss function $\mathcal{L}$ of a NN w.r.t. the parameters $\mathbf{x}_k \in \mathbb{R}^d$ of a neuron followed by a BN is radially invariant. The neuron update $\mathbf{x}_k \to \mathbf{x}_{k+1}$ in the original space, with velocity $\eta_k \nabla \mathcal{L}(\mathbf{x}_k)$, corresponds to an update $\mathbf{u}_k \to \mathbf{u}_{k+1}$ of its projection through an exponential map on the unit hypersphere $\mathcal{S}_{d-1}$ with velocity $\eta_k^e \|\nabla \mathcal{L}(\mathbf{u}_k)\|$ at order 2 (see details in Section 2.3).

The paper is organized as follows. In **Section 2**, we introduce our spherical framework to study the optimization of any radially-invariant model. We also define a generic optimization scheme that encompasses methods such as SGD with momentum (SGD-M) and Adam. We then derive its image step on the unit hypersphere, leading to definitions and expressions of *effective learning rate* and *effective learning direction*. These new definitions are explicit and have a clear interpretation, whereas the definition of [24] is asymptotic and the definitions of [1] and of [10] are variational. In **Section 3**, we leverage the tools of our spherical framework to demonstrate that in presence of NLs, SGD is equivalent to AdaGradG, a combination of AdaGrad [7] (a special case of Adam without momentum) and AdamG [6] (a variant of Adam constrained to the unit hypersphere). In other words, AdaGradG is a variant of Adam without momentum and constrained to the unit hypersphere. In Appendix A, these results are put in perspective with related work.

Our main contributions are the following:

- A framework to analyze/compare order-1 optimization schemes of radially-invariant models;
- The first explicit expression of the effective learning rate for Adam;
- The demonstration that, in the presence of NL, standard SGD is equivalent to AdaGradG, a variant of Adam without momentum constrained to the unit hypersphere;

2. Spherical framework and effective quantities

We provide here background on radial invariance and introduce a generic optimization scheme that encompasses SGD, SGD with momentum (SGD-M) and Adam, with and without L_2-regularization.

Projecting the scheme update on the unit hypersphere leads to the formal definitions of effective learning rate and direction for any order-1 optimization scheme (main notations in Figure 1). This geometric perspective leads to the first explicit expression of the effective learning rate for Adam.

2.1. Radial invariance

We consider a family of functions $\phi_{\mathbf{x}}: \mathbb{R}^{in} \to \mathbb{R}^{out}$ parameterized by radially-invariant parameters $\mathbf{x} \in \mathbb{R}^d \setminus \{\mathbf{0}\}$, i.e., $\forall \rho > 0, \phi_{\rho\mathbf{x}} = \phi_{\mathbf{x}}$ (possible other parameters of $\phi_{\mathbf{x}}$ omitted for clarity), a dataset $\mathcal{D} \subset \mathbb{R}^{in} \times \mathbb{R}^{out}$, a loss function $\ell : \mathbb{R}^{out} \times \mathbb{R}^{out} \to \mathbb{R}$ and a training loss function $\mathcal{L}: \mathbb{R}^d \to \mathbb{R}$ defined as:

$$\mathcal{L}(\mathbf{x}) \overset{\text{def}}{=} \frac{1}{|\mathcal{D}|} \sum_{(\mathbf{s},\mathbf{t}) \in \mathcal{D}} \ell(\phi_{\mathbf{x}}(\mathbf{s}), \mathbf{t}). \tag{1}$$

It verifies: $\forall \rho > 0,\ \mathcal{L}(\rho\mathbf{x}) = \mathcal{L}(\mathbf{x})$. In the context of BN or WN equipped NNs, the group of radially-invariant parameters $\mathbf{x}$ can be the parameters of a single neuron in a linear layer or the parameters of a whole filter in a convolutional layer, followed by the NL (see Appendix B for details, and Appendix C for the application to other NL such as IN, LN or GN).

The quotient of the parameter space by the equivalence relation associated to radial invariance is topologically equivalent to a sphere. We consider here the L_2 sphere $\mathcal{S}_{d-1} = \{\mathbf{u} \in \mathbb{R}^d / \|\mathbf{u}\|_2 = 1\}$ whose canonical metric corresponds to angles: $d_{\mathcal{S}}(\mathbf{u}_1, \mathbf{u}_2) = \arccos(\langle \mathbf{u}_1, \mathbf{u}_2 \rangle)$. This choice of metric is relevant to study NNs since filters in CNNs or neurons in MLPs are applied through scalar product to input data. Besides, normalization in NL is also performed using the L_2 norm.

Our framework relies on the decomposition of vectors into radial and tangential components. During optimization, we write the radially-invariant parameters at step $k \geq 0$ as $\mathbf{x}_k = r_k \mathbf{u}_k$ where $r_k = \|\mathbf{x}_k\|$ and $\mathbf{u}_k = \mathbf{x}_k / \|\mathbf{x}_k\|$. For any quantity $\mathbf{q}_k \in \mathbb{R}^d$ at step k, we write $\mathbf{q}_k^{\perp} = \mathbf{q}_k - \langle \mathbf{q}_k, \mathbf{u}_k \rangle \mathbf{u}_k$ its tangential component relatively to the current direction $\mathbf{u}_k$.

The following lemma states that the gradient of a radially-invariant loss function is tangential and -1 homogeneous:

Lemma 1 (Gradient of a function with radial invariance) *If $\mathcal{L}: \mathbb{R}^d \to \mathbb{R}$ is radially invariant and almost everywhere differentiable, then, for all $\rho > 0$ and all $\mathbf{x} \in \mathbb{R}^d$ where $\mathcal{L}$ is differentiable:*

$$\langle \nabla\mathcal{L}(\mathbf{x}), \mathbf{x} \rangle = 0 \quad \text{and} \quad \nabla\mathcal{L}(\mathbf{x}) = \rho \nabla\mathcal{L}(\rho\mathbf{x}). \tag{2}$$

2.2. Generic optimization scheme

There is a large body of literature on optimization schemes [7, 12, 17, 21, 22]. We focus here on two of the most popular ones, namely SGD and Adam [12]. Yet, to establish general results that may apply to a variety of other schemes, we introduce here a *generic optimization update*:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \eta_k \mathbf{a}_k \oslash \mathbf{b}_k, \tag{3}$$

$$\mathbf{a}_k = \beta \mathbf{a}_{k-1} + \nabla\mathcal{L}(\mathbf{x}_k) + \lambda \mathbf{x}_k, \tag{4}$$

where $\mathbf{x}_k \in \mathbb{R}^d$ is the group of radially-invariant parameters at iteration k, $\mathcal{L}$[1] is the group's loss estimated on a batch of input data, $\mathbf{a}_k \in \mathbb{R}^d$ is a momentum, $\mathbf{b}_k \in \mathbb{R}^d$ is a division vector that can depend on the trajectory $(\mathbf{x}_i, \nabla\mathcal{L}(\mathbf{x}_i))_{i \in [\![0,k]\!]}$, $\eta_k \in \mathbb{R}$ is the scheduled trajectory-independent learning rate, $\oslash$ is the Hadamard element-wise division, β is the momentum parameter, and λ is the L_2-regularization parameter. Appendix D.1 shows how it encompasses known optimization schemes.

[1]To simplify equations $\mathcal{L}_k$ estimated on the k-th batch is noted $\mathcal{L}$.

2.3. Image optimization on the hypersphere

The radial invariance implies that the radial part of the parameter update $\mathbf{x}$ does not change the function $\phi_{\mathbf{x}}$ encoded by the model, nor does it change the loss $\mathcal{L}(\mathbf{x})$. Due to radial invariance, the parameter space projected on the unit hypersphere is topologically closer to the functional space of the network than the full parameter space. Therefore, we wish to analyze the optimization behaviour on the unit hypersphere. To that end, we separate the quantities that can (tangential part) and cannot (radial part) change the model function. Theorem 2 formulates the spherical decomposition of our generic optimization scheme (Eqs. 3-4) in simple terms. It relates the update of radially-invariant parameters in the parameter space $\mathbb{R}^d$ and their update on $\mathcal{S}_{d-1}$ through an exponential map.

