Journal of Machine Learning Research 25 (2024) 1-38 Submitted 4/23; Revised 10/24; Published 12/24

Infeasible Deterministic, Stochastic, and Variance-Reduction Algorithms for Optimization under Orthogonality Constraints

Pierre Ablin PIERRE.ABLIN@APPLE.COM
Apple

Simon Vary SIMON.VARY@STATS.OX.AC.UK
Department of Statistics, University of Oxford

Bin Gao GAOBIN@LSEC.CC.AC.CN
ICMSEC/LSEC, AMSS, Chinese Academy of Sciences

P.-A. Absil PA.ABSIL@UCLOUVAIN.BE
ICTEAM/INMA, UCLouvain

Editor: Prateek Jain

Abstract

Orthogonality constraints naturally appear in many machine learning problems, from principal component analysis to robust neural network training. They are usually solved using Riemannian optimization algorithms, which minimize the objective function while enforcing the constraint. However, enforcing the orthogonality constraint can be the most time-consuming operation in such algorithms. Recently, Ablin and Peyré (2022) proposed the landing algorithm, a method with cheap iterations that does not enforce the orthogonality constraints but is attracted towards the manifold in a smooth manner. This article provides new practical and theoretical developments for the landing algorithm. First, the method is extended to the Stiefel manifold, the set of rectangular orthogonal matrices. We also consider stochastic and variance reduction algorithms when the cost function is an average of many functions. We demonstrate that all these methods have the same rate of convergence as their Riemannian counterparts that exactly enforce the constraint, and converge to the manifold. Finally, our experiments demonstrate the promise of our approach to an array of machine-learning problems that involve orthogonality constraints.

Keywords: Orthogonal manifold, Stiefel manifold, stochastic optimization, variance reduction

1. Introduction

We consider the problem of optimizing a function f from $\mathbb{R}^{n\times p}$ to $\mathbb{R}$ with orthogonality constraints:

$$\min_{X\in\mathbb{R}^{n\times p}} f(X), \qquad \text{s.t.} \quad X \in \mathrm{St}(p,n) = \{X \in \mathbb{R}^{n\times p} |\ X^\top X = I_p\}, \tag{1}$$

where $\mathrm{St}(p,n)$ is *the Stiefel manifold*. Such a problem appears naturally in many machine learning problems as a way to control dissimilarity between learned features, e.g. in principal component analysis (PCA) (Hotelling, 1933), independent component analysis

(ICA) (Hyvarinen, 1999; Theis et al., 2009; Ablin et al., 2018), canonical correlation analysis (Hotelling, 1936), and more recently, for the training of neural networks to improve their stability (Arjovsky et al., 2015; Zhang et al., 2021; Wang et al., 2020) and robustness against adversarial attacks (Cisse et al., 2017; Li et al., 2019b,a; Singla and Feizi, 2021).

Riemannian optimization techniques are based on the observation that the orthogonality constraints in (1) define a smooth matrix manifold (Absil et al., 2008; Boumal, 2023) called the Stiefel manifold. The geometry of the manifold constraint allows for the extension of optimization techniques from the Euclidean to the manifold setting, including second-order methods (Absil et al., 2007), stochastic gradient descent (Bonnabel, 2013), and variance-reduced methods (Zhang et al., 2016; Tripuraneni et al., 2018; Zhou et al., 2019).

A crucial part of Riemannian optimization methods is the use of *retraction* (Adler et al., 2002; Absil and Malick, 2012), which is a projection map on the manifold preserving the first-order information, and ensures that the iterates remain on the manifold. Computing retractions with orthogonality constraints involves linear algebra operations, such as matrix exponentiation, inverse, or QR decomposition. In some applications, e.g., when evaluating the gradient is relatively cheap, computing the retraction is the dominant cost of the optimization method. Unlike Euclidean optimization, ensuring that the iterates move on the manifold can be more costly than computing the gradient direction.

Additionally, the need to perform retractions, and more generally, to take the manifold's curvature into account, causes challenges in developing accelerated techniques in Riemannian optimization that the community has just started to overcome (Bécigneul and Ganea, 2018; Alimisis et al., 2021; Criscitiello and Boumal, 2022). As a result, practitioners in deep learning sometimes rely on the use of adding a squared penalty term and minimize $f(X) + \lambda\mathcal{N}(X)$ with $\mathcal{N}(X) = \|X^\top X - I_p\|^2/4$ in the Frobenius norm, which does not perfectly enforce the constraint.

Unlike Riemannian techniques, where the constraint is exactly enforced in every iteration, and the squared penalty method, where the optimum of the problem is not exactly on the manifold, we employ a method that is in between the two. Motivated by the previous work of Ablin and Peyré (2022) for the orthogonal matrix manifold, we design an algorithm that does not enforce the constraint exactly at every iteration but instead controls the distance to the constraint employing inexpensive matrix multiplication. The iterates *land* on the manifold exactly at convergence with the same convergence guarantees as standard Riemannian methods for solving (1).

The following subsection provides a brief prior on the current optimization techniques for solving (1). The rest of the paper is organized as follows:

- In Section 2 we extend the landing algorithm to $\mathrm{St}(p, n)$. By bounding the step size, we guarantee that the iterates remain close to the manifold. We show that a function introduced in (Gao et al., 2019) based on the augmented Lagrangian is a merit function for the landing algorithm. This forms the cornerstone of our improved analysis over that of Ablin and Peyré (2022).

- We use these tools for the theoretical analysis of the basic and variants of the landing algorithm in Section 3. In Subsection 3.1, thanks to the new merit function, we greatly improve the theoretical results of Ablin and Peyré (2022): we demonstrate that the basic landing method with constant step size achieves $\inf_{k\leq K} \|\mathrm{grad} f(X_k)\|^2 =$

$O(K^{-1})$. In Subsection 3.2, we introduce a stochastic algorithm dubbed stochastic landing algorithm. We show that with a step size decaying in $K^{-\frac{1}{2}}$, it achieves $\inf_{k\leq K} \|\mathrm{grad} f(X_k)\|^2 = O(K^{-\frac{1}{2}})$. In Subsection 3.3, we extend the SAGA algorithm and show that the SAGA landing algorithm achieves $\inf_{k\leq K} \|\mathrm{grad} f(X_k)\|^2 = O(K^{-1})$. We recover each time the same convergence rate and sample complexity as the classical Riemannian counterpart of the methods—that uses retractions. We also demonstrate the convergence of all these methods to the Stiefel manifold.

- In Section 4, we numerically demonstrate the merits of the method when computing retractions is a bottleneck of classical Riemannian methods.

Regarding the convergence speed results, the reader should be aware that we use the *square norm* of the gradient as a criterion, while some readers might be more familiar with results stated without a square.

This paper improves on the results of Ablin and Peyré (2022) in several important and non-trivial directions: i) the landing algorithm is extended to the Stiefel manifold, while Ablin and Peyré (2022) only consider the square $n = p$ case, ii) stochastic and variance reduced extensions of the landing algorithm are proposed, while Ablin and Peyré (2022) only consider gradient descent, and iii) we greatly improve the theoretical results, thanks to a new analysis. We essentially prove that all proposed algorithms converge at the same rate as their Riemannian counterparts using the same step-size schedule, and converge to the manifold. Ablin and Peyré (2022) could only prove convergence of the base landing method using *decaying* step sizes, leading to a sub-optimal $K^{-\frac{1}{3}}$ convergence rate, while our result Theorem 9 proves that the base landing algorithm converges at a K^{-1} rate with constant step-size, heavily improving the rate.[1]

Notation The norm of a matrix $\|M\|$ denotes the Frobenius norm, i.e., the vectorized ℓ_2 norm. We let I_p denote the $p \times p$ identity matrix, and $\mathrm{St}(p, n)$ denote the Stiefel manifold, which is the set of $n \times p$ matrices such that $X^\top X = I_p$. The tangent space of the Stiefel manifold at $X \in \mathrm{St}(p, n)$ is denoted by $\mathrm{T}_X\mathrm{St}(p, n)$. The projection on the set of $p \times p$ skew-symmetric matrices denoted by $\mathrm{skew}_p \subset \mathbb{R}^{p\times p}$ is $\mathrm{skew}(M) = \frac{1}{2}(M - M^\top)$ and on the set of symmetric matrices is $\mathrm{sym}(M) = \frac{1}{2}(M + M^\top)$. The exponential of a matrix is $\exp(M)$. The gradient of a function $f : \mathbb{R}^{n\times p} \to \mathbb{R}$ is denoted by ∇f and we define its Riemannian gradient as $\mathrm{grad} f(X) = \mathrm{skew}(\nabla f(X)X^\top)X$ for all $X \in \mathbb{R}^{n\times p}$ as explained in detail in Section 2. We say that a function is L-smooth if it is differentiable and its gradient is L-Lipschitz. The constant L is then the smoothness constant of f.

1.1 Prior Work on Optimization with Orthogonality Constraints

Equation (1) is an optimization problem over a matrix manifold. In the literature, we find two main approaches to solving this problem, reviewed next.

1. When this work was ready to be made public, the preprint (Schechtman et al., 2023) appeared that pursues similar goals. It addresses the more general problem of equality-constrained optimization, it uses a different merit function than the one we introduce in Section 2.2 and it does not consider variance reduction as we do in Section 3.3. The numerical experiments also differ considerably, as they only consider a Procrustes problem, while we experiment with deep neural networks.

1.1.1 Trivializations

This approach proposed by Lezcano-Casado and Martínez-Rubio (2019); Lezcano-Casado (2019) consists in finding a differentiable surjective function $\phi : E \to \mathrm{St}(p, n)$ where E is a Euclidean space, and to solve

$$\min_{A \in E} f(\phi(A)) \ . \tag{2}$$

The main advantage of this method is that it turns a Riemannian optimization problem on $\mathrm{St}(p, n)$ into an optimization on a Euclidean space, for which we can apply all existing Euclidean methods, such as gradient descent, stochastic gradient descent, or Adam. This is especially convenient in deep learning, where most standard optimizers are not meant to handle Riemannian constraints. However, this method has two major drawbacks. First, it can drastically change the optimization landscape. Second, the gradient of the function that is optimized is, following the chain rule, $\nabla(f \circ \phi) = \left(\frac{\partial \phi}{\partial z}\right)^\top \nabla f \circ \phi$, and the Jacobian-vector product can be very expensive to compute.

To give a concrete example, we consider the parametrization ϕ used by Singla and Feizi (2021): $\phi(A) = \exp(A) \begin{pmatrix} I_p \\ 0 \end{pmatrix}$, where $A \in \mathrm{skew}_n$, with skew_n the set of $n \times n$ skew symmetric matrices. We see that computing this trivialization requires computing the exponential of a potentially large $n \times n$ matrix. Furthermore, when computing the gradient of $f \circ \phi$, one needs to compute the Jacobian-vector product with a matrix exponential, which requires computing a larger $2n \times 2n$ matrix exponential (Lezcano-Casado and Martínez-Rubio, 2019): this becomes prohibitively costly when n is large.

1.1.2 Riemannian optimization

This approach consists in extending the classical Euclidean methods such as gradient descent or stochastic gradient descent to the Riemannian setting. For instance, consider Euclidean gradient descent to minimize f in the Euclidean setting, which iterates $X_{k+1} = X_k - \eta \nabla f(X_k)$ where $\eta > 0$ is a step size. There are two ingredients to transform it into a Riemannian method. First, the Euclidean gradient $\nabla f(X)$ is replaced by the Riemannian gradient $\mathrm{grad} f(X)$. Second, the subtraction is replaced by a retraction $\mathcal{R}$, which allows moving while staying on the manifold. We obtain the iterations of the Riemannian gradient descent:

$$X_{k+1} = \mathcal{R}(X_k, -\eta \mathrm{grad} f(X_k)) \ . \tag{3}$$

In the case of $\mathrm{St}(p, n)$ (Edelman et al., 1998), the tangent space at X is the set

$$\mathrm{T}_X \mathrm{St}(p, n) = \{X\Omega + X_\perp K : \Omega \in \mathrm{skew}_p, K \in \mathbb{R}^{n-p \times p}\} = \{WX : W \in \mathrm{skew}_n\} \tag{4}$$

and the Riemannian gradient with respect to the canonical metric (Edelman et al., 1998) is

$$\mathrm{grad} f(X) = \mathrm{skew}(\nabla f(X) X^\top) X \tag{5}$$

where $\mathrm{skew}(M) = \frac{1}{2}(M - M^\top)$ is the skew-symmetric part of a matrix. In (5), for convenience, we have omitted a factor of 2; compare with (Gao et al., 2022, §3). This omission is equivalent to magnifying the canonical metric by a factor of 2. From a computational point of view, computing this gradient is cheap: it can be rewritten, for $X \in \mathrm{St}(p, n)$, as

$\mathrm{grad} f(X) = \frac{1}{2}(\nabla f(X) - X\nabla f(X)^\top X)$, which can be computed with one matrix-matrix product of size $p \times n$ and $n \times p$, and one matrix-matrix product of size $n \times p$ and $p \times p$.

A retraction $\mathcal{R}(X, Z) = Y$ is a mapping that takes as inputs $X \in \mathrm{St}(p, n)$ and $Z \in \mathrm{T}_X\mathrm{St}(p, n)$, and outputs $Y \in \mathrm{St}(p, n)$, such that it "goes in the direction of Z at the first order", i.e., we have $Y = X + Z + o(\|Z\|)$. It defines a way to move on the manifold $\mathrm{St}(p, n)$. We give several examples, where we write $Z = X\Omega + X_\perp K = WX$ in view of (4).

1. *Exponential retraction*:

$$\mathcal{R}(X, Z) = \begin{pmatrix} X & X_\perp \end{pmatrix} \exp \begin{pmatrix} \Omega & -K^\top \\ K & 0 \end{pmatrix} \begin{pmatrix} I_p \\ 0 \end{pmatrix} \ . \tag{6}$$

   This is the exponential map—that follows geodesics—for the canonical metric on the manifold (Zimmermann and Hüper, 2022, (10)). The most expensive computations are a matrix exponentiation of a matrix of size $2p \times 2p$ and a matrix-matrix product between matrices of size $n \times 2p$ and $2p \times 2p$.