Theorem 2 (Image step on $\mathcal{S}_{d-1}$) *Let's consider the update of a group of radially-invariant parameters $\mathbf{x}_k$ at step k following the generic optimization scheme (Eqs. 3-4) and the corresponding update of its projection $\mathbf{u}_k$ on $\mathcal{S}_{d-1}$. Under the hypothesis (H1) and (H2) formalized in D.2.1, the update of $\mathbf{u}_k$ is given by an exponential map at $\mathbf{u}_k$ with velocity $\eta_k^e \mathbf{c}_k^\perp$:*

$$\mathbf{u}_{k+1} = \mathrm{Exp}_{\mathbf{u}_k}\left(-\left[1 + O\left(\|\eta_k^e \mathbf{c}_k^\perp\|^2\right)\right]\eta_k^e \mathbf{c}_k^\perp\right), \tag{5}$$

where $\mathrm{Exp}_{\mathbf{u}_k}$ *is the exponential map on $\mathcal{S}_{d-1}$, and with*

$$\mathbf{c}_k \overset{\text{def}}{=} r_k \mathbf{a}_k \oslash \frac{\mathbf{b}_k}{d^{-1/2}\|\mathbf{b}_k\|}, \tag{6}$$

$$\eta_k^e \overset{\text{def}}{=} \frac{\eta_k}{r_k^2 d^{-1/2}\|\mathbf{b}_k\|}\left(1 - \frac{\eta_k \langle \mathbf{c}_k, \mathbf{u}_k\rangle}{r_k^2 d^{-1/2}\|\mathbf{b}_k\|}\right)^{-1}. \tag{7}$$

More precisely:

$$\mathbf{u}_{k+1} = \frac{\mathbf{u}_k - \eta_k^e \mathbf{c}_k^\perp}{\sqrt{1 + (\eta_k^e \|\mathbf{c}_k^\perp\|)^2}}. \tag{8}$$

The proof is given in D.2.1 and the theorem is illustrated in the case of SGD in Figure 1. Note that for CNN training the hypothesis (H1) and (H2), empirically discussed in the appendix are typically verified. In particular, with typical values $1-\frac{\eta_k\langle \mathbf{c}_k,\mathbf{u}_k\rangle}{r_k^2 d^{-1/2}\|\mathbf{b}_k\|} > 0$ (H1) is true. The other hypothesis $\eta_k^e\|\mathbf{c}_k^\perp\| < \pi$ (H2) where steps are supposed shorter than π is also true (see D.2.2).

2.4. Effective quantities

In Theorem 2, the normalized parameters update in Eq. 5 can be read $\mathbf{u}_{k+1} \approx \mathrm{Exp}_{\mathbf{u}_k}\left(-\eta_k^e \mathbf{c}_k^\perp\right)$, where η_k^e and $\mathbf{c}_k^\perp$ can then be respectively interpreted as the learning rate and the direction of an optimization step constrained to $\mathcal{S}_{d-1}$. Since $\mathbf{a}_k$ is the momentum and, considering Lemma 1, the quantity $r_k \mathbf{a}_k$ in $\mathbf{c}_k$ can be seen as *a momentum on the hypersphere*. Due to radial invariance, only the change of parameter on the unit hypersphere corresponds to a change of model function. Hence we

Table 1: Effective learning rate and direction for optimization schemes with $\nu = rd^{-1/2}\|\mathbf{b}\|$ (we omit here the iteration index k).

Scheme	η^e	$\mathbf{c}^\perp$
SGD	$\frac{\eta}{r^2}$	$\nabla\mathcal{L}(\mathbf{u})$
SGD + L_2	$\frac{\eta}{r^2(1-\eta\lambda)}$	$\nabla\mathcal{L}(\mathbf{u})$
SGD-M	$\frac{\eta}{r^2}(1-\frac{\eta\langle\mathbf{c},\mathbf{u}\rangle}{r^2})^{-1}$	$\mathbf{c}^\perp$
Adam	$\frac{\eta}{r\nu}(1-\frac{\eta\langle\mathbf{c},\mathbf{u}\rangle}{r\nu})^{-1}$	$\mathbf{c}^\perp$

can interpret η_k^e and $\mathbf{c}_k^\perp$ as *effective learning rate* and *effective learning direction*, i.e., the learning rate and direction on the hypersphere that reproduce the function update of the optimization step.

Using Theorem 2, we can derive actual effective learning rates for any optimization scheme that fits our generic framework. These expressions, summarized in Table 1 are explicit and have a clear interpretation, in contrast to learning rates in [24], which are approximate and asymptotic, and in [1, 9], which are variational and restricted to SGD without momentum only.

In particular, we provide the first explicit expression of the effective learning rate for Adam:

$$\eta_k^e = \frac{\eta_k}{r_k \nu_k}\left(1 - \frac{\eta_k \langle \mathbf{c}_k, \mathbf{u}_k \rangle}{r_k \nu_k}\right)^{-1} \tag{9}$$

where $\nu_k = r_k d^{-1/2} \|\mathbf{b}_k\|$ is homogeneous to the norm of a gradient on the hypersphere and can be related to an *second-order moment on the hypersphere* (see D.2.3 for details).

The expression of the effective learning rate of Adam, i.e., the amplitude of the step taken on the hypersphere, reveals a dependence on the dimension d (through ν) of the update of the considered group of radially-invariant parameters. In the case of an MLP or CNN that stacks layers with neurons or filters of different dimensions, the learning rate is thus tuned differently from one layer to another.

For all schemes, the learning rate is tuned by the dynamics of radiuses r_k, which follow:

$$\frac{r_{k+1}}{r_k} = \left(1 - \frac{\eta_k \langle \mathbf{c}_k, \mathbf{u}_k \rangle}{r_k^2 d^{-1/2} \|\mathbf{b}_k\|}\right) \sqrt{1 + (\eta_k^e \|\mathbf{c}_k^\perp\|)^2}. \tag{10}$$

In contrast to [1, 24], this result demonstrates that for momentum methods, $\langle \mathbf{c}_k, \mathbf{u}_k \rangle$, which involves accumulated gradients terms in the momentum as well as L_2 regularization, tunes the learning rate.

3. SGD is equivalent to AdaGrad

We leverage the tools introduced in the spherical framework of Section 2 to find a scheme constrained to the hypersphere that is equivalent to SGD. We show that, for radially-invariant models, SGD is actually a second order moment adaptive optimization method. Formally, SGD is equivalent to a special case of AdamG [6] without momentum, where AdamG is a variant of Adam adapted and constrained to the unit hypersphere. Alternatively, we can also say that SGD is equivalent to a variant of AdaGrad adapted and constrained to the hypersphere, where AdaGrad is a special case of Adam without momentum. We illustrate this theoretical equivalence empirically.

3.1. Equivalence between two optimization schemes

Due to radial invariance, the functional space of the model is encoded in $\mathcal{S}_{d-1}$. Two schemes with the same sequence of groups of radially-invariant parameters on the hypersphere $(\mathbf{u}_k)_{k\geq 0}$ encode the same sequence of model functions. We say that two optimization schemes S and $\tilde{S}$ are equivalent if and only if $\forall k \geq 0, \mathbf{u}_k = \tilde{\mathbf{u}}_k$; starting from the same parameters, they reach the same optimum. By using Eq. 8, we obtain the following lemma for proving the equivalence of two optimization schemes:

Lemma 3 (Sufficient condition for the equivalence of optimization schemes)

$$\begin{cases} \mathbf{u}_0 = \tilde{\mathbf{u}}_0 \\ \forall k \geq 0, \eta_k^e = \tilde{\eta}_k^e, \mathbf{c}_k^\perp = \tilde{\mathbf{c}}_k^\perp \end{cases} \Rightarrow \forall k \geq 0, \mathbf{u}_k = \tilde{\mathbf{u}}_k. \tag{11}$$

3.2. A hypersphere-constrained scheme equivalent to SGD

We now study, within our spherical framework, SGD with L_2 regularization, i.e., the update $\mathbf{x}_{k+1} = \mathbf{x}_k - \eta_k(\nabla\mathcal{L}(\mathbf{x}_k) - \lambda_k \mathbf{x}_k)$. From the effective learning rate expression, we know that SGD yields an adaptive behaviour because it is scheduled by the radius dynamic, which depends on gradients. In fact, the tools in our framework allow us to find that SGD is equivalent to a variant of Adam constrained to the unit hypersphere, similar to AdamG [6], and without momentum, similar to AdaGrad. AdamG [6] uses the same updates as Adam (eq. 21-23) but project the weight on the hyper-sphere after each optimization step. More precisely, SGD is equivalent to AdamG with a null momentum factor $\beta_1 = 0$ (like AdaGrad), a non-null initial second-order moment v_0, an offset of the scalar second-order moment $k+1 \to k$ and without the bias correction term $1 - \beta_2^{k+1}$. Dubbed AdaGradG, this scheme reads:

$$\text{(AdaGradG)} : \begin{cases} \hat{\mathbf{x}}_{k+1} = \mathbf{x}_k - \eta_k \frac{\nabla\mathcal{L}(\mathbf{x}_k)}{\sqrt{v_k}}, \\ \mathbf{x}_{k+1} = \frac{\hat{\mathbf{x}}_{k+1}}{\|\hat{\mathbf{x}}_{k+1}\|}, \\ v_{k+1} = \beta v_k + \|\nabla\mathcal{L}(\mathbf{x}_k)\|^2. \end{cases}$$

AdaGradG, like AdamG, is an adaptive method. Unlike Adam, which is adaptive with respect to the second-order moment for each parameter, AdaGradG and AdamG are adaptive for each group of radially-invariant parameters (e.g., filters for CNNs with BN or WN). In other words, each filter is adapted individually and independently by the optimization algorithm; it is not a global scheduling.