2. *Cayley retraction*:

$$\mathcal{R}(X, Z) = (I_n - \frac{W}{2})^{-1}(I_n + \frac{W}{2})X \ .$$

   Though the inverse exists for any $W \in \mathrm{skew}_n$, it requires solving a $n \times n$ linear system that dominates the cost. When Z takes the form (5) and $\nabla f(X)$ is available, this retraction can be computed in $8np^2 + O(p^3)$ flops, see Wen and Yin (2012, §2.2)

3. *Projection retraction*:

$$\mathcal{R}(X, Z) = \mathrm{Proj}(X + Z), \ \text{ with } \mathrm{Proj}(Y) = Y(Y^\top Y)^{-\frac{1}{2}} \ . \tag{7}$$

   Computing this retraction requires computing the inverse-square root of a matrix, which is also a costly linear algebra operation.

These operations allow us to implement Riemannian gradient descent. Many variants have then been proposed to accelerate convergence, among which trust-region methods (Absil et al., 2007) and Nesterov acceleration (Ahn and Sra, 2020).

In most machine learning applications, the function f corresponds to empirical risk minimization, and so it has a sum structure. It can be written as:

$$f(X) = \frac{1}{N}\sum_{i=1}^{N} f_i(X) \ , \tag{8}$$

where N is the number of samples and each f_i is a "simple" function. In the Euclidean case, stochastic gradient descent (SGD) (Robbins and Monro, 1951) is the algorithm of choice to minimize such a function. At each iteration, it takes a random function f_i and goes in the direction opposite to its gradient. Similarly, in the Riemannian case, we can implement Riemannian-SGD (Bonnabel, 2013) by iterating

$$X_{k+1} = \mathcal{R}(X_k, -\eta_k \mathrm{grad} f_i(X_k)), \ \text{ where } i \sim \mathcal{U}[1, N] \tag{9}$$

with $i \sim \mathcal{U}[1, N]$ meaning that index i is drawn from the discrete uniform distribution between 1 and N at each iteration. This method only requires one sample at each iteration; hence it scales gracefully with N. However, its convergence is quite slow and typically requires diminishing step sizes.

Variance reduction techniques (Johnson and Zhang, 2013; Schmidt et al., 2017; Defazio et al., 2014) are classical ways to mitigate this problem and allow to obtain algorithms that are stochastic (i.e., use only one sample at a time) and that provably converge while having a constant step size, with a faster rate of convergence than SGD.

1.2 Infeasible methods

Like the present work, several infeasible methods have been proposed to solve the optimization problem, while not enforcing the orthogonality constraint at each iteration. Ablin et al. (2018) propose a method we extend in this work in several directions. Gao et al. (2019) propose an approximated augmented Lagrangian method, taking steps in directions that approximate the gradient of the augmented Lagrangian. We draw inspiration from this work in order to perform our theoretical analysis; a detailed account of the differences with our algorithm is given in subsection 2.2. Finally, Liu et al. (2024) propose a proximal method to solve possibly non-smooth problems on the manifold, with an approximate orthogonalization scheme at each iteration. Owing to the more complicated problem structure they consider, the proposed method invokes at each step an inner iterative solver to compute a proximal operator.

2. Geometry of the Landing Field and its Merit Function

For $\lambda > 0$ and $X \in \mathbb{R}^{n\times p}$, we define the landing field as

$$\Lambda(X) = \mathrm{grad} f(X) + \lambda X(X^\top X - I_p), \tag{10}$$

where $\mathrm{grad} f(X) = \mathrm{skew}(\nabla f(X) X^\top) X$. This field will be used to define iterates $\tilde{X} = X - \eta\Lambda(X)$ and different variants in the next sections. We define

$$\mathcal{N}(X) = \frac{1}{4}\|X^\top X - I_p\|^2 \tag{11}$$

where $\|\cdot\|$ is the Frobenius norm so that the second term in (10) is $\lambda\nabla\mathcal{N}(X)$.

Here, $\mathrm{grad} f(X)$ denotes the extension to all $X \in \mathbb{R}^{n\times p}$ of formula (5). It thus coincides with the Riemannian gradient when $X \in \mathrm{St}(p, n)$. This extension has several attractive properties. First, for all full-rank $X \in \mathbb{R}^{n\times p}$, $\mathrm{grad} f(X)$ can still be seen as a Riemannian gradient of f on the manifold $\{Y \in \mathbb{R}^{n\times p} \mid Y^\top Y = X^\top X\}$ with respect to a canonical-type metric, as shown in (Gao et al., 2022, Proposition 4). Second, this formula yields orthogonality between the two terms of the field Λ, for any X. Indeed, we have $\langle \mathrm{grad} f(X), X(X^\top X - I_p)\rangle = \langle \mathrm{skew}(\nabla f(X) X^\top), X(X^\top X - I_p) X^\top\rangle$, which cancels since it is the scalar product between a skew-symmetric and a symmetric matrix.

The intuition behind the landing direction is fairly simple: the component $-\mathrm{grad} f(X)$ is here to optimize the function, while the term $-X(X^\top X - I_p)$ attracts X towards $\mathrm{St}(p, n)$. More formally, since these two terms are orthogonal, the field cancels if and only if both

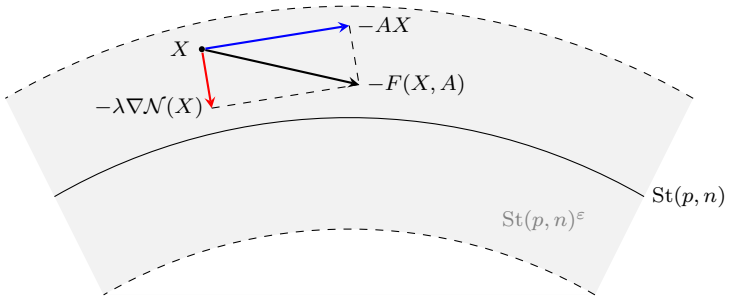


Figure 1: Illustration of the geometry of the field F. Note the orthogonality of the two components.

terms cancel. The fact that $X(X^\top X - I_p) = 0$ gives, assuming X injective, $X^\top X = I_p$, hence $X \in \mathrm{St}(p,n)$. Then, the fact that $\mathrm{grad} f(X) = 0$ combined with $X \in \mathrm{St}(p,n)$ shows that X is a first-order critical point of the function f on the manifold. This reasoning is qualitative: the next part formalizes this geometrical intuition.

2.1 Geometrical Results and Intuitions

In the remainder of the paper, we will always consider algorithms whose iterates stay close to the manifold $\mathrm{St}(p,n)$. We measure this closeness with the function $\mathcal{N}(X)$ (introduced in (11)), and define the *safe region* as

$$\mathrm{St}(p,n)^\varepsilon = \{X \in \mathbb{R}^{n\times p} |\ \mathcal{N}(X) \leq \frac{1}{4}\varepsilon^2\} = \{X \in \mathbb{R}^{n\times p} |\ \|X^\top X - I_p\| \leq \varepsilon\} \tag{12}$$

for some ε between 0 and 1. A critical part of our work is to ensure that the iterates of our algorithms remain in $\mathrm{St}(p,n)^\varepsilon$, which in turn guarantees the following bounds on the singular values of X. All proofs are deferred to the appendix.

Lemma 1 *For all $X \in \mathrm{St}(p,n)^\varepsilon$, the singular values of X are between $\sqrt{1-\varepsilon}$ and $\sqrt{1+\varepsilon}$.*

Note that when $\varepsilon = 0$, the singular values of X are all equal to 1, thus making the columns of the matrix orthogonal and ensuring that $X \in \mathrm{St}(p,n)$.

As noted before, a critical feature of the landing field is the orthogonality of the two components, which holds between $\nabla\mathcal{N}(X)$ and any direction AX with a skew-symmetric matrix A. In order for the results to generalize to the stochastic and variance reduction setting, in the rest of this section we consider a more general form of the landing field as

$$F(X,A) = AX + \lambda\nabla\mathcal{N}(X), \tag{13}$$

where A is an $n \times n$ skew-symmetric matrix. The formula of the landing field in (10) is recovered by taking $A = \mathrm{skew}(\nabla f(X)X^\top)$ in the above equation (13), that is to say $\Lambda(X) = F(X, \mathrm{skew}(\nabla f(X)X^\top))$.

Fields of the form (13) will play a key role in our analysis, exhibiting interesting geometrical properties which stem from the orthogonality of the two terms: AX and

$\nabla\mathcal{N}(X) = X(X^\top X - I_p)$ are orthogonal. Figure 1 illustrates the geometry of the problem and of the field F. We have the following inequalities:

Proposition 2 *For all $X \in \mathrm{St}(p,n)^\varepsilon$ and $A \in \mathrm{skew}_n$, the norm of the field (13) admits the following bounds:*

$$\|AX\|^2 + 4\lambda^2(1-\varepsilon)\mathcal{N}(X) \leq \|F(X,A)\|^2 \leq \|AX\|^2 + 4\lambda^2(1+\varepsilon)\mathcal{N}(X)$$

The orthogonality of the two terms also ensures that going in the direction of $-F(X,A)$ allows us to remain in the safe region $\mathrm{St}(p,n)^\varepsilon$ as long as the step size is small enough.

Lemma 3 (Step-size safeguard) *Let $X \in \mathrm{St}(p,n)^\varepsilon$, $A \in \mathrm{skew}_n$, and consider the update $\tilde{X} = X - \eta F(X,A)$, where $\eta > 0$ is a step size and $F(X,A)$ is the field (13). Define $g = \|F(X,A)\|$ and $d = \|X^\top X - I_p\|$. If the step size satisfies*

$$\eta \leq \eta(X) := \min\left\{\frac{\lambda d(1-d) + \sqrt{\lambda^2 d^2(1-d)^2 + g^2(\varepsilon - d)}}{g^2}, \frac{1}{2\lambda}\right\}, \tag{14}$$

then the next iterate $\tilde{X}$ remains in $\mathrm{St}(p,n)^\varepsilon$.

This lemma is of critical importance, both from a practical and theoretical point of view. In practice, at each iteration of our algorithms introduced later, we always compute the safeguard step (14), and if the safeguard step is smaller than the prescribed step size, we use the safeguard step instead. Note that the formula for the safeguard step only involves quantities that are readily available when the field F has been computed: computing $\eta(X)$ at each iteration does not add a significant computational overhead.

We can furthermore lower bound the safeguard step in (14) by a quantity independent of X:

Lemma 4 (A lower-bound on the step-size safeguard) *Assuming that $\|AX\|_F \leq \tilde{a}$, we have that the upper-bound in Lemma 3 is lower-bounded by*

$$\eta(X) \geq \eta^*(\tilde{a}, \varepsilon, \lambda) \tag{15}$$

where the quantity $\eta^*(\tilde{a}, \varepsilon, \lambda)$, given in Appendix B.4, is positive for any λ, ε and $\tilde{a}$.

This lemma serves to prove that the iterates of our algorithms all stay in the safe region provided that the step size is small enough. We have:

Proposition 5 *Consider a sequence of iterates X_k defined by recursion, starting from $X_0 \in \mathrm{St}(p,n)^\varepsilon$. We assume that there is a family of maps $\mathcal{A}_k(X_0, \dots, X_k) = A_k \in \mathrm{skew}_n$ such that $X_{k+1} = X_k - \eta_k F(X_k, A_k)$ where $\eta_k > 0$ is a step size. In addition, we assume that there is a constant $\tilde{a} > 0$ such that for all $X_0, \dots, X_k \in \mathrm{St}(p,n)^\varepsilon$, we have $\|\mathcal{A}_k(X_0, \dots, X_k)X_k\| \leq \tilde{a}$. Then, if all η_k are such that $\eta_k \leq \eta^*(\tilde{a}, \varepsilon, \lambda)$ with η^* defined in Lemma 4, we have that all iterates satisfy $X_k \in \mathrm{St}(p,n)^\varepsilon$.*

This proposition shows that an algorithm that follows a direction of the form (13) with sufficiently small steps will stay within the safe region $\mathrm{St}(p,n)^\varepsilon$. The definition of the maps $\mathcal{A}_k$ is cumbersome as it depends on the past iterates, but it is needed to handle the variance reduction algorithm that we study later. This result is central to this article since all algorithms considered in Section 3 produce sequences that satisfy the hypothesis of Proposition 5.

2.2 A Merit Function

The next proposition defines a smooth merit function for the landing field $\Lambda(X)$ defined in (10). The existence of such a merit function is central to a simple analysis of the landing algorithm and its different extensions. We consider as in (Gao et al., 2019):

$$\mathcal{L}(X) = f(X) + h(X) + \mu\mathcal{N}(X), \tag{16}$$

where $h(X) = -\frac{1}{2}\langle \mathrm{sym}(X^\top \nabla f(X)), X^\top X - I_p\rangle$ and $\mu > 0$, which is suitably chosen in the following result.

Proposition 6 (Merit function bound) *Let $\mathcal{L}(X)$ be the merit function defined in* (16)*. For all $X \in \mathrm{St}(p,n)^\varepsilon$ we have with $\nu = \lambda\mu$:*

$$\langle \nabla\mathcal{L}(X),\, \Lambda(X)\rangle \geq \frac{1}{2}\|\mathrm{grad} f(X)\|^2 + \nu\mathcal{N}(X), \tag{17}$$

for the choice of

$$\mu \geq \frac{2}{3-4\varepsilon}\left(L(1-\varepsilon) + 3s + \hat{L}^2\frac{(1+\varepsilon)^2}{\lambda(1-\varepsilon)}\right) \ , \tag{18}$$

where $s = \sup_{X\in\mathrm{St}(p,n)^\varepsilon} \|\mathrm{sym}(X^\top\nabla f(X))\|$, $L > 0$ is the smoothness constant of f over $\mathrm{St}(p,n)^\varepsilon$, $\hat{L} = \max(L, L')$ with $L' = \max_{X\in\mathrm{St}(p,n)^\varepsilon} \|\nabla f(X)\|$, and $\varepsilon < \frac{3}{4}$.