Now if we call « equivalent at order 2 in the step » a scheme equivalence that holds when we use for r_k an expression that satisfies the radius dynamic with a Taylor expansion at order 2, then we have the following theorem:

Theorem 4 (SGD equivalent scheme on the unit hypersphere) *For any $\lambda \geq 0, \eta > 0, r_0 > 0$, we have the following equivalence when using the radius dynamic at order 2 in $(\eta_k\|\nabla\mathcal{L}(\mathbf{u}_k)\|)^2/r_k^2$:*

$$\begin{cases} \text{(SGD)} \\ \mathbf{x}_0 = r_0\mathbf{u}_0 \\ \lambda_k = \lambda \\ \eta_k = \eta \end{cases} \text{is scheme-equivalent at order 2 in step with} \begin{cases} \text{(AdaGradG)} \\ \mathbf{x}_0 = \mathbf{u}_0 \\ \beta = (1-\eta\lambda)^4 \\ \eta_k = (2\beta)^{-1/2} \\ v_0 = r_0^4(2\eta^2\beta^{1/2})^{-1}. \end{cases}$$

Sketch of proof. Starting from SGD, we first use Lemma 3 to find a strict equivalence scheme with a simpler radius dynamic. We resolve this radius dynamic with a Taylor expansion at order 2 in $(\eta_k\|\nabla\mathcal{L}(\mathbf{u}_k)\|)^2/r_k^2$. A second use of Lemma 3 finally leads to the scheme equivalence in Theorem 4. The formal complete proof can be found in D.2.4. The fact that r_k is well approximated at order 2 in practice is illustrated in D.2.5 and Figure 5.

This Theorem is unexpected because SGD, which is not adaptive by itself, is equivalent to a second-order moment adaptive method. The scheduling performed by the radius dynamics actually replicates the effect of dividing the learning rate by the second-order moment of the gradient norm: v_k.

For standard values of hyperparameters $\lambda < 1$ (on the order of 1e-4) and $\eta < 1$ (at most 1e-1), the higher-order terms of the radius in the Taylor expansion empirically become negligible in practice.

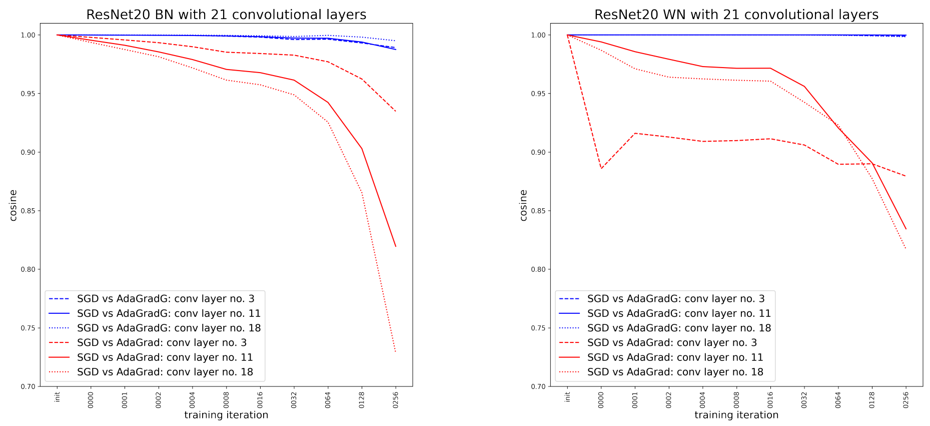

Figure 2: Comparison of the trajectories of radially-invariant parameters using different optimization schemes. For three randomly selected filters in each block of a ResNet20 architecture, with BN (left) or WN (right), we compute the cosine similarity between the parameter values obtained with SGD and the parameters values obtained respectively by AdaGradG and AdaGrad, at different iteration stages of a classification training on CIFAR10.

Second, with standard values of hyperparameters, namely learning rate $\eta < 1$ and L_2 regularization $\lambda < 1$, we have $\beta \leq 1$ which corresponds to a standard value for a moment factor. Interestingly, the L_2-regularization parameter λ controls the memory of the past gradients' norm. If $\beta = 1$ (with $\lambda = 0$), there is no attenuation; each gradient norm has the same contribution in the order-2 moment. If $\lambda \neq 0$, there is a decay factor ($\beta < 1$) on past gradients' norm in the order-2 moment. This gives a new interpretation of the role of L_2-regularization parameter λ in SGD with NL. We illustrate this theoretical result with empirical evidences in Figure 2 with implementation detailed in Appendix D.3. We observe that the trajectories on the L_2 unit hypersphere remain aligned for SGD and AdaGradG whereas, for SGD and AdaGrad, they quickly diverge.

4. Conclusion

The spherical framework introduced in this study provides a powerful tool to analyse order 1 optimization schemes through their projection on the L_2 unit hypersphere. It allows us to give a precise definition and expression of the effective learning rate for Adam and to relate SGD to a variant of Adam. This theoretical finding is corroborated by empirical evidences.

Limitations. Our study only concerns the optimization of radially-invariant parameters. It does not include the impact on other parameters, e.g., for CNNs with BN, scaling and bias in BN layers, and last linear layer.

References

[1] Sanjeev Arora, Zhiyuan Li, and Kaifeng Lyu. Theoretical analysis of auto rate-tuning by batch normalization. In *International Conference on Learning Representations (ICLR)*, 2019.

[2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[3] Nils Bjorck, Carla P Gomes, Bart Selman, and Kilian Q Weinberger. Understanding batch normalization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[4] Avrim Blum and Ronald L Rivest. Training a 3-node neural network is np-complete. In *Advances in Neural Information Processing Systems (NeurIPS)*, 1989.

[5] Yongqiang Cai, Qianxiao Li, and Zuowei Shen. A quantitative analysis of the effect of batch normalization on gradient descent. In *36th International Conference on Machine Learning (ICML)*, 2019.

[6] Minhyung Cho and Jaehyung Lee. Riemannian approach to batch normalization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[7] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research (JMLR)*, 2011.

[8] Behrooz Ghorbani, Shankar Krishnan, and Ying Xiao. An investigation into neural net optimization via hessian eigenvalue density. In *36th International Conference on Machine Learning (ICML)*, 2019.

[9] Elad Hoffer, Ron Banner, Itay Golan, and Daniel Soudry. Norm matters: efficient and accurate normalization schemes in deep networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[10] Elad Hoffer, Itay Hubara, and Daniel Soudry. Fix your classifier: the marginal value of training the last weight layer. In *International Conference on Learning Representations (ICLR)*, 2018.

[11] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *32nd International Conference on Machine Learning (ICML)*, 2015.

[12] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

[13] Zhiyuan Li and Sanjeev Arora. An exponential learning rate schedule for deep learning. In *International Conference on Learning Representations (ICLR)*, 2020.

[14] Xiangru Lian and Ji Liu. Revisit batch normalization: New understanding and refinement via composition optimization. In *The 22nd International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2019.

[15] Weiyang Liu, Yan-Ming Zhang, Xingguo Li, Zhiding Yu, Bo Dai, Tuo Zhao, and Le Song. Deep hyperspherical learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[16] Weiyang Liu, Zhen Liu, Zhiding Yu, Bo Dai, Rongmei Lin, Yisen Wang, James M Rehg, and Le Song. Decoupled networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[17] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

[18] Tim Salimans and Durk P Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Advances in neural information processing systems (NeurIPS)*, 2016.

[19] Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How does batch normalization help optimization? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[20] Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro. The implicit bias of gradient descent on separable data. *The Journal of Machine Learning Research (JMLR)*, 2018.

[21] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *30th International Conference on Machine Learning (ICML)*, Atlanta, Georgia, USA, 2013.

[22] Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 2012.

[23] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016.

[24] Twan van Laarhoven. L2 regularization versus batch and weight normalization, 2017. arXiv preprint arXiv:1706.05350.

[25] Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[26] Guodong Zhang, Chaoqi Wang, Bowen Xu, and Roger Grosse. Three mechanisms of weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

Supplementary Material to "Spherical Perspective on Learning with Normalization Layers"

Appendix A. Related work

Understanding Batch Normalization. Albeit conceptually simple, BN has been shown to have complex implications over optimization. The argument of Internal Covariate Shift reduction [11] has been challenged and shown to be secondary to smoothing of optimization landscape [8, 19] or its modification by creating a different objective function [14], or enabling high learning rates through improved conditioning [3]. [1] demonstrate that (S)GD with BN is robust to the choice of learning rate, with guaranteed asymptotic convergence, while [5] makes a similar finding for GD with BN.

Invariances in neural networks. [6] proposes optimizing over the Grassmann manifold using Riemannian GD. [15] projects weights and activations on the unit hypersphere and compute a function of the angle between them instead of inner products, and subsequently generalizes these operators by scaling the angle [16]. [13] leverages radial invariance to prove that weight decay (WD) can be replaced by an exponential learning-rate scheduling for SGD with or without momentum. [1] investigates radial invariance and shows that radius dynamics depends on past gradients, offering an adaptive behavior to the learning rate. We go further and show that SGD projected on the unit hypersphere corresponds to Adam constrained to the hypersphere, and we give an accurate definition of this adaptive behavior.

Effective learning rate. Due to its scale invariance, BN can adaptively adjust the learning rate [1, 6, 13, 24]. [24] shows that in BN-equipped networks, WD increases the effective learning rate by reducing the norm of the weights. Conversely, without WD, the norm grows unbounded [20], decreasing the effective learning rate. [26] brings additional evidence supporting [24], while [9] finds an exact formulation of the effective learning rate for SGD in normalized networks. In contrast, we a find generic definition of the effective learning rate with exact expressions for SGD and Adam.

Appendix B. Radial invariance of filters with BN

In this section, we show the radial invariance of a set of filters equipped with BN. Please note that the following notations are specific and restricted to this section.

For the sake of simplicity, we only consider the case of a convolutional layer that preserves the spatial extension of the input. We also focus on a single filter. Since all filters act independently on input data, the following calculation holds for any filter.

Let $\mathbf{x} \in \mathbb{R}^{C \times K}$ be the parameters of a single filter, where C is the number of input channels and K is the kernel size. During training, this layer is followed by BN and applied to a batch $\mathbf{s} \in \mathbb{R}^{B \times C \times D}$ of B inputs of spatial size D. The output of the convolution operator ϕ applied to a filter $\mathbf{x} \in \mathbb{R}^{C \times K}$ and to a given batch element $\mathbf{s}_b \in \mathbb{R}^{C \times D}$, with $b \in [\![1, B]\!]$, is thus:

$$\mathbf{t}_b \overset{\text{def}}{=} \phi(\mathbf{x}, \mathbf{s}_b) \in \mathbb{R}^D. \tag{12}$$

The application $(\mathbf{x}, \mathbf{s}_b) \mapsto \phi(\mathbf{x}, \mathbf{s}_b)$ is bilinear. BN then centers and normalizes the output $\mathbf{t}$ using the mean and variance over the batch and the spatial dimension:

$$\mu = \frac{1}{BD} \sum_{b,j} t_{b,j}, \tag{13}$$

$$\sigma^2 = \frac{1}{BD} \sum_{b,j} \left(t_{b,j} - \mu\right)^2, \tag{14}$$

$$\hat{\mathbf{t}}_b \overset{\text{def}}{=} \left(\sigma^2 + \epsilon\right)^{-1/2} \left(\mathbf{t}_b - \mu \mathbf{1}_D\right), \tag{15}$$

where $\mathbf{1}_D$ denotes the all-ones vector of dimension D and ϵ is a small constant.

Now if the coefficients of the filter are rescaled by $\rho > 0$, then, by bilinearity, the new output of the layer for this filter verifies:

$$\tilde{\mathbf{t}}_b = \phi(\rho\mathbf{x}, \mathbf{s}_b) = \rho\phi(\mathbf{x}, \mathbf{s}_b). \tag{16}$$

Since the variance of inputs is generally large in practice, for small ϵ, the mean and variance are:

$$\tilde{\mu} = \rho\mu, \tag{17}$$

$$\tilde{\sigma}^2 \approx \rho^2\sigma^2. \tag{18}$$

It can then be considered that the subsequent BN layer is invariant to this rescaling, *i.e.*, $\hat{\tilde{\mathbf{t}}}_b \approx \hat{\mathbf{t}}_b$.

Appendix C. Extension to other normalization layers

The radial invariance for BN described above in Appendix B applies as well to InstanceNorm (IN) [23] as the normalization is also done with respect to channels but without the batch dimension. Regarding LayerNorm [2] (LN), the normalization is performed over all channels and the entire weight layer can thus be rescaled too, without impacting the output. As for GroupNorm [25] (GN), it associates several channels for normalization; the radial invariance in this case concerns the corresponding group of filters.

Thanks to this general property of radial invariance, the results in this paper not only concern BN but also IN. In fact, they apply as well to LN and GN when considering the suitable group of parameters. The optimization in this case concerns the proper slice of the parameter tensor of the layer, i.e., the whole tensor for LN, and the selected group of filters for GN.

Appendix D. Results in Sections 2 and 3

In this section, we provide proofs and/or empirical results supporting the claims in Sections 2 and 3 of the paper.

In the following, the double parentheses around an equation number, e.g., ((5)), indicate that we recall an equation that was previously stated in the main paper, rather than introduce a new one, e.g., noted (26). Also, framed formulas actually refer to results stated in the main paper, thus with double-bracket equation numbering.

D.1. General optimization schemes

Stochastic gradient descent (SGD) has proven to be an effective optimization method in deep learning. It can include L_2 regularization (also called weight decay) and momentum. Its updates are:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \eta_k \mathbf{m}_k, \tag{19}$$
$$\mathbf{m}_k = \beta \mathbf{m}_{k-1} + \nabla \mathcal{L}(\mathbf{x}_k) + \lambda \mathbf{x}_k, \tag{20}$$

where $\mathbf{m}_k$ is the momentum, β is the momentum parameter, and λ is the L_2-regularization parameter. It corresponds to our generic scheme (Eqs. 3-4) with $\mathbf{a}_k = \mathbf{m}_k$ and $\mathbf{b}_k = [1 \cdots 1]^\top$.

Adam is likely the most common adaptive scheme for NNs. Its updates are:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \eta_k \frac{\mathbf{m}_k}{1-\beta_1^{k+1}} \oslash \sqrt{\frac{\mathbf{v}_k}{1-\beta_2^{k+1}} + \epsilon}, \tag{21}$$
$$\mathbf{m}_k = \beta_1 \mathbf{m}_{k-1} + (1-\beta_1)(\nabla \mathcal{L}(\mathbf{x}_k) + \lambda \mathbf{x}_k), \tag{22}$$
$$\mathbf{v}_k = \beta_2 \mathbf{v}_{k-1} + (1-\beta_2)(\nabla \mathcal{L}(\mathbf{x}_k) + \lambda \mathbf{x}_k)^2_; \tag{23}$$

where $\mathbf{m}_k$ is the momentum with parameter β_1, $\mathbf{v}_k$ is the second-order moment with parameter β_2, and ϵ prevents division by zero. (Here and in the following, the square and the square root of a vector are to be understood as element-wise.) It corresponds to our generic scheme (Eqs. 3-4) with $\beta{=}\beta_1$ and:

$$\mathbf{a}_k = \frac{\mathbf{m}_k}{1-\beta_1}, \tag{24}$$
$$\mathbf{b}_k = \frac{1-\beta_1^{k+1}}{1-\beta_1} \sqrt{\frac{\mathbf{v}_k}{1-\beta_2^{k+1}} + \epsilon}. \tag{25}$$

D.2. Proof of theorems and validity of assumptions

D.2.1. PROOF OF THEOREM 2 (IMAGE STEP ON $\mathcal{S}_{d-1}$) IN SECTION 2.3

We recall the main theorem in Section 2.3.

Theorem 2 (Image step on $\mathcal{S}_{d-1}$) *If the following hypothesis are verified:*

- *(H1):* $1 - \frac{\eta_k \langle \mathbf{c}_k, \mathbf{u}_k \rangle}{r_k^2 d^{-1/2} \|\mathbf{b}_k\|} > 0.$
- *(H2):* $\eta_k^e \|\mathbf{c}_k^\perp\| < \pi.$

The update of a group of radially-invariant parameters $\mathbf{x}_k$ at step k following the generic optimization scheme (Eqs. 3-4) and the corresponding update of its projection $\mathbf{u}_k$ on $\mathcal{S}_{d-1}$ is given by an exponential map at $\mathbf{u}_k$ with velocity $\eta_k^e \mathbf{c}_k^\perp$:

$$\mathbf{u}_{k+1} = \mathrm{Exp}_{\mathbf{u}_k}\left(- \left[1 + O\left(\left(\eta_k^e \|\mathbf{c}_k^\perp\| \right)^2 \right) \right] \eta_k^e \mathbf{c}_k^\perp \right), \tag{(5)}$$

where $\mathrm{Exp}_{\mathbf{u}_k}$ *is the exponential map on* $\mathcal{S}_{d-1}$*, and with*

$$\mathbf{c}_k \overset{\text{def}}{=} r_k \mathbf{a}_k \oslash \frac{\mathbf{b}_k}{d^{-1/2}\|\mathbf{b}_k\|}, \quad \eta_k^e \overset{\text{def}}{=} \frac{\eta_k}{r_k^2 d^{-1/2}\|\mathbf{b}_k\|}\left(1 - \frac{\eta_k\langle \mathbf{c}_k, \mathbf{u}_k\rangle}{r_k^2 d^{-1/2}\|\mathbf{b}_k\|}\right)^{-1}. \tag{(7)}$$

More precisely:

$$\mathbf{u}_{k+1} = \frac{\mathbf{u}_k - \eta_k^e \mathbf{c}_k^\perp}{\sqrt{1 + (\eta_k^e \|\mathbf{c}_k^\perp\|)^2}}. \tag{(8)}$$

Proof To simplify the calculation in the demonstration, we introduce the following notation:

$$A_k \overset{\text{def}}{=} \frac{\eta_k}{r_k^2 d^{-1/2}\|\mathbf{b}_k\|}. \tag{26}$$

We first demonstrate the expression for the radius dynamics in Eq. (10) and the precise step for $\mathbf{u}$ in Eq. (8). Then we use geometric arguments and a Taylor expansion to derive the update on the sphere stated in Eq.(5).