This demonstrates that $\mathcal{L}$ is in fact a merit function. Indeed, the landing direction is an ascent direction for the merit function, since $\langle \nabla\mathcal{L}(X), \Lambda(X)\rangle \geq 0$. We can then combine this proposition and Proposition 2 get the following result:

Proposition 7 *Under the same conditions as in Proposition 6, defining $\rho = \min(\frac{1}{2}, \frac{\nu}{4\lambda^2(1+\varepsilon)})$, we have for $X \in \mathrm{St}(p,n)^\varepsilon$:*

$$\langle \Lambda(X), \nabla\mathcal{L}(X)\rangle \geq \rho\|\Lambda(X)\|^2.$$

We now turn to the intuition behind the merit function. The merit function $\mathcal{L}$ is composed of three terms. The terms $f(X)$ and $\mu\mathcal{N}(X)$ are easy to interpret: the first one controls optimality, while the second controls the distance to the manifold. The term $h(X)$ might be mysterious at first. Its role is best understood when X is on the manifold. Indeed, for $X \in \mathrm{St}(p,n)$ we get

$$\nabla h(X) = -X\mathrm{sym}(X^\top\nabla f(X)) \ .$$

This vector is, in fact, the opposite of the projection of $\nabla f(X)$ on the normal space to $\mathrm{St}(p,n)$ at X. Hence, if $X \in \mathrm{St}(p,n)$ then $\mathcal{L}(X) = f(X)$ and $\nabla\mathcal{L}(X)$ is a vector in the tangent space $\mathrm{T}_X\mathrm{St}(p,n)$ which is makes an acute angle with $\mathrm{grad} f(X)$. Note that Fletcher's penalty function (Fletcher, 1970) is similar to the merit function $\mathcal{L}(X)$ while the $h(X)$ term is determined by the solution of the least squares problem. Notice that this merit function $\mathcal{L}(X)$ is the same as that of Gao et al. (2019) where $\mathcal{L}$ is constructed from the augmented Lagrangian function and $h(X)$ serves as a multiplier term since the multiplier of orthogonality constraints has a closed-form solution, $\mathrm{sym}(X^\top\nabla f(X))$, at any first-order stationary point. The main difference with the present work is that Gao et al. (2019) solve

the optimization problem by taking steps in a direction that approximates $-\nabla\mathcal{L}(X)$, but that does not satisfy the orthogonality hypothesis and hence is not guaranteed to converge for any value of $\lambda > 0$ (see also the discussion in Appendix B in (Ablin and Peyré, 2022)).

As a sum of smooth terms, the merit function is also smooth:

Proposition 8 (Smoothness of the merit function) *The merit function $\mathcal{L}$ is L_g-smooth on $\mathrm{St}(p,n)^\varepsilon$, with $L_g = L_{f+h} + \mu L_{\mathcal{N}}$ where L_{f+h} is the smoothness constant of $f+h$ and $L_{\mathcal{N}}$ is that of $\mathcal{N}$, upper bounded for instance by $2 + 3\varepsilon$.*

Schechtman et al. (2023) consider instead the non-smooth merit function $\mathcal{L}'(X) = f(X) + M\|X^\top X - I_p\|$. Our merit function decreases faster in the direction normal to the manifold, which is why the term $h(X)$ is introduced to tame the contribution of f in that direction. The smoothness of our merit function renders the subsequent analysis of the algorithms particularly simple since the smoothness lemma applied on $\mathcal{L}$ directly gives a descent lemma. This forms the basic tool to analyze the landing method and its variants.

This new merit function allows us to obtain far stronger convergence results than those of Ablin and Peyré (2022), which only analyzed the deterministic method, and could only prove a slow rate with decreasing step-sizes. In the following, we demonstrate that the classical gradient descent, SGD, and variance reduction method can be straightforwardly extended to the landing setting, provably converge to the manifold, and recover the same convergence rates at their Riemannian counterparts, all thanks to this new merit function.

3. A Family of Landing Algorithms, and their Convergence Properties

In this section, we consider a family of methods all derived from a base algorithm, the landing gradient descent algorithm. All of our algorithms follow directions of the form (13).

3.1 Landing Gradient Descent: the Base Algorithm

This algorithm produces a sequence of iterates $X_k \in \mathbb{R}^{n\times p}$ by iterating

$$X_{k+1} = X_k - \eta\Lambda(X_k) \tag{19}$$

where $\eta > 0$ is a step size and Λ is the landing field defined in (10). Note that this method falls into the hypothesis of Proposition 5 with the simple maps $\mathcal{A}_k(X_0,\dots,X_k) = \mathrm{grad}f(X_k)$, so we can just take $\tilde{a} = \sup_{X\in\mathrm{St}(p,n)^\varepsilon} \|\mathrm{grad}f(X)\|$ to get a safeguard step size η^* that guarantees that the iterates of the landing algorithm stay in $\mathrm{St}(p,n)^\varepsilon$.

We will start with the analysis of this method, where we find that it achieves a rate of convergence of $\frac{1}{K}$: we have $\frac{1}{K}\sum_{k=0}^K \|\mathrm{grad}f(X_k)\|^2 = O(\frac{1}{K})$ and $\frac{1}{K}\sum_{k=0}^K \mathcal{N}(X_k) = O(\frac{1}{K})$. We, therefore, obtain the same properties as classical Riemannian gradient descent, with a reduced cost per iteration, but with different constants.

Proposition 9 *Consider the iteration (19) starting from $X_0 \in \mathrm{St}(p,n)^\varepsilon$. Define $\tilde{a} = \sup_{X\in\mathrm{St}(p,n)^\varepsilon} \|\mathrm{skew}(\nabla f(X)X^\top)X\|_F$, and let η^* be the safeguard step size chosen from Lemma 4. Let $\mathcal{L}^*$ be a lower bound of the merit function $\mathcal{L}$ on $\mathrm{St}(p,n)^\varepsilon$. Then, for $\eta \le \min(\frac{1}{2L_g}, \frac{\nu}{4\lambda^2 L_g(1+\varepsilon)}, \eta^*)$, we have*

$$\frac{1}{K}\sum_{k=1}^K \|\mathrm{grad}f(X_k)\|^2 \le \frac{4(\mathcal{L}(X_0)-\mathcal{L}^*)}{\eta K} \quad \textit{and} \quad \frac{1}{K}\sum_{k=1}^K \mathcal{N}(X_k) \le \frac{2(\mathcal{L}(X_0)-\mathcal{L}^*)}{\eta\nu K}.$$

This result demonstrates weak convergence to the stationary points of f on the manifold at a rate $\frac{1}{K}$, just like classical Riemannian gradient descent (Boumal et al., 2019). Some readers might be more familiar with the following consequence "without squares": $\inf_{k\leq K} \|\mathrm{grad} f(X_k)\| + \|X_k^\top X_k - I_p\| = O(\frac{1}{\sqrt{K}})$, i.e., there exists an iterate which has both a low gradient norm and is close to the manifold. This result is an important improvement over that of Ablin and Peyré (2022) since we do not require decreasing step sizes to get convergence and obtain a much better convergence rate.

3.2 Landing Stochastic Gradient Descent: Large Scale Orthogonal Optimization

We now consider the case where the function f is the average of N functions:

$$f(X) = \frac{1}{N}\sum_{i=1}^{N} f_i(X) \ .$$

We can define the landing field associated with each f_i by

$$\Lambda_i(X) = \mathrm{grad} f_i(X) + \lambda X(X^\top X - I_p) \ .$$

This way, we have

$$\Lambda(X) = \frac{1}{N}\sum_{i=1}^{N} \Lambda_i(X),$$

and if we take an index i uniformly at random between 1 and N we have

$$\mathbb{E}_i[\Lambda_i(X)] = \Lambda(X) \ .$$

In other words, the direction Λ_i is an unbiased estimator of the landing field Λ. We consider the landing stochastic gradient descent (Landing-SGD) algorithm, which at iteration k samples a random index i_k uniformly between 1 and N and iterates

$$X_{k+1} = X_k - \eta_k \Lambda_{i_k}(X_k) \tag{20}$$

where η_k is a sequence of step size. As is customary in the analysis of stochastic optimization algorithms, we posit a bound on the variance of Λ_i:

Assumption 10 *There exists $B > 0$ such that for all $X \in \mathrm{St}(p,n)^\varepsilon$, we have $\frac{1}{N}\sum_{i=1}^{N} \|\Lambda_i(X) - \Lambda(X)\|^2 \leq B$.*

This assumption is true when the Λ_i are continuous since $\mathrm{St}(p,n)^\varepsilon$ is a compact set. Lemma 4 and Prop. 5 come in handy to define a safeguard step size. Indeed we see that the algorithm follows the hypothesis of Prop. 5 with $\tilde{a} = \sup_{X\in\mathrm{St}(p,n)^\varepsilon,\ i\in\{1,\dots,n\}} \|\mathrm{grad} f_i(X)\|$. This allows to define a safeguard step-size η^* following Prop. 5. We then obtain a simple recursive bound on the iterates using the smoothness inequality:

Proposition 11 *Assume that* $\eta_k \leq \min(\frac{1}{2L_g}, \frac{\nu}{4\lambda^2 L_g(1+\varepsilon)}, \eta^*)$ *where* η^* *is the global safeguard step size obtained as above. Then,*

$$\mathbb{E}_{i_k}[\mathcal{L}(X_{k+1})] \leq \mathcal{L}(X_k) - \frac{\eta_k}{4}\|\mathrm{grad} f(X_k)\|^2 - \frac{\eta_k \nu}{2}\mathcal{N}(X_k) + \frac{L_g B \eta_k^2}{2},$$

where the expectation is taken with respect to the random variable i_k*.*

We get convergence rates of the stochastic landing algorithm with decreasing step sizes.

Proposition 12 *Assume that the step size is* $\eta_k = \eta_0 \times (1+k)^{-\frac{1}{2}}$ *with* $\eta_0 = \min(\frac{1}{2L_g}, \frac{\nu}{4\lambda^2 L_g(1+\varepsilon)}, \eta^*)$*, with* η^* *chosen as the safeguard step size. Then, we have*

$$\inf_{k\leq K} \mathbb{E}[\|\mathrm{grad} f(X_k)\|^2] = O\left(\frac{\log(K+1)}{\sqrt{K}}\right) \quad \textit{and} \quad \inf_{k\leq K} \mathbb{E}[\|\mathcal{N}(X_k)\|^2] = O\left(\frac{\log(K+1)}{\sqrt{K}}\right) \ .$$

The expectation here is taken with respect to all the random realizations of the random variables i_k*,* $k \leq K$*.*

This shows that our method with decreasing step size has the same convergence rate as Riemannian stochastic gradient descent with decreasing step size. With constant step size, we get the following proposition:

Proposition 13 *Assume that the step size is fixed to* $\eta = \eta_0 \times (1+K)^{-\frac{1}{2}}$ *with* $\eta_0 = \min(\frac{1}{2L_g}, \frac{\nu}{4\lambda^2 L_g(1+\varepsilon)}, \eta^*)$*. Then, we have*

$$\inf_{k\leq K} \mathbb{E}[\|\mathrm{grad} f(X_k)\|^2] = O\left(\frac{1}{\sqrt{K}}\right) \quad \textit{and} \quad \inf_{k\leq K} \mathbb{E}[\|\mathcal{N}(X_k)\|^2] = O\left(\frac{1}{\sqrt{K}}\right) \ .$$

Sample complexity The sample complexity of the algorithm is readily obtained from the bound: in order to find an ε-critical point of the problem and get both $\inf_{k\leq K} \|\mathrm{grad} f(X_k)\|^2 \leq \varepsilon$ and $\inf_{k\leq K} \mathcal{N}(X_k) \leq \varepsilon$, we need $O(\varepsilon^{-2})$ iterations. The O here only hides constants of the problem, like conditioning of f and hyperparameter λ, but this quantity is independent of the number of samples N. This matches the classical sample complexity results obtained with SGD in the Euclidean and Riemannian non-convex settings (Zhang et al., 2016).

3.3 Landing SAGA: Variance Reduction for Faster Convergence

In this section, we are in the same finite-sum setting as in Section 3.2. As in classical optimization, SGD suffers from the high variance of its gradient estimator, leading to sub-optimal convergence rates. A classical strategy to overcome this issue consists in using variance reduction algorithms, which build an estimator of the gradient whose variance goes to 0 as training progresses. Such algorithms have also been proposed in a Riemannian setting, but like most other methods, they also require retractions (Zhang et al., 2016).

We propose a retraction-free variance-reduction algorithm that is a crossover between the celebrated SAGA algorithm (Defazio et al., 2014) and the landing algorithm, called the landing SAGA algorithm. The algorithm keeps a memory of the last gradient seen for each sample, $\Phi_k^1, \ldots, \Phi_k^N$ where each $\Phi_k^i \in \mathbb{R}^{n\times p}$. At iteration k, we sample at random an index

i_k between 1 and N, and compute the direction $\Lambda_k^{i_k} = \mathrm{grad} f_{i_k}(X_k) - \mathrm{skew}(\Phi_k^{i_k} X_k^\top) X_k + \frac{1}{N}\sum_{j=1}^N \mathrm{skew}(\Phi_k^j X_k^\top) X_k + \lambda X_k (X_k^\top X_k - I_p)$. We update the memory corresponding to sample i_k by doing $\Phi_{k+1}^{i_k} = \nabla f_{i_k}(X_k)$, and $\Phi_{k+1}^j = \Phi_k^j$ for all $j \neq i_k$. We then move in the direction

$$X_{k+1} = X_k - \eta \Lambda_k^{i_k}.$$

It is important to note that variance reduction is only applied on the "Riemannian" part $\mathrm{grad} f_i(X)$. The other term $X(X^\top X - I_p)$ is treated as usual. Like in the classical SAGA algorithm, we have the unbiasedness property:

$$\mathbb{E}_i[\Lambda_k^i] = \Lambda(X_k) \ .$$

This means that, on average, the direction we take is the landing field, computed over the whole dataset. The gist of this method is that we can have fine control on $\mathbb{E}_i[\|\Lambda_i^k\|^2]$. Indeed, letting $D_k^i = \mathrm{grad} f_i(X_k) - \mathrm{skew}(\Phi_k^i X_k^\top) X_k$, we have

$$\Lambda_k^i = \Lambda(X_k) + \underbrace{D_k^i - \mathbb{E}_j[D_k^j]}_{\text{zero-mean}}, \tag{21}$$

so that a bias-variance decomposition gives

$$\mathbb{E}_i[\|\Lambda_k^i\|^2] = \|\Lambda(X_k)\|^2 + \mathbb{E}_i[\|D_k^i - \mathbb{E}_j[D_k^j]\|^2] \tag{22}$$

$$\leq \|\Lambda(X_k)\|^2 + \mathbb{E}_i[\|D_k^i\|^2], \tag{23}$$

and D_k^i can also be controlled since

$$\|D_k^i\| = \|\mathrm{skew}((\nabla f_i(X_k) - \Phi_k^i) X_k^\top) X_k\| \tag{24}$$

$$\leq (1+\varepsilon)\|\nabla f_i(X_k) - \Phi_k^i\| \tag{25}$$

$$\leq (1+\varepsilon) L_f \|X_k - X_k^i\|^2 \tag{26}$$

where X_k^i is the last iterate that what chosen for the index i (so X_k^i is such that $\nabla f(X_k^i) = \Phi_k^i$). We, therefore, recover that we need to control the distance from the memory X_k^i to the current point X_k, as is customary in the analysis of SAGA.