Radius dynamics. We first show Eq. (10), which we recall here using the A_k notation:

$$\boxed{\frac{r_{k+1}}{r_k} = (1 - A_k\langle \mathbf{c}_k, \mathbf{u}_k\rangle)\sqrt{1 + (\eta_k^e\|\mathbf{c}_k^\perp\|)^2}.} \tag{(10)}$$

First, we rewrite the step of a generic scheme in Eqs. (3-4) along the radial and tangential directions and separate the division vector $\mathbf{b}_k$ into its deformation $\frac{\mathbf{b}_k}{d^{-1/2}\|\mathbf{b}_k\|}$ and its scalar scheduling effect $d^{-1/2}\|\mathbf{b}_k\|$, as stated in the discussion:

$$\begin{aligned} r_{k+1}\mathbf{u}_{k+1} &= r_k\mathbf{u}_k - \frac{\eta_k}{d^{-1/2}\|\mathbf{b}_k\|}\mathbf{a}_k \oslash \frac{\mathbf{b}_k}{d^{-1/2}\|\mathbf{b}_k\|} \\ &= r_k\left[\mathbf{u}_k - \frac{\eta_k}{r_k^2 d^{-1/2}\|\mathbf{b}_k\|} r_k\mathbf{a}_k \oslash \frac{\mathbf{b}_k}{d^{-1/2}\|\mathbf{b}_k\|}\right] \\ &= r_k\left[\mathbf{u}_k - A_k r_k \mathbf{a}_k \oslash \frac{\mathbf{b}_k}{d^{-1/2}\|\mathbf{b}_k\|}\right]. \end{aligned} \tag{27}$$

We can note the appearance of a new term $r_k\mathbf{a}_k$. The vector $\mathbf{a}_k$ is a gradient momentum and therefore homogeneous to a gradient. Using Lemma 1, $r_k\mathbf{a}_k$ is homogeneous to a gradient on the hypersphere and can be interpreted as the momentum on the hypersphere.

From Eq. (27), we introduce $\mathbf{c}_k$ (the deformed momentum on hypersphere) as in Eq. (7) and decompose it into the radial and tangential components. We have:

$$\begin{aligned} \frac{r_{k+1}}{r_k}\mathbf{u}_{k+1} &= \mathbf{u}_k - A_k\mathbf{c}_k \\ &= (1 - A_k\langle \mathbf{c}_k, \mathbf{u}_k\rangle)\,\mathbf{u}_k - A_k\mathbf{c}_k^\perp. \end{aligned} \tag{28}$$

By taking the squared norm of the equation, we obtain:

$$\frac{r_{k+1}^2}{r_k^2} = (1 - A_k\langle \mathbf{c}_k, \mathbf{u}_k\rangle)^2 + \left(A_k\|\mathbf{c}_k^\perp\|\right)^2. \tag{29}$$

Making the assumption that $1 - A_k\langle \mathbf{c}_k, \mathbf{u}_k\rangle > 0$, which is true in practice and discussed in the next subsection, we have:

$$\frac{r_{k+1}}{r_k} = (1 - A_k\langle \mathbf{c}_k, \mathbf{u}_k\rangle)\sqrt{1 + \left(\frac{A_k}{1 - A_k\langle \mathbf{c}_k, \mathbf{u}_k\rangle}\|\mathbf{c}_k^{\perp}\|\right)^2}. \tag{30}$$

After introducing $\eta_k^e = \frac{A_k}{(1-A_k\langle \mathbf{c}_k, \mathbf{u}_k\rangle)}$ as in Eq. (7), we obtain the result of (10).

Update of normalized parameters. We then show Eq. (8):

$$\boxed{\mathbf{u}_{k+1} = \frac{\mathbf{u}_k - \eta_k^e \mathbf{c}_k^{\perp}}{\sqrt{1 + (\eta_k^e\|\mathbf{c}_k^{\perp}\|)^2}}.} \tag{(8)}$$

Combining the radius dynamics previously calculated with Eq. (28), we have:

$$\begin{aligned}\mathbf{u}_{k+1} =& \frac{(1 - A_k\langle \mathbf{c}_k, \mathbf{u}_k\rangle)\mathbf{u}_k - A_k \mathbf{c}_k^{\perp}}{(1 - A_k\langle \mathbf{c}_k, \mathbf{u}_k\rangle)\sqrt{1 + (\eta_k^e\|\mathbf{c}_k^{\perp}\|)^2}} &(31)\\ =& \frac{\mathbf{u}_k - \frac{A_k}{1-A_k\langle \mathbf{c}_k, \mathbf{u}_k\rangle}\mathbf{c}_k^{\perp}}{\sqrt{1 + (\eta_k^e\|\mathbf{c}_k^{\perp}\|)^2}}. &(32)\end{aligned}$$

Hence the result (8) using the definition of η_k^e.

This result provides a unique decomposition of the generic step as a step in $\text{span}(\mathbf{u}_k, \mathbf{c}_k^{\perp})$ for the normalized filter (Eq. (8)) and as a radius update (Eq. (10)).

We split the rest of the proof of the theorem in three parts.

Distance covered on the sphere. The distance covered on the hypersphere $\mathcal{S}_{d-1}$ by an optimization step is:

$$\text{dist}_{\mathcal{S}_{d-1}}(\mathbf{u}_{k+1}, \mathbf{u}_k) = \arccos(\langle \mathbf{u}_{k+1}, \mathbf{u}_k\rangle). \tag{33}$$

From Eq. (8) and with Lemma 1, we also have:

$$\langle \mathbf{u}_{k+1}, \mathbf{u}_k\rangle = \frac{1}{\sqrt{1 + (\eta_k^e\|\mathbf{c}_k^{\perp}\|)^2}}. \tag{34}$$

Therefore, $\text{dist}_{\mathcal{S}_{d-1}}(\mathbf{u}_{k+1}, \mathbf{u}_k) = \varphi(\eta_k^e\|\mathbf{c}_k^{\perp}\|)$ where $\varphi : z \mapsto \arccos\left(\frac{1}{\sqrt{1+z^2}}\right)$, which is equal to $\arctan$ on $\mathbb{R}_+$. Then a Taylor expansion at order 3 of $\arctan$ yields for $\eta_k^e\|\mathbf{c}_k^{\perp}\|$:

$$\text{dist}_{\mathcal{S}_{d-1}}(\mathbf{u}_{k+1}, \mathbf{u}_k) = \eta_k^e\|\mathbf{c}_k^{\perp}\| + O\left(\left(\eta_k^e\|\mathbf{c}_k^{\perp}\|\right)^3\right). \tag{35}$$

The Taylor expansion validity is discussed in the next subsection.

Exponential map on the sphere. Given a Riemannian manifold $\mathcal{M}$, for a point $\mathbf{u} \in \mathcal{M}$ there exists an open set $\mathcal{O}$ of the tangent space $\mathcal{T}_{\mathbf{u}}\mathcal{M}$ containing $\mathbf{0}$, such that for any tangent vector $\mathbf{w} \in \mathcal{O}$ there is a unique geodesic (a path minimizing the local distance on $\mathcal{M}$ when conserving the tangent velocity) $\gamma : [-1, 1] \to \mathcal{M}$ that is differentiable and such that $\gamma(0) = \mathbf{u}$ and $\gamma'(0) = \mathbf{w}$. Then, the exponential map of $\mathbf{w}$ from $\mathbf{u}$ is defined as $\text{Exp}_{\mathbf{u}}(\mathbf{w}) = \gamma(1)$.