We have the following convergence theorem, which is obtained by combining the merit function and the proof technique of Reddi et al. (2016):

Proposition 14 *Define ρ as in Proposition 7, L_g as in Proposition 8 and L_f the smoothness constant of f on $\mathrm{St}(p,n)^\varepsilon$. Assume that the step size is such that*

$$\eta \leq \min\left(\eta^*, \frac{\rho}{L_g}, \frac{1}{\sqrt{8N(1+\varepsilon)}L_f}, \left(\frac{\rho}{8N(4N+2)L_g L_f^2 (1+\varepsilon)^2}\right)^{1/3}\right) \ .$$

Then, we have

$$\inf_{k \leq K} \mathbb{E}[\|\mathrm{grad} f(X_k)\|^2] = O\left(\frac{1}{\eta K}\right) \quad \textit{and} \quad \inf_{k \leq K} \mathbb{E}[\|\mathcal{N}(X_k)\|^2] = O\left(\frac{1}{\eta K}\right) \ .$$

As in classical optimization, using the variance reduction of SAGA recovers a $\frac{1}{K}$ rate with a stochastic algorithm.

Sample complexity When the number of samples N is large, the last term in the above "min" for the choice of step size is the smallest; hence the step size scales as $N^{-2/3}$. This shows that to get to a $\varepsilon-$critical point such that $\|\text{grad} f(X)\|^2 \leq \varepsilon$, we need $O(N^{\frac{2}{3}}\varepsilon^{-1})$ iterations. This matches the sample complexity of classical Euclidean SAGA in the non-convex setting (Reddi et al., 2016).

3.4 Comparison to Penalty Methods

It is a common practice in deep learning applications that the orthogonality is favored by adding an ℓ_2 regularization term and minimizing

$$f(X) + \lambda \mathcal{N}(X),$$

for example in (Balestriero et al., 2018; Xie et al., 2017; Bansal et al., 2018). This method leads to a small computational overhead compared to the simple unconstrained minimization of f, and it allows the use of standard optimization algorithms tailored for deep learning. However, it provides no guarantee that the orthogonality will be satisfied. Generally, there are two possible outcomes based on the choice of $\lambda > 0$. If λ is small, then the final point is far from orthogonality, defeating the purpose of the regularization. If λ is too large, then optimization becomes too hard, as the problem becomes ill-conditioned: its smoothness constant grows with λ while its strong convexity constant does not, since $\mathcal{N}(X)$ is not strongly convex (indeed, it is constant in the direction tangent to $\text{St}(p, n)$).

In order to have a more formal statement than the above intuition, we consider the simple case of a linear problem:

Proposition 15 *Let $M = U\Sigma V^\top$ be a singular value decomposition of M, where $U \in \text{St}(p, n)$, Σ is a diagonal matrix of positive entries, and $V \in \text{St}(p, p)$ is an orthogonal matrix. Let $\sigma_1 \leq \cdots \leq \sigma_p$ be the singular values of M. Then, the minimizer of $g(X) = \langle M, X\rangle + \lambda\mathcal{N}(X)$ is $X^* = -U\Sigma^* V^\top$ where Σ^* is the diagonal matrix of entries σ_i^* where σ_i^* is the minimizer of the scalar function $x \to -\sigma_i x + \frac{\lambda}{4}(x^2-1)^2$. Furthermore, we have two properties:*

- *The distance between X^* and the constrained solution $X_{\text{St}(p,n)} = UV^\top$ is of the order of λ^{-1} when λ goes to $+\infty$.*
- *The maximal and minimal eigenvalues of the Hessian H of g at X^* satisfy $\lambda_{\min}(H) \leq \sigma_p + \frac{\sigma_p^2}{4\lambda}$ and $\lambda_{\max}(H) \geq 2\lambda$. Hence the conditioning of H is at least $\frac{2\lambda}{\sigma_p + \frac{\sigma_p^2}{4\lambda}}$ which behaves like $\frac{2\lambda}{\sigma_p}$ as λ goes to $+\infty$.*

This unfortunate trade-off is illustrated in Figure 2, where we display the contours of g for different values of λ. In practical terms, these two points mean that :

- In order to get a solution close to the correct one $X_{\text{St}(p,n)}$, we have to take a large λ.
- But, taking a large λ makes the conditioning of the problem go to infinity. The number of iterations of gradient descent with fixed step-size on g to achieve $\|X - X^*\| \leq \epsilon$ is locally of the order $\frac{2\lambda}{\sigma_p}\log(\epsilon^{-1})$, which is linear in λ.

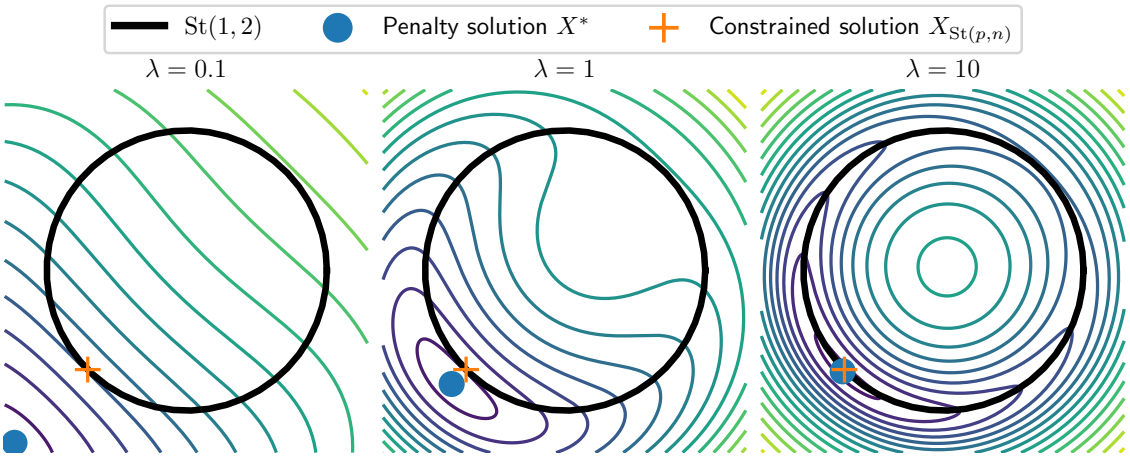


Figure 2: Contours of the function $g(X) = f(X) + \lambda\mathcal{N}(X)$ in dimensions $(n, p) = (2, 1)$. When λ is small, its minimizer is far from $\mathrm{St}(p, n)$, and when λ is large, g is badly conditioned around its optimum.

In order to better formalize the second point, consider that we start gradient descent from a point $X = X^* + \delta E$ where $\delta \ll 1$ and E is an eigenvector of H corresponding to its smallest eigenvalue, i.e., so that $H[E] = \lambda_{\min}(H)E$. We take the step size as the inverse of the local smoothness constant $\eta = 1/\lambda_{\max}(H)$, and define $Y = X - \eta\nabla g(X)$ the output of one iteration of gradient descent. Then, as $\delta \to 0$, we find

$$Y - X^* = (1 - \eta\lambda_{\min}(H))(X - X^*) + o(\delta)$$

which means that we have (approximately in δ) a linear convergence toward X^* with a rate $(1 - \eta\lambda_{\min}(H)) \geq 1 - \frac{\sigma_p}{2\lambda} + o(\frac{1}{\lambda})$. Hence, after K iterations of gradient descent, the error is at least $(1 - \frac{\sigma_p}{2\lambda})^K$, and having an error smaller than ϵ requires at least $K \geq \frac{\log(\epsilon^{-1})}{\log(1-\frac{\sigma_p}{2\lambda})} = \frac{2\lambda}{\sigma_p}\log(\epsilon^{-1}) + o(\frac{1}{\lambda})$ iterations: we need a number of iterations proportional to λ.

We want to insist that this behavior is in stark contrast with that of the landing methods presented above since they all provably converge *to the manifold* regardless of the choice of λ. In terms of computational overhead, they only require an additional computation of a Riemannian gradient (see Equation 5). We argue that this might be a very small price to pay in front of the benefits of having a method that provably converges to the correct points.

3.5 Computational cost

We now analyze the computational cost of one iteration of the landing methods presented above and compare it to the cost of classical Riemannian methods. All methods require computing the (perhaps stochastic) gradient of f, which we denote for short as $G \in \mathbb{R}^{n\times p}$. We denote t_G as the cost of this computation, which depends on the complexity of the function f itself. In the following, we use the fact that the cost of multiplying an $a \times b$ matrix with a $b \times c$ matrix is $O(abc)$.

Penalty method The penalty method needs to compute $\nabla(f(X) + \lambda\mathcal{N}(X)) = G + \lambda X(X^\top X - I_p)$. Hence, on top of computing G, it requires two matrix-matrix multiplications; and its overall cost is $t_G + O(np^2)$. We recall that this method does not converge to the stationary points of the correct problem as illustrated in subsection 3.4.

Landing methods The landing methods presented above then compute the direction $\Lambda(X) = \operatorname{skew}(GX^\top)X + \lambda X(X^\top X - I_p)$. Since $n \geq p$, the ordering of operations to compute this quantity that leads to the optimal complexity is:

- $A = X^\top X$ `Shape: p, p` `Cost:` $O(np^2)$
- $B = (\frac{1}{2}G + \lambda X)A$ `Shape: n, p` `Cost:` $O(np^2)$
- $C = G^\top X$ `Shape: p, p` `Cost:` $O(np^2)$
- $D = XC$ `Shape: n, p` `Cost:` $O(np^2)$
- Return $\Lambda = B - \frac{1}{2}D - \lambda X$ `Shape: n, p` `Cost:` $O(np)$

This makes it clear that the cost of implementing the landing method is $t_G + O(np^2)$ in terms of time and $O(np)$ in terms of memory. It requires only four matrix multiplications, which are heavily parallelizable. Note that compared to the penalty method, it requires only two more matrix multiplications. Interestingly, when $n \simeq p$ and we can afford to form $n \times n$ matrices, we can simply compute the field as $\Lambda = \left(\operatorname{skew}(GX^\top) + \lambda(XX^\top - I_n)\right)X$, which only requires three matrix multiplications.

Retraction-based methods Retraction-based methods have to compute costly retractions, as described in subsubsection 1.1.2. The cost of these computations is usually $O(np^2)$, but they are much slower than matrix multiplications when $n \simeq p$. Hence, the overall cost has the same order of magnitude as the landing methods, $t_G + O(np^2)$, but with much worse constants, making overall one landing iteration faster to compute. The cost of computing the gradient plays a key role here: indeed, if t_G is much greater than np^2, then the cost of both methods becomes very similar.

4. Experiments

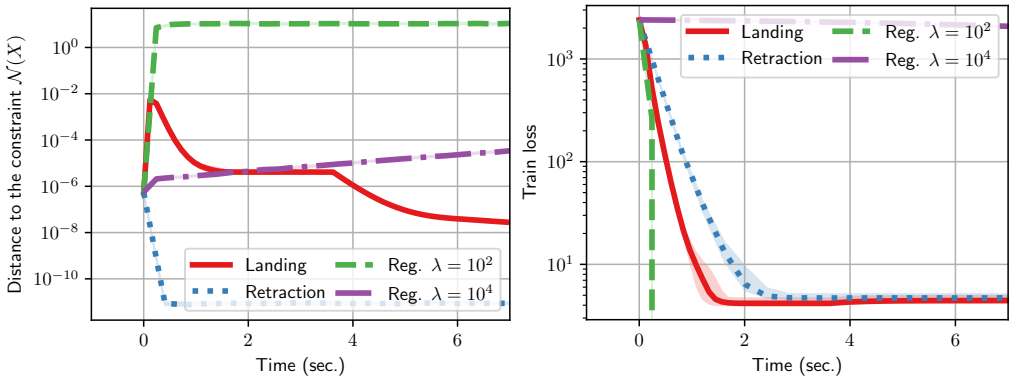
We numerically compare the landing method against the two main alternatives, the Riemannian gradient descent with QR retraction and the Euclidean gradient descent with added ℓ_2 squared penalty norm, with stochastic gradients.[2] In all experiments, we take the safe region parameter to be $\varepsilon = \frac{1}{2}$. Unless specified otherwise, we use for the landing term $\lambda = 1$, and choose the learning rate η with a grid search, just like for all other methods.

4.1 Online PCA

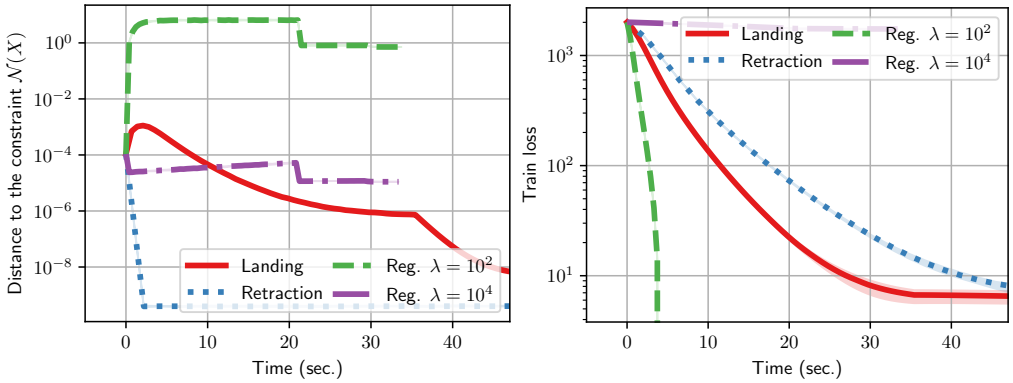
We test the methods performance on the online principal component analysis (PCA) problem

$$\min_{X \in \mathbb{R}^{n\times p}} -\frac{1}{2}\|AX\|_F^2, \quad \text{s.t.} \quad X \in \operatorname{St}(p, n), \tag{27}$$

2. The code to reproduce the experiments in this section is publicly available at: `https://github.com/simonvary/landing-stiefel`.