In the case of the manifold $\mathcal{S}_{d-1}$, the geodesics are complete (they are well defined for any point $\mathbf{u} \in \mathcal{S}_{d-1}$ and any velocity $\mathbf{w} \in \mathcal{T}_{\mathbf{u}}\mathcal{S}_{d-1}$) and are the great circles: for any $\mathbf{u} \in \mathcal{S}_{d-1}$ and any $\mathbf{w} \in \mathcal{T}_{\mathbf{u}}\mathcal{S}_{d-1}$, the map $\psi : t \in \mathbb{R} \mapsto \mathrm{Exp}_{\mathbf{u}}(t\mathbf{w}))$ verifies $\psi(\mathbb{R}) = \mathcal{S}_{d-1} \cap \mathrm{span}(\{\mathbf{u}, \mathbf{w}\})$ which is a great circle passing through $\mathbf{u}$ with tangent $\mathbf{w}$. Furthermore, since the circumference of the great circle is 2π, we have that for any $\mathbf{p} \in \mathcal{S}_{d-1} \backslash \{-\mathbf{u}\}$ there is a unique $\mathbf{w}$ verifying $\|\mathbf{w}\| < \pi$ such that $\mathbf{p} = \mathrm{Exp}_{\mathbf{u}}(\mathbf{w})$ and we have:

$$\mathrm{dist}_{\mathcal{S}_{d-1}}(\mathbf{u}, \mathbf{p}) = \|\mathbf{w}\| \text{ and } \langle \mathbf{p}, \mathbf{w} \rangle \geq 0. \tag{36}$$

Optimization step as an exponential map. We will use the previously stated differential geometry properties to prove:

$$\boxed{\mathbf{u}_{k+1} = \mathrm{Exp}_{\mathbf{u}_k}\left(-\left[1 + O\left(\left(\eta_k^e \|\mathbf{c}_k^\perp\|\right)^2\right)\right] \eta_k^e \mathbf{c}_k^\perp\right).} \tag{(5)}$$

For an optimization step we have:

- by construction, $\mathbf{c}_k^\perp \in \mathcal{T}_{\mathbf{u}_k}\mathcal{S}_{d-1}$;
- from Eq. (8), $\mathbf{u}_{k+1} \in \mathcal{S}_{d-1} \cap \mathrm{span}(\{\mathbf{u}_k, \mathbf{c}_k^\perp\})$;
- from Eq. (8), $\langle \mathbf{u}_{k+1}, \mathbf{c}_k^\perp \rangle \leq 0$.

Then, there exists α that verifies $\|\alpha \mathbf{c}_k^\perp\| < \pi$ such that:

$$\mathbf{u}_{k+1} = \mathrm{Exp}_{\mathbf{u}_k}\left(\alpha \mathbf{c}_k^\perp\right). \tag{37}$$

From Eq. (36), because of the inequality $\langle \mathbf{u}_{k+1}, \mathbf{c}_k^\perp \rangle \leq 0$, we have $\alpha < 0$. We also have that $\|\alpha \mathbf{c}_k^\perp\| = \mathrm{dist}_{\mathcal{S}_{d-1}}(\mathbf{u}_{k+1}, \mathbf{u}_k)$. Then, using the distance previously calculated in Eq. (35), we have:

$$|\alpha| \|\mathbf{c}_k^\perp\| = \eta_k^e \|\mathbf{c}_k^\perp\| + O\left(\left(\eta_k^e \|\mathbf{c}_k^\perp\|\right)^3\right), \tag{38}$$

$$|\alpha| = \eta_k^e \left[1 + O\left(\left(\eta_k^e \|\mathbf{c}_k^\perp\|\right)^2\right)\right]. \tag{39}$$

Combining the sign and absolute value of α, we get the final exponential map expression:

$$\mathbf{u}_{k+1} = \mathrm{Exp}_{\mathbf{u}_k}\left(-\left[1 + O\left(\left(\eta_k^e \|\mathbf{c}_k^\perp\|\right)^2\right)\right] \eta_k^e \mathbf{c}_k^\perp\right), \tag{(5)}$$

$$\approx \mathrm{Exp}_{\mathbf{u}_k}\left(-\eta_k^e \mathbf{c}_k^\perp\right). \tag{40}$$

Note that we implicitly assume here that $|\alpha| \|\mathbf{c}_k^\perp\| \approx \eta_k^e \|\mathbf{c}_k^\perp\| < \pi$, which is discussed in the next subsection.

■

D.2.2. ASSUMPTIONS IN THEOREM 2 AND VALIDITY

Sign of $1 - A_k \langle \mathbf{c}_k, \mathbf{u}_k \rangle$. We tracked the maximum of the quantity $A_k \langle \mathbf{c}_k, \mathbf{u}_k \rangle$ for all the filters of a ResNet20 CIFAR trained on CIFAR10 and optimized with SGD-M or Adam (see Appendix **??** for implementation details). As can be seen on Figure 3, this quantity is always small compared to 1, making $1 - A_k \langle \mathbf{c}_k, \mathbf{u}_k \rangle$ always positive in practice. The order of magnitude of this quantity is roughly the same for different architectures and datasets.

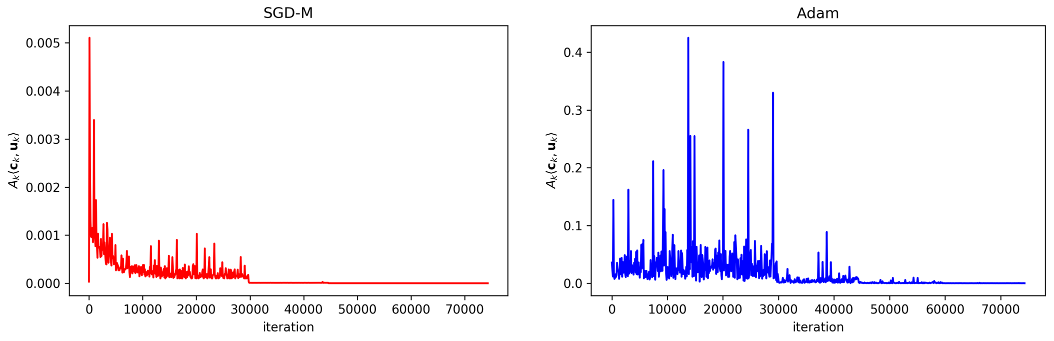

Figure 3: **Tracking of $A_k\langle \mathbf{c}_k, \mathbf{u}_k\rangle$ for SGD-M and Adam.** The above graphs show the maximum of the absolute value of $A_k\langle \mathbf{c}_k, \mathbf{u}_k\rangle$ for all filters in all layers of a ResNet20 CIFAR trained on CIFAR10 and optimized with SGD-M (left) or Adam (right). The quantity is always small compared to 1. Therefore we may assume that $1 - A_k\langle \mathbf{c}_k, \mathbf{u}_k\rangle \geq 0$.

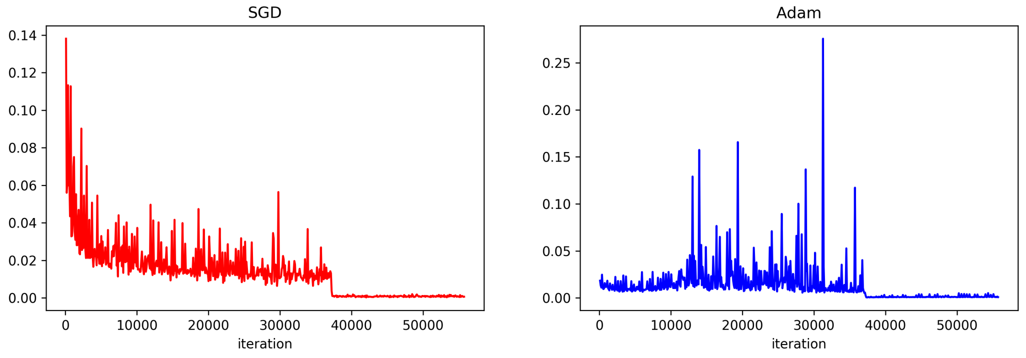

Figure 4: **Tracking of $\eta_k^e\|\mathbf{c}_k^\perp\|$ for SGD-M and Adam.** The above graphs show the maximum of the absolute value of $\eta_k^e\|\mathbf{c}_k^\perp\|$ for all filters in all layers of a ResNet20 CIFAR trained on CIFAR10 and optimized with SGD-M (left) or Adam (right).

Taylor expansion. We tracked the maximum of the quantity $\eta_k^e\|\mathbf{c}_k^\perp\|$ for all the filters of a ResNet20 CIFAR trained on CIFAR10 and optimized with SGD-M or Adam. The observed values justify the Taylor expansion and validate the assumption $|\alpha|\|\mathbf{c}_k^\perp\| \approx \eta_k^e\|\mathbf{c}_k^\perp\| < \pi$. (cf. Figure 4 and Appendix **??** for implementation details). The order of magnitude of this quantity is roughly the same for other different architectures and datasets.

D.2.3. ν_k, ORDER 2 MOMENT ON THE HYPERSPHERE FOR ADAM

Scheduling effect of Adam division vector. In the case of Adam, we recall that:

$$\mathbf{b}_k = \frac{1-\beta_1^{k+1}}{1-\beta_1}\sqrt{\frac{\mathbf{v}_k}{1-\beta_2^{k+1}}+\epsilon}\,. \qquad ((25))$$

Omitting ϵ for simplicity we have:

$$d^{-1/2}\|\mathbf{b}_k\| = \frac{1-\beta_1^{k+1}}{1-\beta_1}\left(\frac{1}{1-\beta_2^{k+1}}\right)^{\frac{1}{2}} d^{-1/2}\|\sqrt{\mathbf{v}_k}\|. \tag{41}$$

Let us calculate $\|\sqrt{\mathbf{v}_k}\|$. Developing the recursion of $\mathbf{v}_k$, as defined in Eq. (23), leads to:

$$\mathbf{v}_k = (1-\beta_2)\sum_{i=0}^{k}\beta_2^{k-i}\left(\nabla\mathcal{L}(\mathbf{x}_i)+\lambda\mathbf{x}_i\right)^2, \tag{42}$$

$$\sqrt{\mathbf{v}_k} = \sqrt{1-\beta_2}\sqrt{\sum_{i=0}^{k}\beta_2^{k-i}\left(\nabla\mathcal{L}(\mathbf{x}_i)+\lambda\mathbf{x}_i\right)^2}, \tag{43}$$

where the square and the square-root are element-wise operations. Hence, if we take the square norm:

$$\begin{aligned}
\|\sqrt{\mathbf{v}_k}\|^2 &= (1-\beta_2)\sum_{j=1}^{d}\left(\sqrt{\sum_{i=0}^{k}\beta_2^{k-i}\left(\nabla\mathcal{L}(\mathbf{x}_i)+\lambda\mathbf{x}_i\right)^2}\right)_j^2 \\
&= (1-\beta_2)\sum_{j=1}^{d}\sum_{i=0}^{k}\beta_2^{k-i}\left(\nabla\mathcal{L}(\mathbf{x}_i)+\lambda\mathbf{x}_i\right)_j^2 \\
&= (1-\beta_2)\sum_{i=0}^{k}\beta_2^{k-i}\sum_{j=1}^{d}\left(\nabla\mathcal{L}(\mathbf{x}_i)+\lambda\mathbf{x}_i\right)_j^2 \\
&= (1-\beta_2)\sum_{i=0}^{k}\beta_2^{k-i}\|\nabla\mathcal{L}(\mathbf{x}_i)+\lambda\mathbf{x}_i\|^2,
\end{aligned} \tag{44}$$

where the j subscript denotes the j-th element of the vector. It is exactly the order-2 moment of the gradient norm.

With Eq. (44) and using Lemma 1, we can give the expression of the second-order moment on the sphere, defined as $\nu_k = r_k d^{-1/2}\|\mathbf{b}_k\|$:

$$\boxed{\nu_k = d^{-1/2}\frac{1-\beta_1^{k+1}}{1-\beta_1}\left(\frac{1-\beta_2}{1-\beta_2^{k+1}}\right)^{1/2}\left(\sum_{i=0}^{k}\beta_2^{k-i}\frac{r_k^2}{r_i^2}\|\nabla\mathcal{L}(\mathbf{u}_i)+\lambda r_i^2\mathbf{u}_i\|^2\right)^{1/2}.} \tag{45}$$

D.2.4. PROOF OF THEOREM 4 (SGD EQUIVALENT SCHEME ON THE UNIT HYPERSPHERE) IN SECTION 3.2

We prove the following theorem:

Theorem 4 (SGD equivalent scheme on the unit hypersphere.) *For any $\lambda > 0, \eta > 0, r_0 > 0$, we have the following equivalence at order 2 in the radius dynamics:*

$$\begin{cases} \text{(SGD)} \\ \mathbf{x}_0 = r_0\mathbf{u}_0 \\ \lambda_k = \lambda \\ \eta_k = \eta \end{cases} \quad \textit{is scheme-equivalent at order 2 to} \quad \begin{cases} \text{(AdamG*)} \\ \mathbf{x}_0 = \mathbf{u}_0 \\ \beta = (1-\eta\lambda)^4 \\ \eta_k = (2\beta)^{-1/2} \\ v_0 = r_0^4(2\eta^2\beta^{1/2})^{-1} \end{cases}$$

Proof As summarized in Table 1, the expressions of the effective learning rates and directions for SGD are $\mathbf{c}_k^\perp = r_k \nabla\mathcal{L}(\mathbf{x}_k) = \nabla\mathcal{L}(\mathbf{u}_k)$ and $\eta_k^e = \frac{\eta_k}{r_k^2(1-\eta_k\lambda_k)}$.

Equivalence with SGD and L_2 regularization. We look for conditions leading to an equivalence between SGD with L_2 regularization and SGD without L_2 regularization. Using Lemma 3, the equality of effective directions is trivial and the equality of effective learning rates for any step k yields the following equivalence:

$$\begin{cases} \text{(SGD)} \\ \tilde{\mathbf{x}}_0 = r_0\mathbf{u}_0 \\ \tilde{\lambda}_k = \lambda \\ \tilde{\eta}_k = \eta \end{cases} \quad \text{is scheme-equivalent to} \quad \begin{cases} \text{(SGD)} \\ \mathbf{x}_0 = r_0\mathbf{u}_0 \\ \lambda_k = 0 \\ \eta_k = \eta(1-\eta\lambda)^{-2k-1} \end{cases} \tag{46}$$

L_2 regularization is equivalent to an exponential scheduling of the learning rate, as found in [13]. Here, we provide a proof in a constructive manner. We are going to use Lemma 3 and find a sufficient condition to have:

$$\begin{cases} \text{(i) } \mathbf{u}_0 = \tilde{\mathbf{u}}_0 \\ \text{(ii) } \forall k \geq 0, \eta_k^e = \tilde{\eta}_k^e, \mathbf{c}_k^\perp = \tilde{\mathbf{c}}_k^\perp. \end{cases}$$

Equation (i) is trivially satisfied by simply taking the same starting point: $\tilde{\mathbf{x}}_0 = \mathbf{x}_0$.

Regarding (ii), because effective directions are the same and only depend on $\mathbf{u}_k$, we only need a sufficient condition on η_k^e. For effective learning rates, using Eq. ((10)) and expressions in Table 1, we have:

$$\eta_k^e = \tilde{\eta}_k^e \Leftrightarrow \frac{\eta_k}{r_k^2} = \frac{\tilde{\eta}_k}{\tilde{r}_k^2(1-\tilde{\eta}_k\lambda)}. \tag{47}$$

Since $\tilde{\eta}_k = \eta$, we obtain:

$$(47) \Leftrightarrow \eta_k = \left(\frac{r_k}{\tilde{r}_k}\right)^2 \frac{\eta}{(1-\eta\lambda)}.$$

Therefore:

$$\frac{\eta_{k+1}}{\eta_k} = \left(\frac{r_{k+1}\tilde{r}_k}{\tilde{r}_{k+1}r_k}\right)^2 = \left(\frac{r_{k+1}/r_k}{\tilde{r}_{k+1}/\tilde{r}_k}\right)^2.$$

By using the radius dynamics in Eq. (10) for the two schemes, SGD and SGD with L_2 regularization, and by the equality of effective learning rates and directions, we have:

$$\begin{aligned} \frac{\eta_{k+1}}{\eta_k} &= \left(\frac{\sqrt{1+(\eta_k^e\|\mathbf{c}_k^\perp\|)^2}}{(1-\eta\lambda)\sqrt{1+(\tilde{\eta}_k^e\|\tilde{\mathbf{c}}_k^\perp\|)^2}}\right)^2 \\ &= (1-\eta\lambda)^{-2}. \end{aligned}$$

By taking Eq. (47) for $k = 0$, because $r_0 = \tilde{r}_0$ we have: $\eta_0 = \eta(1-\eta\lambda)^{-1}$. Combining the previous relation and the initialization case, we derive by induction that $\eta_k = \eta(1-\eta\lambda)^{-2k-1}$ is a sufficient condition. We can conclude, using Lemma 3, the equivalence stated in Eq. (46).

Resolution of the radius dynamics. Without L_2 regularization, the absence of radial component in $\mathbf{c}_k$ makes the radius dynamics simple:

$$r_{k+1}^2 = r_k^2 + \frac{(\eta_k \|\nabla \mathcal{L}(\mathbf{u}_k)\|)^2}{r_k^2}. \tag{48}$$

With a Taylor expansion at order 2, we can show that for $k \geq 1$ the solution

$$r_k^2 = \sqrt{2\sum_{i=0}^{k-1} (\eta_i \|\nabla \mathcal{L}(\mathbf{u}_i)|)^2 + r_0^4}$$

satisfies the previous equation. Indeed using the expression at step $k+1$ gives:

$$\begin{aligned} r_{k+1}^2 &= \sqrt{2\sum_{i=0}^{k-1} (\eta_i \|\nabla \mathcal{L}(\mathbf{u}_i)|)^2 + r_0^4 + 2(\eta_k \|\nabla \mathcal{L}(\mathbf{u}_k)\|)^2} \\ &= r_k^2 \sqrt{1 + 2\frac{(\eta_k \|\nabla \mathcal{L}(\mathbf{u}_k)\|)^2}{r_k^4}} \\ &= r_k^2 \left(1 + (1/2)2\frac{(\eta_k \|\nabla \mathcal{L}(\mathbf{u}_k)\|)^2}{r_k^4} + o\left(\frac{(\eta_k \|\nabla \mathcal{L}(\mathbf{u}_k)\|)^2}{r_k^4}\right)\right) \\ &= r_k^2 + \frac{(\eta_k \|\nabla \mathcal{L}(\mathbf{u}_k)\|)^2}{r_k^2} + o\left(\frac{(\eta_k \|\nabla \mathcal{L}(\mathbf{u}_k)\|)^2}{r_k^2}\right). \end{aligned}$$