(a) $n = 5000, p = 200$



(b) $n = 5000, p = 1000$

Figure 3: Experiments with online PCA.

where $A \in \mathbb{R}^{N \times n}$ is a synthetically generated data matrix with $N = 15\,000$ being the number of samples each with dimension $n = 5000$. The columns of A are independently sampled from the normal distribution $\mathcal{N}(0, UU^\top + \sigma I_n)$, where $\sigma = 0.1$ and $U \in \mathbb{R}^{n \times p}$ is sampled from the Stiefel manifold with the uniform Haar distribution.

We compare the landing stochastic gradient method with the classical Riemannian gradient descent and with the "penalty" method which minimizes $f(X) + \lambda \mathcal{N}(X)$, where λ is now a regularization hyperparameter, using standard SGD. Figure 3 shows the convergence of the objective and the distance to the constraint against the computation time of the three methods using stochastic gradients with a batch size of 128 and a fixed step size, which decreases after 30 epochs. The training loss is computed as $f(X_k) - f(X_*)$, where X_* is the matrix of p right singular vectors of A. We see that in both cases of $p = 200$ and $p = 1000$ the landing is able to reach a lower objective value faster compared to the Riemannian gradient descent, however, at the cost of not being precisely on the constraint but with $\mathcal{N}(X) \leq 10^{-6}$. The distance is further decreased after 30 epochs as the fixed step size of the iterations is decreased as well. This test is implemented in PyTorch and performed using a single GPU.

Euclidean gradient descent with ℓ_2 regularization performs poorly with both choices of regularizer λ ("Reg." in the figure). In the first case when $\lambda = 10^2$ is too small, the distance

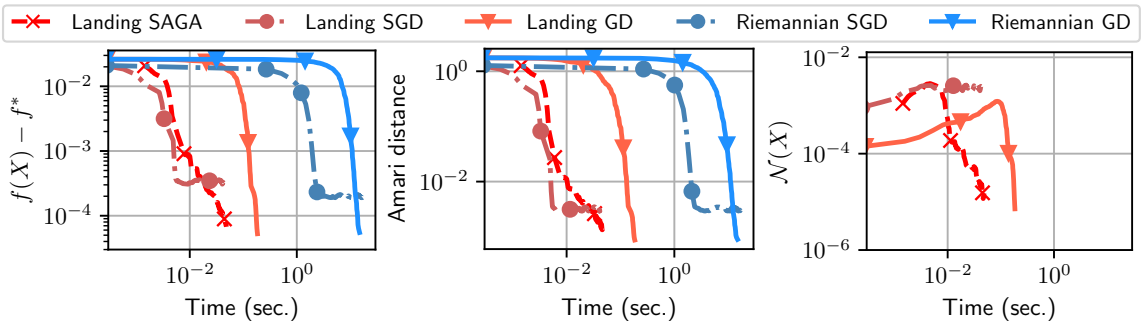


Figure 4: Experiments on ICA. Both scales are logarithmic.

remains large, and as a result, the training loss becomes negative since we are comparing against X_* which is on the constraint. In the second case of the large penalty term, when $\lambda = 10^4$, the iterates remain relatively close to the constraint, but the convergence rate is very slow. These experimental findings are in line with the theory explained in Section 3.4.

In general, we see that the landing method outperforms Riemannian gradient descent in cases when the computational cost of the retraction is more expensive relative to computing the gradient. This occurs especially in cases when p is large or the batch size of the stochastic gradient is small, as can be seen also in the additional experiments for $p = 100$ and $p = 500$ shown in Figure 6 in Appendix A.

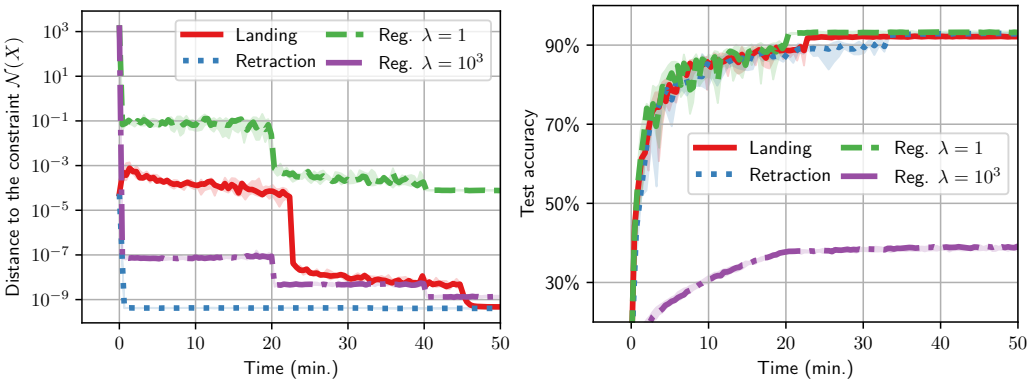
4.2 Independent Component Analysis

Given a data matrix $A = [a_1, \dots, a_N] \in \mathbb{R}^{N \times n}$, we perform the ICA of A by solving (Hyvarinen, 1999)

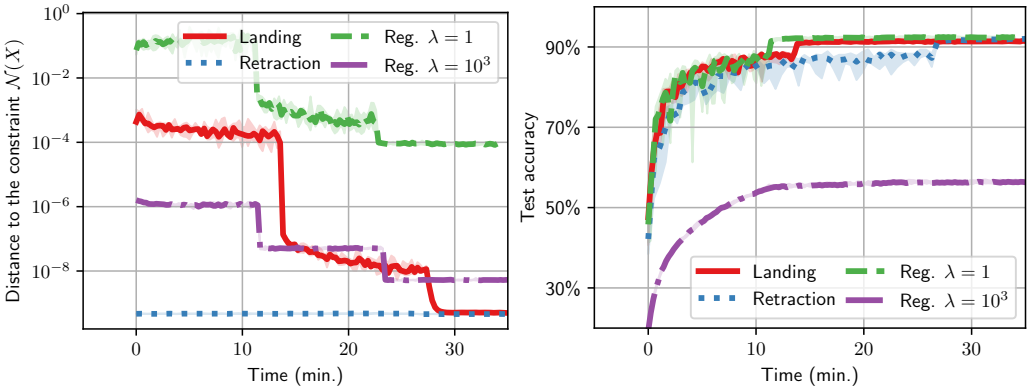
$$\min_{X \in \mathbb{R}^{n \times n}} \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{n} \sigma([AX]_{ij}), \text{ such that } X \in \mathrm{St}(n, n)$$

where σ is a scalar function defined by $\sigma(x) = \log(\cosh(x))$, so that $\sigma'(x) = \tanh(x)$.

We generate A as $A = SB^\top$, where $S \in \mathbb{R}^{N \times n}$ is the sources matrix, containing i.i.d. samples drawn from a Laplace distribution, and B is a random orthogonal $n \times n$ mixing matrix. We take $n = 10$ and $N = 10^4$. We run the base landing algorithm, the landing SGD, and the landing SAGA algorithm, with a batch size of 100 and compare them with Riemannian gradient descent and Riemannian SGD. Additionally to the loss and distance to the manifold, we also record the Amari distance, which measures how well $B^\top X$ is a scale and permutation matrix, i.e., how well we recover the sources. Results are displayed in Figure 4. We observe a fast convergence for the SGD algorithms followed by a plateau. The landing SAGA algorithm overcomes this plateau thanks to variance reduction. It is also much faster than the full-batch algorithm since it is a stochastic algorithm. We finally observe that its distance to the manifold reaches a very low value. Here, Riemannian methods are much slower than the landing because of the costly retractions. This experiment was run on a CPU using Benchopt (Moreau et al., 2022).



(a) Resnet18 experiments



(b) VGG16 experiments

Figure 5: Experiments with orthogonal convolutions.

4.3 Neural Networks with Orthogonality Constraints

We test the methods for training neural networks whose weights are constrained to the Stiefel manifold. Orthogonality of weights plays a prominent role in deep learning, for example in the recurrent neural networks to prevent the problem of instable gradients (Arjovsky et al., 2015), or in orthogonal convolutional neural networks that impose kernel orthogonality to improve the stability of models (Bansal et al., 2018; Wang et al., 2020).

We perform the test using two standard models, VGG16 (Simonyan and Zisserman, 2014) and Resnet18 (He and Sun, 2016), while constraining the kernels of all convolution layers to be on the Stiefel manifold. We reshape the convolutional kernels to the size $n_{\text{out}} \times n_{\text{in}} n_{\text{x}} n_{\text{y}}$, where $n_{\text{in}}, n_{\text{out}}$ is the number of input and output channels respectively and $n_{\text{x}}, n_{\text{y}}$ is the filter size. In the case when the reshaping results in a wide instead of a tall matrix, we impose the orthogonality on its transposition. We train the models using Riemannian gradient descent, Euclidean gradient descent with ℓ_2 regularization, and the landing method, with batch size of 128 samples for 150 epochs, and with a fixed step size that decreases as $\eta = \eta/10$ every 50 epochs. We repeat each training 5 times for different random seed. This test is implemented in PyTorch and performed using a single GPU.

Figure 5 shows the convergence of the test accuracy and the sum of distances to the constraints against the computation time, with the light shaded areas showing minimum

and maximum values of the 5 runs. The figure shows the landing is a strict improvement over the Euclidean gradient descent with the added ℓ_2 regularization, which, for the choice of $\lambda = 1$, achieves a good test accuracy, but at the cost of the poor distance to the constraint of 10^{-3}, and for the choice of $\lambda = 10^3$ converges to a solution that has similar distance as the landing of 10^{-8}, but has poor test accuracy. In comparison, training the models with the Riemannian gradient descent with QR retractions, achieves the lowest distance to the constraint, but also takes longer to reach the test accuracy of roughly 90%.

We also compared with the trivialization approach (Lezcano-Casado, 2019) using the Geotorch library, however this approach is not readily suitable for optimization over large Stiefel manifolds. See the experiments in Figure 7 in the Appendix A, which takes over 7 hours, i.e. approximately 14 times as long as the other methods, to reach the test accuracy of around 90% with VGG16.

5. Conclusion

We have extended the landing method of Ablin and Peyré (2022) from the orthogonal group to the Stiefel manifold, yielding an iterative method for smooth optimization problems where the decision variables take the form of a rectangular matrix constrained to be orthonormal. The iterative method is infeasible in the sense that orthogonality is not enforced at the iterates. We have obtained a computable bound on the step size ensuring that the next iterate stays in a safe region. This safeguard step size, along with the smooth merit function (16), has allowed for a streamlined complexity analysis in Section 3, both for the deterministic and stochastic cases. The various numerical experiments have illustrated the value of the proposed approach.

Acknowledgments

The authors thank Gabriel Peyré for fruitful discussions. This work was supported by the Fonds de la Recherche Scientifique – FNRS and the Fonds Wetenschappelijk Onderzoek – Vlaanderen under EOS Project no 30468160, and by the Fonds de la Recherche Scientifique – FNRS under Grant no T.0001.23. Most of this work was performed while Simon Vary was a beneficiary of an FSR Incoming Post-doctoral Fellowship at UCLouvain.

References

Pierre Ablin and Gabriel Peyré. Fast and accurate optimization on the orthogonal manifold without retraction. In *International Conference on Artificial Intelligence and Statistics*, pages 5636–5657. PMLR, 2022.

Pierre Ablin, Jean-François Cardoso, and Alexandre Gramfort. Faster ICA under orthogonal constraint. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4464–4468. IEEE, 2018.

P.-A. Absil and Jérôme Malick. Projection-like retractions on matrix manifolds. *SIAM Journal on Optimization*, 22(1):135–158, 2012.

P.-A. Absil, Christopher G Baker, and Kyle A Gallivan. Trust-region methods on Riemannian manifolds. *Foundations of Computational Mathematics*, 7:303–330, 2007.

P.-A. Absil, Robert Mahony, and Rodolphe Sepulchre. *Optimization Algorithms on Matrix Manifolds*, volume 36. Princeton University Press, Princeton, NJ, January 2008. ISBN 978-1-4008-3024-4. doi: 10.1515/9781400830244.

Roy L. Adler, Jean-Pierre Dedieu, Joseph Y. Margulies, Marco Martens, and Mike Shub. Newton's method on Riemannian manifolds and a geometric model for the human spine. *IMA J. Numer. Anal.*, 22(3):359–390, July 2002. doi: 10.1093/imanum/22.3.359.

Kwangjun Ahn and Suvrit Sra. From Nesterov's estimate sequence to Riemannian acceleration. In *Conference on Learning Theory*, pages 84–118. PMLR, 2020.

Foivos Alimisis, Antonio Orvieto, Gary Becigneul, and Aurelien Lucchi. Momentum improves optimization on Riemannian manifolds. In *International Conference on Artificial Intelligence and Statistics*, pages 1351–1359. PMLR, 2021.

Martin Arjovsky, Amar Shah, and Yoshua Bengio. Unitary Evolution Recurrent Neural Networks. *International Conference on Machine Learning*, 48, November 2015.

Randall Balestriero et al. A spline theory of deep learning. In *International Conference on Machine Learning*, pages 374–383. PMLR, 2018.

Nitin Bansal, Xiaohan Chen, and Zhangyang Wang. Can we gain more from orthogonality regularizations in training deep networks? *Advances in Neural Information Processing Systems*, 31, 2018.

Gary Bécigneul and Octavian-Eugen Ganea. Riemannian adaptive optimization methods. *arXiv preprint arXiv:1810.00760*, 2018.

Silvere Bonnabel. Stochastic gradient descent on Riemannian manifolds. *IEEE Transactions on Automatic Control*, 58(9):2217–2229, 2013.

Nicolas Boumal. *An introduction to optimization on smooth manifolds*. Cambridge University Press, 2023.

Nicolas Boumal, P.-A. Absil, and Coralia Cartis. Global rates of convergence for nonconvex optimization on manifolds. *IMA Journal of Numerical Analysis*, 39(1):1–33, 2019.

Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. Parseval networks: Improving robustness to adversarial examples. In *International Conference on Machine Learning*, pages 854–863. PMLR, 2017.

Christopher Criscitiello and Nicolas Boumal. Negative curvature obstructs acceleration for strongly geodesically convex optimization, even with exact first-order oracles. In *Conference on Learning Theory*, pages 496–542. PMLR, 2022.

Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. *Advances in neural information processing systems*, 27, 2014.

Alan Edelman, Tomás A Arias, and Steven T Smith. The geometry of algorithms with orthogonality constraints. *SIAM journal on Matrix Analysis and Applications*, 20(2): 303–353, 1998.

Roger Fletcher. A class of methods for nonlinear programming with termination and convergence properties. *Integer and nonlinear programming*, pages 157–173, 1970.

Bin Gao, Xin Liu, and Ya-xiang Yuan. Parallelizable algorithms for optimization problems with orthogonality constraints. *SIAM Journal on Scientific Computing*, 41(3):A1949–A1983, 2019.

Bin Gao, Simon Vary, Pierre Ablin, and P.-A. Absil. Optimization flows landing on the Stiefel manifold. *IFAC-PapersOnLine*, 55(30):25–30, 2022. ISSN 2405-8963. doi: https://doi.org/10.1016/j.ifacol.2022.11.023. URL `https://www.sciencedirect.com/science/article/pii/S2405896322026519`. 25th IFAC Symposium on Mathematical Theory of Networks and Systems MTNS 2022.

Kaiming He and Jian Sun. Deep Residual Learning for Image Recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. doi: 10.1109/CVPR.2016.90.

Harold Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24(6):417–441, 1933. ISSN 0022-0663. doi: 10.1037/h0071325.

Harold Hotelling. Relations between two sets of variates. *Biometrika*, 28(3-4):321–377, 1936. ISSN 0006-3444. doi: 10.1093/biomet/28.3-4.321.

Aapo Hyvarinen. Fast and robust fixed-point algorithms for independent component analysis. *IEEE transactions on Neural Networks*, 10(3):626–634, 1999.

Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. *Advances in neural information processing systems*, 26, 2013.

Mario Lezcano-Casado. Trivializations for gradient-based optimization on manifolds. *Advances in Neural Information Processing Systems*, 32, 2019.

Mario Lezcano-Casado and David Martínez-Rubio. Cheap orthogonal constraints in neural networks: A simple parametrization of the orthogonal and unitary group. In *International Conference on Machine Learning*, pages 3794–3803. PMLR, 2019.

Qiyang Li, Saminul Haque, Cem Anil, James Lucas, Roger B Grosse, and Jörn-Henrik Jacobsen. Preventing gradient attenuation in Lipschitz constrained convolutional networks. *Advances in neural information processing systems*, 32, 2019a.

Shuai Li, Kui Jia, Yuxin Wen, Tongliang Liu, and Dacheng Tao. Orthogonal deep neural networks. *IEEE transactions on pattern analysis and machine intelligence*, 43(4):1352–1368, 2019b.

Xin Liu, Nachuan Xiao, and Ya-xiang Yuan. A penalty-free infeasible approach for a class of nonsmooth optimization problems over the Stiefel manifold. *Journal of Scientific Computing*, 99(2):30, 2024.

Thomas Moreau, Mathurin Massias, Alexandre Gramfort, Pierre Ablin, Pierre-Antoine Bannier, Benjamin Charlier, Mathieu Dagréou, Tom Dupré la Tour, Ghislain Durif, Cassio F. Dantas, Quentin Klopfenstein, Johan Larsson, En Lai, Tanguy Lefort, Benoit Malézieux, Badr Moufad, Binh T. Nguyen, Alain Rakotomamonjy, Zaccharie Ramzi, Joseph Salmon, and Samuel Vaiter. Benchopt: Reproducible, efficient and collaborative optimization benchmarks. In *NeurIPS*, 2022. URL `https://arxiv.org/abs/2206.13424`.

Sashank J Reddi, Suvrit Sra, Barnabás Póczos, and Alex Smola. Fast incremental method for nonconvex optimization. *arXiv preprint arXiv:1603.06159*, 2016.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

Sholom Schechtman, Daniil Tiapkin, Michael Muehlebach, and Eric Moulines. Orthogonal Directions Constrained Gradient Method: from non-linear equality constraints to Stiefel manifold. *arXiv preprint arXiv:2303.09261*, 2023.

Mark Schmidt, Nicolas Le Roux, and Francis Bach. Minimizing finite sums with the stochastic average gradient. *Mathematical Programming*, 162:83–112, 2017.

Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *Information and Software Technology*, 51(4):769–784, September 2014.

Sahil Singla and Soheil Feizi. Skew orthogonal convolutions. In *International Conference on Machine Learning*, pages 9756–9766. PMLR, 2021.

Fabian J. Theis, Thomas P. Cason, and P.-A. Absil. Soft dimension reduction for ICA by joint diagonalization on the Stiefel manifold. In *Proc. ICA 2009*, volume 5441 of *LNCS*, pages 354–361, Paraty, Brazil, 2009. Springer. doi: 10.1007/978-3-642-00599-2_45.

Nilesh Tripuraneni, Nicolas Flammarion, Francis Bach, and Michael I Jordan. Averaging stochastic gradient descent on Riemannian manifolds. In *Conference On Learning Theory*, pages 650–687. PMLR, 2018.

Jiayun Wang, Yubei Chen, Rudrasis Chakraborty, and Stella X. Yu. Orthogonal Convolutional Neural Networks. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11502–11512, Seattle, WA, USA, June 2020. IEEE. ISBN 978-1-72817-168-5. doi: 10.1109/CVPR42600.2020.01152.

Zaiwen Wen and Wotao Yin. A feasible method for optimization with orthogonality constraints. *Math. Program.*, 2012. ISSN 0025-5610. doi: 10.1007/s10107-012-0584-1.

Di Xie, Jiang Xiong, and Shiliang Pu. All you need is beyond a good init: Exploring better solution for training extremely deep convolutional neural networks with orthonormality and modulation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6176–6185, 2017.

Aston Zhang, Alvin Chan, Yi Tay, Jie Fu, Shuohang Wang, Shuai Zhang, Huajie Shao, Shuochao Yao, and Roy Ka-Wei Lee. On orthogonality constraints for transformers. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, volume 2, pages 375–382. Association for Computational Linguistics, 2021.

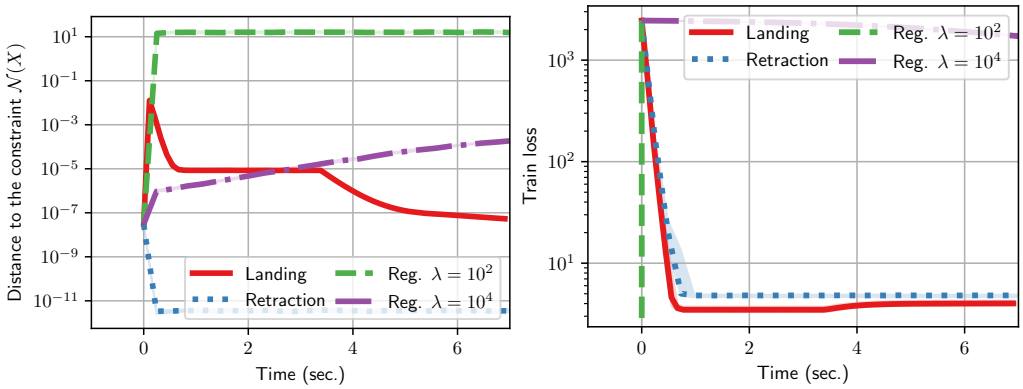
Hongyi Zhang, Sashank J Reddi, and Suvrit Sra. Riemannian SVRG: Fast stochastic optimization on Riemannian manifolds. *Advances in Neural Information Processing Systems*, 29, 2016.

Pan Zhou, Xiao-Tong Yuan, and Jiashi Feng. Faster first-order methods for stochastic nonconvex optimization on Riemannian manifolds. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 138–147. PMLR, 2019.

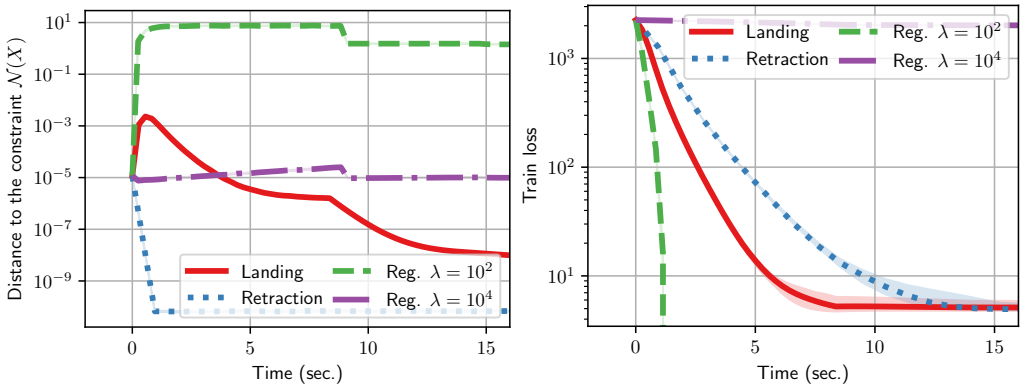
Ralf Zimmermann and Knut Hüper. Computing the Riemannian logarithm on the Stiefel manifold: Metrics, methods, and performance. *SIAM Journal on Matrix Analysis and Applications*, 43(2):953–980, 2022. doi: 10.1137/21M1425426.

Appendix A. Further numerical experiments

Figure 6 shows the convergence plots of the landing method, Riemannian stochastic gradient descent (RGD) with QR retractions, and the stochastic gradient descent with the ℓ_2 penalty (marked as "Reg") applied to the online PCA as described in Section 4. The landing compared to the stochastic RGD converges faster to the critical point in terms of computational time, especially with larger $p = 500$.



(a) $n = 5000, p = 100$



(b) $n = 5000, p = 500$

Figure 6: Convergence plots of the tested methods on online PCA.

Figure 7 shows the convergence of training of VGG16 on CIFAR-10 where each convolutional kernel is restricted to be orthogonal with the `GeoTorch` library using trivializations (Lezcano-Casado, 2019). We see that the straightforward implementation of the method takes around 10 hours to complete, which is roughly 14 times slower compared to the other methods tested in Section 3.

Appendix B. Proofs

B.1 Proof of Lemma 1

Proof Letting $\sigma_1, \dots, \sigma_p$ the singular values of X, we have $\|X^\top X - I_p\|^2 = \sum_{i=1}^p (\sigma_i^2 - 1)^2$. Hence, if $\|X^\top X - I_p\|^2 \leq \varepsilon^2$, we must have that all singular values satisfy $|\sigma^2 - 1| \leq \varepsilon$,

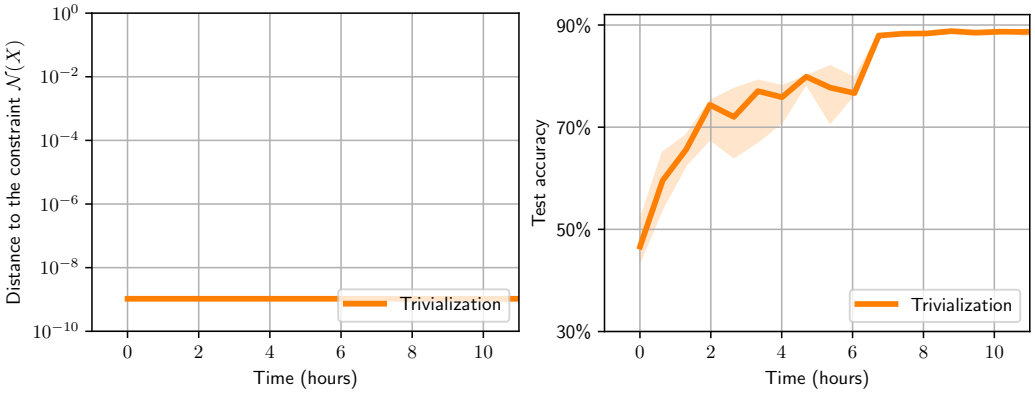


Figure 7: Training of VGG16 on CIFAR10 with orthogonal convolutional kernels with `GeoTorch` library using trivializations.

which proves the result. ■

B.2 Proof of Proposition 2

Proof Since the field (13) is the sum of two orthogonal terms, we have $\|F(X,A)\|^2 = \|AX\|^2 + \lambda^2\|X(X^\top X - I_p)\|^2$. Using the bound on the singular values of X from Lemma 1, we have $4(1-\varepsilon)\mathcal{N}(X) \leq \|X(X^\top X - I_p)\|^2 \leq 4(1+\varepsilon)\mathcal{N}(X)$, which concludes the proof. ■

B.3 Proof of Lemma 3

Proof Let the residual at X be $\Delta = X^\top X - I_p$ and the landing field be $F = F(X,A)$. The residual $\tilde{\Delta} = \tilde{X}^\top \tilde{X} - I_p$ at the updated $\tilde{X}$ can be expressed as

$$\begin{aligned}
\tilde{\Delta} = \tilde{X}^\top \tilde{X} - I_p &= (X - \eta F)^\top (X - \eta F) - I_p && (28)\\
&= X^\top X - \eta(X^\top F + F^\top X) + \eta^2 F^\top F - I_p && (29)\\
&= \Delta - \eta\left(X^\top(AX + \lambda X\Delta) + (-X^\top A + \lambda\Delta X^\top)X\right) + \eta^2 F^\top F && (30)\\
&= \Delta - 2\eta\lambda(\Delta + \Delta^2) + \eta^2 F^\top F, && (31)
\end{aligned}$$

where for the last equality we use that $X^\top X = \Delta + I_p$. By the triangle inequality and norm submultiplicativity, we can bound the Frobenius norm as

$$\|\tilde{\Delta}\| \leq (1 - 2\eta\lambda)\|\Delta\| + 2\eta\lambda\|\Delta\|^2 + \eta^2\|F\|^2, \tag{32}$$

when $\eta < 1/(2\lambda)$. Rewriting the above using scalar notation of $d = \|\Delta\|$ and $g = \|F\|$ and fixed $\lambda > 0$, we wish to find an interval for η such that

$$\eta^2 g^2 + 2\eta\lambda d(d-1) + d \leq \varepsilon, \tag{33}$$

which in turn ensures that $\|\tilde{\Delta}\| \leq \varepsilon$. For $\varepsilon \geq d$, i.e., when X is in the safe region, we can discard the negative root of the quadratic inequality in (33) resulting in the upper bound

in (14). Note that if $d \geq 1$, the upper bound on the step size is not positive when on the boundary $d = \varepsilon$. To prevent this case, we need to have $\varepsilon < 1$. ■