Using $\eta_k = \eta(1-\eta\lambda)^{-2k-1}$, introducing $\beta = (1-\eta\lambda)^4$, omitting the $o\left(\frac{(\eta_k \|\nabla \mathcal{L}(\mathbf{u}_k)\|)^2}{r_k^2}\right)$ and injecting the previous solution in the effective learning rate, we obtain the closed form:

$$\begin{aligned} \eta_k^e &= \frac{\eta(1-\eta\lambda)^{-2k-1}}{\sqrt{2\sum_{i=0}^{k-1} \eta^2 (1-\eta\lambda)^{-4i-2} \|\nabla \mathcal{L}(\mathbf{u}_i)\|^2 + r_0^4}} \\ &= \frac{(2\beta)^{-\frac{1}{2}}}{\sqrt{\sum_{i=0}^{k-1} \beta^{(k-1)-i} \|\nabla \mathcal{L}(\mathbf{u}_i)\|^2 + \beta^k \frac{r_0^4}{2\eta^2 \beta^{\frac{1}{2}}}}}. \end{aligned} \tag{49}$$

AdamG*. The AdamG* scheme is constrained on the hypersphere thanks to the normalization; the radius is therefore constant and equal to 1. The absence of radial component in the update gives: $\mathbf{c}_k^\perp = \nabla \mathcal{L}(\mathbf{u}_k)$ and $\eta_k^e = \frac{\eta_k}{\sqrt{v_k}}$. Thus, the resolution of the induction on v_k leads to the the closed form:

$$\eta_k^e = \frac{\eta_k}{\sqrt{\sum_{i=0}^{k-1} \beta^{(k-1)-i} \|\nabla \mathcal{L}(\mathbf{u}_i)\|^2 + \beta^k v_0}}. \tag{50}$$

Hence the final theorem, when identifying the closed-form expressions of effective learning rates and using Lemma 3. ■

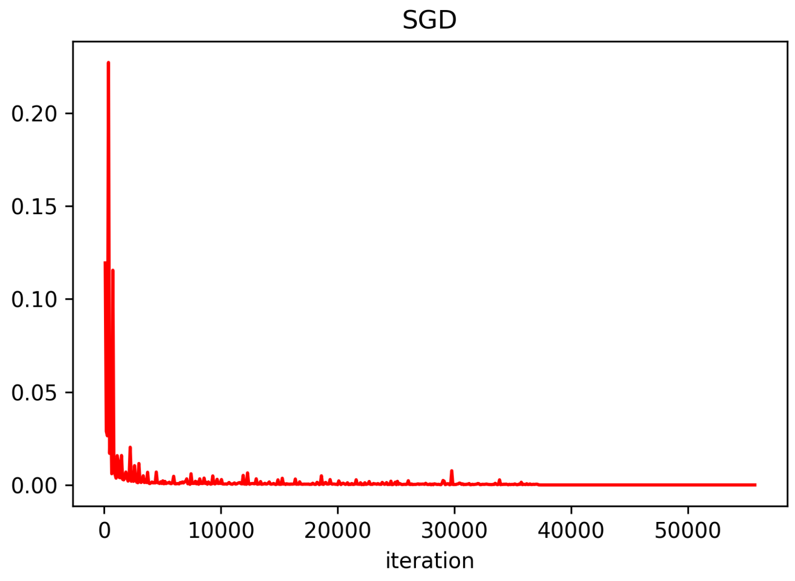

Figure 5: **Validity of Taylor expansion**. We tracked the maximum value of $(\eta_k \|\nabla\mathcal{L}(\mathbf{u}_k)\|)^2/r_k^2$ for all filters in all layers of a ResNet20 CIFAR trained on CIFAR10 with SGD. The order of magnitude of the gradient is roughly the same for other architectures or datasets. It empirically validates the approximation by the Taylor expansion.

D.2.5. VALIDITY OF THE ASSUMPTIONS IN THEOREM 4

Validity of the Taylor expansion. For a CNN trained with SGD optimization, we tracked the quantity $(\eta_k \|\nabla\mathcal{L}(\mathbf{u}_k)\|)^2/r_k^2$, which is the variable of the Taylor expansion. As can be seen in Figure 5, the typical order of magnitude is 10^{-2}, justifying the Taylor expansion (see Appendix **??** for implementation details).

A quick formal analysis also suggests the validity of this hypothesis. Thanks to the expression of $\eta_k = (1-\eta\lambda)^{-2i-k}\eta$ shown in the previous section, if we replace $\|\nabla\mathcal{L}(\mathbf{u}_k)\|$ by a constant for asymptotic analysis, the comparison becomes:

$$(1-\eta\lambda)^{-4k-2} \ll (1-\eta\lambda)^{-2}\frac{1-(1-\eta\lambda)^{-4k}}{1-(1-\eta\lambda)^{-4}} \tag{51}$$

$$1 \ll \frac{1-(1-\eta\lambda)^{4k}}{(1-\eta\lambda)^{-4}-1}. \tag{52}$$

It is asymptotically true.

D.3. Empirical validation of Theorem 4

D.4. Details on Figure 2

In order to illustrate the equivalence in Theorem 4, we experiment with learning an image classifier on CIFAR10 using different optimization schemes. We consider two architectures: ResNet20 with BN and ResNet20 with WN.

The set of parameters $\boldsymbol{\theta}$ of the above architectures can be split into two disjoint subsets: $\boldsymbol{\theta} = \mathcal{F} \cup \mathcal{R}$, where $\mathcal{F}$ is the set of groups of radially-invariant parameters and $\mathcal{R}$ is the set of remaining parameters. Note that the set of remaining parameters in $\mathcal{R}$ differs from one architecture to another: for ResNet20 BN, it includes the last linear layer as well as the scaling and bias of BN layers; for ResNet20 WN, it includes the magnitude parameters in each convolutional layer as well as the last linear layer.
For each architecture, we experiment with tracking the trajectory of parameters under different optimization schemes: SGD, AdaGradG and AdaGrad. As our analysis is restricted to radially-invariant parameters, we only track the trajectory of parameters belonging to $\mathcal{F}$, while the remaining parameters, belonging to $\mathcal{R}$, are always optimized in the same way, i.e., with SGD. For stability purposes, we finetune a previously trained architecture with SGD. The order of batches as well as the random seed for data augmentation are fixed to obtain comparable trajectories. The hyperparameters are chosen for SGD and AdaGrad so that gradient steps have the same order of magnitude (see D.5 for details); the hyperparameters for AdaGradG are provided by the equivalence in Theorem 4.

In Figure 2, we show the angle between the training trajectories using SGD and AdaGradG (resp. SGD and AdaGrad), for three different filters in each block of the ResNet20 architectures. We observe that the trajectories on the L_2 unit hypersphere remain aligned for SGD and AdaGradG whereas, for SGD and AdaGrad, they quickly diverge. It empirically validates the equivalence mentioned in Theorem 4.

D.5. Implementation details

Due to the high non-convexity of the optimization landscape, we choose to start from a relatively stable point in the parameter space. The finetuning of each architecture (ResNet20 BN, ResNet20 BN w/o affine and ResNet WN) starts from previously trained architectures on CIFAR10 via a simple SGD with an initial learning rate of 10^{-1}, a L_2-regularization parameter of 10^{-4} and a momentum parameter of 0.9. The training is performed during 200 epochs, and the learning rate is multiplied by 0.1 at epochs 80, 120 and 160.

Then we track the trajectory obtained with SGD, AdaGrad and AdaGradG. The effective learning rate for SGD is fixed to 10^{-2} and the L_2-regularization parameter is set to 10^{-3} during finetuning. It gives us the following equivalent parameters for AdaGradG: order-2 moment parameter $\beta \approx 0.99996$ and learning rate $\eta \approx 0.71$. Since the effective direction is the same for both SGD and AdaGrad (Adam without momentum), in order to have the same order of magnitude for the gradient steps we need to have effective learning rates of same order of magnitude. From Table 1, in the case of SGD we have $\eta^e_{k\,\text{SGD}} = \frac{\eta_k}{r_k^2}$, and in the case of AdaGrad we have $\eta^e_{k\,\text{AdaGrad}} = \frac{\eta_k}{r_k \nu_k} = \frac{\eta_k}{r_k^2 d^{-1/2} \|\mathbf{b}_k\|} = \eta^e_{k\,\text{SGD}} \frac{1}{d^{-1/2}\|\mathbf{b}_k\|}$. We track the quantity $\frac{1}{d^{-1/2}\|\mathbf{b}_k\|}$ during training, which is roughly in the order of magnitude of 10^{-1}. Therefore, to have gradient steps of equivalent order of magnitude between SGD and AdaGrad, we have to choose a learning rate of 10^{-3} for AdaGrad.