B.4 Proof of Lemma 4

Proof Let X be with distance $d > 0$, i.e.,

$$\|X^\top X - I_p\|_F^2 = \sum_{i=1}^{p} (\sigma_i^2 - 1)^2 = d^2,$$

where σ_i are the singular values of X, which, as a result, must be bounded as: $\sigma_i^2 \in [1-d, 1+d]$. Consider the bound on the normalizing component

$$\|\nabla \mathcal{N}(X)\|_F^2 = \|X(X^\top X - I_p)\|_F^2 = \sum_{i=1}^{p} \sigma_i^2 (\sigma_i^2 - 1)^2,$$

implying that

$$(1-d)d^2 \leq \|\nabla \mathcal{N}(X)\|_F^2 \leq (1+d)d^2.$$

By the orthogonality of the two components in the landing field, we have that

$$\|F(X, A)\|_F^2 = \|AX\|_F^2 + \lambda^2 \|\nabla \mathcal{N}(X)\|_F^2. \tag{34}$$

We proceed to lower-bound the first term in the minimum stated in (14) to make it independent of X. In the following, we denote $a = \|AX\|_F$ and $g = \|F(X, A)\|_F$

$$\frac{1}{g^2}\left(\lambda d(1-d) + \sqrt{\lambda^2 d^2 (1-d)^2 + g^2(\varepsilon - d)}\right) \tag{35}$$

$$\geq \frac{1}{a^2 + \lambda^2(1+d)d^2}\left(\lambda d(1-d) + \sqrt{\lambda^2 d^2 (1-d)^2 + (a^2 + \lambda^2(1-d)^2 d^2)(\varepsilon - d)}\right) \tag{36}$$

$$\geq \frac{1}{a^2 + \lambda^2(1+d)d^2}\left(\lambda d(1-d)\left(1 + \sqrt{\frac{1+\varepsilon-d}{2}}\right) + a\sqrt{\frac{\varepsilon - d}{2}}\right) \tag{37}$$

$$\geq \frac{\lambda(1-\varepsilon)d + a\sqrt{(\varepsilon-d)/2}}{a^2 + \lambda^2(1+\varepsilon)d^2} := K(\lambda, \varepsilon, a, d) \tag{38}$$

where in (36) we used the bound from (34) to g and that $(1-d)^2 d^2 \leq (1-d)d^2$ for $d < 1$, in (37) we used the fact that for any $x, y \geq 0$ we have that $\sqrt{x^2 + y^2} \geq (x+y)/\sqrt{2}$, and in (38) we used that $d \leq \varepsilon < 1$.

We now define

$$Q(\lambda, \varepsilon, \tilde{a}) = \inf_{a \in]0, \tilde{a}], d \in]0, \varepsilon]} K(\lambda, \varepsilon, a, d) \ .$$

We have that for all $\lambda > 0, \varepsilon > 0$ and $\tilde{a} > 0$, $Q(\lambda, \varepsilon, \tilde{a}) > 0$ since $K > 0$ and 0 cannot be attained on the boundaries where $a \to 0$ or $d \to 0$. Indeed, as $a \to 0$, we have $K(\lambda, \varepsilon, a, d) = \frac{1-\varepsilon}{\lambda(1+\varepsilon)d} + O(a)$, and as $d \to 0$ we have $K(\lambda, \varepsilon, a, d) = \frac{\sqrt{\varepsilon}}{a\sqrt{2}} + O(d)$, both of these limits are bounded away from 0.

As a result, the upper bound on the safeguard step size for all X with the ε distance cannot decrease below

$$\eta(X) \geq \min \eta^*(\tilde{a}, \varepsilon, \lambda) := \left\{ Q(\lambda, \varepsilon, \tilde{a}),\ \frac{1}{2\lambda} \right\}$$

■

B.5 Proof of Proposition 5

Proof The proof is a simple recursion. The property that $X_0 \in \mathrm{St}(p,n)^\varepsilon$ holds by assumption. Then, assuming that $X_0, \dots, X_k \in \mathrm{St}(p,n)^\varepsilon$, we have by assumption that A_k is such that $\|A_k X_k\| \leq \tilde{a}$. It follows that $X_{k+1} \in \mathrm{St}(p,n)^\varepsilon$ following Lemma 4, which concludes the recursion. ■

Lemma 16 (Jacobian of $\Phi(X)$) *Let $\Phi(X) = \mathrm{sym}(\nabla f(X)^\top X) : \mathbb{R}^{n\times p} \to \mathbb{R}^{p\times p}$, let $\mathcal{J}_X(\Phi)$ denote its derivative at X, and let $\mathcal{J}_X(\Phi)^*[\dot{X}]$ denote its adjoint in the sense of the Frobenius inner product at X applied to $\dot{X}$. Let $\mathrm{vec}(\cdot) : \mathbb{R}^{m\times n} \to \mathbb{R}^{mn}$ denote the vectorization operation, and let H_X denote the matrix representation of the Hessian of f at X; namely, $H_X \in \mathbb{R}^{np\times np}$ such that, for all $\dot{X} \in \mathbb{R}^{n\times p}$, $H_X[\mathrm{vec}\left(\dot{X}\right)] = \mathrm{vec}\left(\mathrm{D}(\nabla f)(X)[\dot{X}]\right)$. Then*

$$\mathrm{vec}\left(\mathcal{J}_X(\Phi)^*[X^\top X - I_p]\right) = H_X \mathrm{vec}(\nabla \mathcal{N}(X)) + \mathrm{vec}\left(\nabla f(X)(X^\top X - I)\right).$$

Proof Let $\mathrm{D}F(X)[\dot{X}] = \lim_{t\to 0}(F(X+t\dot{X}) - F(X))/t$ denote the derivative of F at X along $\dot{X}$. We have $\mathrm{D}\Phi(X)[\dot{X}] = \mathrm{sym}(\left(\mathrm{D}(\nabla f)(X)[\dot{X}]\right)^\top X + \nabla f(X)^\top \dot{X})$. Hence, for all $\dot{X} \in \mathbb{R}^{n\times p}$ and $Z \in \mathbb{R}^{p\times p}$, it holds that

$$\begin{aligned}
\langle \dot{X}, (\mathrm{D}\Phi)^*(X)[Z] \rangle &= \left\langle \mathrm{D}\Phi(X)[\dot{X}],\ Z \right\rangle && (39)\\
&= \left\langle \mathrm{sym}(\left(\mathrm{D}(\nabla f)(X)[\dot{X}]\right)^\top X + \nabla f(X)^\top \dot{X}),\ Z \right\rangle && (40)\\
&= \left\langle \left(\mathrm{D}(\nabla f)(X)[\dot{X}]\right)^\top X + \nabla f(X)^\top \dot{X},\ \mathrm{sym}(Z) \right\rangle && (41)\\
&= \left\langle \mathrm{D}(\nabla f)(X)[\dot{X}],\ X\mathrm{sym}(Z) \right\rangle + \left\langle \dot{X},\ \nabla f(X)\mathrm{sym}(Z) \right\rangle && (42)\\
&= \left\langle \dot{X},\ \mathrm{D}(\nabla f)(X)[X\mathrm{sym}(Z)] \right\rangle + \left\langle \dot{X},\ \nabla f(X)\mathrm{sym}(Z) \right\rangle. && (43)
\end{aligned}$$

Hence

$$(\mathrm{D}\Phi)^*(X)[Z] = \mathrm{D}(\nabla f)(X)[X\mathrm{sym}(Z)] + \nabla f(X)\mathrm{sym}(Z),$$

where $(\mathrm{D}\Phi)^*(X)$ is the adjoint of the Jacobian of Φ at X, and $\mathrm{D}(\nabla f)(X)[\cdot] : \mathbb{R}^{n\times p} \to \mathbb{R}^{n\times p}$ is the Hessian operator of f at X. For $Z = X^\top X - I_p$, this yields the result of Lemma 16. ■

B.6 Proof of Proposition 6

Proof Computing the gradient of $\mathcal{L}(X)$ has four terms

$$\nabla\mathcal{L}(X) = \nabla f(X) - \frac{1}{2}\mathcal{J}_X\left(\Phi\right)^*\left[X^\top X - I_p\right] - X\mathrm{sym}(X^\top\nabla f(X)) + \mu\nabla\mathcal{N}(X), \tag{44}$$

where $\mathcal{J}_X\left(\Phi\right)^*\left[X^\top X - I_p\right]$ denotes the adjoint of the Jacobian in the sense of the Frobenius inner product of $\Phi(X) = \mathrm{sym}(X^\top\nabla f(X))$ in X evaluated in the direction $X^\top X - I_p$. We proceed by expressing the inner product of the landing field $\Lambda(X)$ with each of the four terms of $\nabla\mathcal{L}(X)$ in (44) separately.

The inner product between the first term of (44) and the landing field $\Lambda(X)$ is

$$\begin{aligned}
\langle\Lambda(X),\, \nabla f(X)\rangle &= \left\langle \mathrm{skew}(\nabla f(X)X^\top)X + \lambda X(X^\top X - I_p),\, \nabla f(X)\right\rangle && (45)\\
&= \left\langle \mathrm{skew}(\nabla f(X)X^\top),\, \nabla f(X)X^\top\right\rangle + \lambda\left\langle X^\top X - I_p,\, X^\top\nabla f(X)\right\rangle && (46)\\
&= \|\mathrm{skew}(\nabla f(X)X^\top)\|_F^2 + \lambda\left\langle \mathrm{sym}(X^\top\nabla f(X)),\, X^\top X - I_p\right\rangle, && (47)
\end{aligned}$$

where we used the fact that the inner product of a skew-symmetric and a symmetric matrix is zero.

We can express the inner product between the Jacobian using Lemma 16 in the second term of (44) as

$$\begin{aligned}
\left\langle \Lambda(X),\, -\frac{1}{2}\mathcal{J}_X\left(\Phi\right)^*\left[X^\top X - I_p\right]\right\rangle = &-\frac{1}{2}\lambda\left\langle H_X\mathrm{vec}\left(\nabla\mathcal{N}(X)\right),\, \mathrm{vec}\left(\nabla\mathcal{N}(X)\right)\right\rangle\\
&-\frac{1}{2}\left\langle H_X\mathrm{vec}\left(\nabla\mathcal{N}(X)\right),\, \mathrm{vec}\left(\mathrm{grad}f(X)\right)\right\rangle\\
&-\frac{1}{2}\left\langle \nabla f(X)^\top\mathrm{grad}f(X),\, X^\top X - I_p\right\rangle\\
&-\frac{1}{2}\lambda\left\langle \mathrm{sym}(X^\top\nabla f(X)),\, (X^\top X - I_p)^2\right\rangle,
\end{aligned} \tag{48}$$

where $\mathrm{vec}\left(\cdot\right) : \mathbb{R}^{n\times p} \to \mathbb{R}^{np}$ vectorizes a matrix and H_X denote the matrix representation of the Hessian of f at X; namely, $H_X \in \mathbb{R}^{np\times np}$ such that, for all $\dot{X} \in \mathbb{R}^{n\times p}$, $H_X[\mathrm{vec}\left(\dot{X}\right)] = \mathrm{vec}\left(\mathrm{D}(\nabla f)(X)[\dot{X}]\right)$.

The third term is

$$\left\langle \Lambda(X),\, -X\mathrm{sym}(X^\top\nabla f(X))\right\rangle = \left\langle \mathrm{skew}(\nabla f(X)X^\top)X + \lambda\nabla\mathcal{N}(X),\, -X\mathrm{sym}(X^\top\nabla f(X))\right\rangle \tag{49}$$

$$= -\lambda\left\langle \mathrm{sym}(X^\top\nabla f(X)),\, X^\top X(X^\top X - I_p)\right\rangle, \tag{50}$$

where the inner product with the skew-symmetric matrix in the first term is zero.

The inner product of the fourth term and the landing field is

$$\langle\Lambda(X),\, \mu\nabla\mathcal{N}(X)\rangle = \lambda\mu\|\nabla\mathcal{N}(X)\|_F^2 \tag{51}$$

by orthogonality of the two components of the landing field.

Adding all the four terms expressed in (47), (48), (50), and (51) together gives

$$\langle \nabla \mathcal{L}(X),\, \Lambda(X) \rangle = \|\mathrm{skew}(\nabla f(X) X^\top)\|_F^2 \tag{52}$$

$$+ \lambda \left\langle \left(\mu I_{np} - \frac{1}{2} H_X \right) \mathrm{vec}\left(\nabla \mathcal{N}(X)\right),\, \mathrm{vec}\left(\nabla \mathcal{N}(X)\right) \right\rangle \tag{53}$$

$$- \lambda \frac{3}{2} \left\langle (X^\top X - I_p)^2,\, \mathrm{sym}(X^\top \nabla f(X)) \right\rangle \tag{54}$$

$$- \frac{1}{2} \left\langle H_X \mathrm{vec}\left(\nabla \mathcal{N}(X)\right),\, \mathrm{vec}\left(\mathrm{grad} f(X)\right) \right\rangle \tag{55}$$

$$- \frac{1}{2} \left\langle \nabla f(X)^\top \mathrm{grad} f(X),\, X^\top X - I_p \right\rangle, \tag{56}$$

where the first line in (52) comes from the first term of (47), the second line (53) is a combination of the first term in (48) and (51), the third line in (54) comes from the second term in (47), the third term in (48), and (48).

We will bound lower bound each term separately. The term in (53) is lower bounded as

$$\lambda \left\langle \left(\mu I_{np} - \frac{1}{2} H_X \right) \mathrm{vec}\left(\nabla \mathcal{N}(X)\right),\, \mathrm{vec}\left(\nabla \mathcal{N}(X)\right) \right\rangle \geq \lambda \left(\mu - \frac{L}{2} \right) \|\nabla \mathcal{N}(X)\|_F^2 \tag{57}$$

$$\geq 4\lambda \left(\mu - \frac{L}{2} \right) \sigma_p^2 \mathcal{N}(X), \tag{58}$$

where σ_p is the smallest singular value of X and L is the Lipschitz constant of $\nabla f(X)$.

The term in (54) is lower bounded using the Cauchy-Schwarz inequality as

$$-\lambda \frac{3}{2} \left\langle (X^\top X - I_p)^2,\, \mathrm{sym}(X^\top \nabla f(X)) \right\rangle \geq -6\lambda \mathcal{N}(X) \|\mathrm{sym}(X^\top \nabla f(X))\|_F, \tag{59}$$

by the fact that $\|X^\top X - I_p\|_F^2 = 4\mathcal{N}(X)$.

The term in (55) is lower bounded as

$$-\frac{1}{2} \left\langle H_X \mathrm{vec}\left(\nabla \mathcal{N}(X)\right),\, \mathrm{vec}\left(\mathrm{grad} f(X)\right) \right\rangle \geq -\frac{L}{2} \|X(X^\top X - I_p)\|_F \|\mathrm{grad} f(X)\|_F \tag{60}$$

$$\geq -L\sigma_1 \sqrt{\mathcal{N}(X)} \|\mathrm{grad} f(X)\|_F, \tag{61}$$

by the fact that the operator norm of $\|X\|_2 = \sigma_1$ and $\|X^\top X - I_p\|_F = 2\sqrt{\mathcal{N}(X)}$.

The term in (56) is lower bounded as

$$-\frac{1}{2} \left\langle \nabla f(X)^\top \mathrm{grad} f(X),\, X^\top X - I_p \right\rangle \geq -\frac{1}{2} \|\nabla f(X)\|_F \|\mathrm{grad} f(X)\|_F \|X^\top X - I_p\|_F \tag{62}$$

$$\geq -L' \sqrt{\mathcal{N}(X)} \|\mathrm{grad} f(X)\|_F, \tag{63}$$

where L' is such that $\|\nabla f(X)\| \leq L'$ for all $X \in \mathrm{St}(p,n)^\varepsilon$.

Now we bound the terms in (61) and (63) together

$$-\frac{1}{2} \left\langle H_X \mathrm{vec}\left(\nabla \mathcal{N}(X)\right),\, \mathrm{vec}\left(\mathrm{grad} f(X)\right) \right\rangle - \frac{1}{2} \left\langle \nabla f(X)^\top \mathrm{grad} f(X),\, X^\top X - I_p \right\rangle$$

$$\geq -(L' + L\sigma_1) \sqrt{\mathcal{N}(X)} \|\mathrm{grad} f(X)\| \tag{64}$$

$$\geq -\frac{1}{2} (L' + L\sigma_1) \left(\beta \mathcal{N}(X) + \beta^{-1} \|\mathrm{grad} f(X)\|^2 \right), \tag{65}$$

where in the first inequality we used the previously derived bounds in (61) and (63), and the second inequality is a consequence of the AG inequality $\sqrt{xy} \leq (x+y)/2$ with $x = \beta\mathcal{N}(X)$ and $y = \beta^{-1}\|\mathrm{grad} f(X)\|^2$ for an arbitrary $\beta > 0$, which will be specified later.

Adding the bounds in (58), (59), and (65) together, we have a lower bound as

$$\langle \nabla\mathcal{L}(X),\, \Lambda(X)\rangle \geq \|\mathrm{grad} f(X)\|_F^2 \left(\sigma_1^{-2} - \frac{1}{2}(L' + L\sigma_1)\beta^{-1}\right) \tag{66}$$

$$+ \mathcal{N}(X)\left(4\lambda\left(\mu - \frac{L}{2}\right)\sigma_p^2 - 6\lambda\|\mathrm{sym}(X^\top \nabla f(X))\|_F - \frac{1}{2}(L' + L\sigma_1)\beta\right) \tag{67}$$

where we used that $\|\mathrm{skew}(\nabla f(X)X^\top)\|_F \geq \sigma_1^{-1}\|\mathrm{grad} f(X)\|_F$. Choosing $\beta = \sigma_1^2 \frac{L'+L\sigma_1}{2-\sigma_1^2}$ and bounding $\|\mathrm{sym}(X^\top \nabla f(X))\|_F \leq s$ we have

$$\langle \nabla\mathcal{L}(X),\, \Lambda(X)\rangle \geq \frac{1}{2}\|\mathrm{grad} f(X)\|_F^2 + \mathcal{N}(X)\left(4\lambda\left(\mu - \frac{L}{2}\right)\sigma_p^2 - 6\lambda s - \frac{1}{2}\sigma_1^2\frac{(L' + L\sigma_1)^2}{2-\sigma_1^2}\right) \tag{68}$$

$$\geq \frac{1}{2}\|\mathrm{grad} f(X)\|_F^2 + \mathcal{N}(X)\left(4\lambda\left(\mu - \frac{L}{2}\right)(1-\varepsilon) - 6\lambda s - \frac{1}{2}\hat{L}^2(1+\varepsilon)\frac{(2\sqrt{1+\varepsilon})^2}{2-(1+\varepsilon)}\right) \tag{69}$$

$$\geq \frac{1}{2}\|\mathrm{grad} f(X)\|_F^2 + \mathcal{N}(X)\left(4\lambda\left(\mu - \frac{L}{2}\right)(1-\varepsilon) - 6\lambda s - 2\hat{L}^2\frac{(1+\varepsilon)^2}{1-\varepsilon}\right), \tag{70}$$

where in the second line we define $\hat{L} = \max\{L, L'\}$ and we used that $\sqrt{1+\varepsilon} \geq 1$. The coefficient in front of the distance term $\mathcal{N}(X)$ in (70) is lower bounded by $\lambda\mu$ for

$$\mu \geq \frac{2}{3-4\varepsilon}\left(L(1-\varepsilon) + 3s + \hat{L}^2\frac{(1+\varepsilon)^2}{\lambda(1-\varepsilon)}\right). \tag{71}$$

■

B.7 Proof of Proposition 7

Proof We have

$$\frac{1}{2}\|\mathrm{grad} f(X)\|^2 + \nu\mathcal{N}(X) \geq \rho(\|\mathrm{grad} f(X)\|^2 + 4\lambda^2(1+\varepsilon)\mathcal{N}(X)) \tag{72}$$

$$\geq \rho\|\Lambda(X)\|^2 \ , \tag{73}$$

where the last inequality comes from Proposition 2. ■

B.8 Proof of Proposition 8

Proof We start by considering the smoothness constant of $\mathcal{N}$. We have $\nabla\mathcal{N}(X) = X(X^\top X - I_p)$, hence we find that its Hessian is such that in a direction $E \in \mathbb{R}^{n\times p}$:

$$\nabla^2\mathcal{N}(X)[E] = E(X^\top X - I_p) + X(E^\top X + X^\top E) \ .$$

We can then bound crudely using the triangular inequality

$$\begin{aligned}\|\nabla^2\mathcal{N}(X)[E]\| &\leq \|E(X^\top X - I_p)\| + \|XE^\top X\| + \|XX^\top E\| &(74)\\ &\leq \varepsilon\|E\| + 2(1+\varepsilon)\|E\| &(75)\\ &\leq (2+3\varepsilon)\|E\| \ . &(76)\end{aligned}$$

This implies that all the absolute values of the eigenvalues of $\nabla^2\mathcal{N}(X)$ are bounded by $2+3\varepsilon$, so that $\mathcal{N}$ is $(2+3\varepsilon)$-smooth. The result follows since the sum of smooth functions is smooth. ■

B.9 Proof of Proposition 9

Proof Since we use the safeguard step size, the iterates remain in $\mathrm{St}(p,n)^\varepsilon$. Using the L_g-smoothness of $\mathcal{L}$ (Proposition 8), we get

$$\mathcal{L}(X_{k+1}) \leq \mathcal{L}(X_k) - \eta\langle\Lambda(X_k), \nabla\mathcal{L}(X_k)\rangle + \frac{L_g\eta^2}{2}\|\Lambda(X_k)\|^2 \ .$$

Using Proposition 6 and Proposition 2, we get

$$\mathcal{L}(X_{k+1}) \leq \mathcal{L}(X_k) - (\frac{\eta}{2} - \frac{\eta^2 L_g}{2})\|\mathrm{grad} f(X_k)\|^2 - (\eta\nu - 2\lambda^2\eta^2 L_g(1+\varepsilon))\mathcal{N}(X_k) \ .$$

Using the hypothesis on η, we have both $\frac{\eta}{2} - \frac{\eta^2 L_g}{2} \geq \frac{\eta}{4}$ and $\eta\nu - 2\lambda^2\eta^2 L_g(1+\varepsilon) \geq \frac{\eta\nu}{2}$. We, therefore, obtain the inequality

$$\frac{\eta}{4}\|\mathrm{grad} f(X_k)\|^2 + \frac{\eta\nu}{2}\mathcal{N}(X_k) \leq \mathcal{L}(X_k) - \mathcal{L}(X_{k+1}) \ .$$

Summing these terms gives

$$\frac{\eta}{4}\sum_{k=1}^{K}\|\mathrm{grad} f(X_k)\|^2 + \frac{\eta\nu}{2}\sum_{k=1}^{K}\mathcal{N}(X_k) \leq \mathcal{L}(X_0) - \mathcal{L}(X_{K+1}) \leq \mathcal{L}(X_0) - \mathcal{L}^*, \tag{77}$$

which implies that we have both

$$\frac{\eta}{4}\sum_{k=1}^{K}\|\mathrm{grad} f(X_k)\|^2 \leq \mathcal{L}(X_0) - \mathcal{L}^* \ \text{ and } \ \frac{\eta\nu}{2}\sum_{k=1}^{K}\mathcal{N}(X_k) \leq \mathcal{L}(X_0) - \mathcal{L}^* \ . \tag{78}$$

These two inequalities then directly provide the result. ■

B.10 Proof of Proposition 11

Proof We use once again the smoothness of $\mathcal{L}$ and unbiasedness of $\Lambda_i(X_k)$ to get

$$\mathbb{E}_i[\mathcal{L}(X_{k+1})] \leq \mathcal{L}(X_k) - \eta_k \langle \Lambda(X_k), \nabla\mathcal{L}(X_k)\rangle + \frac{\eta_k^2 L_g}{2}\mathbb{E}[\|\Lambda_i(X_k)\|^2] \tag{79}$$

$$\leq \mathcal{L}(X_k) - \eta_k \langle \Lambda(X_k), \nabla\mathcal{L}(X_k)\rangle + \frac{\eta_k^2 L_g}{2}(B + \|\Lambda(X_k)\|^2) \tag{80}$$

$$\leq \mathcal{L}(X_k) - \eta_k(\frac{1}{2}\|\mathrm{grad} f(X_k)\|^2 + \nu\mathcal{N}(X_k)) \tag{81}$$

$$+ \frac{\eta_k^2 L_g}{2}(\|\mathrm{grad} f(X_k)\|^2 + 4\lambda^2(1+\varepsilon)\mathcal{N}(X_k)) + \frac{\eta_k^2 L_g B}{2} \tag{82}$$

$$\leq \mathcal{L}(X_k) - \frac{\eta_k - \eta_k^2 L_g}{2}\|\mathrm{grad} f(X_k)\|^2 - (\eta_k\nu - 2\eta_k^2 L_g\lambda^2(1+\varepsilon))\mathcal{N}(X_k) + \frac{\eta_k^2 L_g B}{2} \ . \tag{83}$$

Taking $\eta_k \leq \min(\frac{1}{2L_g}, \frac{\nu}{4L_g\lambda^2(1+\varepsilon)})$ simplifies the inequality to

$$\mathbb{E}_i[\mathcal{L}(X_{k+1})] \leq \mathcal{L}(X_k) - \frac{\eta_k}{4}\|\mathrm{grad} f(X_k)\|^2 - \frac{\eta_k\nu}{2}\mathcal{N}(X_k) + \frac{\eta_k^2 L_g B}{2} \ .$$

■

B.11 Proof of Proposition 12

Proof Taking expectations with respect to the past, and summing up the inequality in Proposition 11 gives, using $\sum_{k=0}^{K}(1+k)^{-1} \leq \log(K+1)$:

$$\frac{1}{4}\sum_{k=0}^{K}\eta_k\mathbb{E}[\|\mathrm{grad} f(X_k)\|^2] \leq \mathcal{L}(X_0) - \mathcal{L}^* + \frac{\eta_0^2 L_g B}{2}\log(K+1)$$

and

$$\frac{\nu}{2}\sum_{k=0}^{K}\eta_k\mathbb{E}[\mathcal{N}(X_k)] \leq \mathcal{L}(X_0) - \mathcal{L}^* + \frac{\eta_0^2 L_g B}{2}\log(K+1) \ .$$

Next, we use the bound $\inf_{k\leq K}\mathbb{E}[\|\mathrm{grad} f(X_k)\|^2] \leq \sum_{k=0}^{K}\eta_k\|\mathrm{grad}(X_k)\|^2 \times (\sum_{k=0}^{K}\eta_k)^{-1}$ and the fact that $\sum_{k=0}^{K}\eta_k \geq \eta_0\sqrt{K}$ to get

$$\inf_{k\leq K}\mathbb{E}[\|\mathrm{grad} f(X_k)\|^2] \leq 4\frac{\mathcal{L}(X_0) - \mathcal{L}^* + \frac{\eta_0^2 L_g B}{2}\log(K+1)}{\eta_0\sqrt{K}}$$

and

$$\inf_{k\leq K}\mathbb{E}[\mathcal{N}(X_k)] \leq 2\frac{\mathcal{L}(X_0) - \mathcal{L}^* + \frac{\eta_0^2 L_g B}{2}\log(K+1)}{\nu\eta_0\sqrt{K}} \ .$$

■

B.12 Proof of Proposition 13

Proof Just like for the previous proposition, we get by summing the descent lemmas:

$$\frac{1}{4K}\sum_{k=1}^{K}\mathbb{E}[\|\mathrm{grad} f(X_k)\|^2] + \frac{\nu}{2K}\sum_{k=1}^{K}\mathbb{E}[\mathcal{N}(X_k)] \leq \frac{\mathcal{L}(X_0)-\mathcal{L}^*}{\eta K} + \frac{\eta L_g B}{2} \ .$$

Taking $\eta = \eta_0 K^{-1/2}$ yields as advertised:

$$\inf_{k\leq K}\mathbb{E}[\|\mathrm{grad} f(X_k)\|^2] \leq \frac{4}{\sqrt{K}}\left(\frac{\mathcal{L}(X_0)-\mathcal{L}^*}{\eta_0} + \frac{\eta_0 L_g B}{2}\right)$$

and

$$\inf_{k\leq K}\mathbb{E}[\mathcal{N}(X_k)] \leq \frac{2}{\nu\sqrt{K}}\left(\frac{\mathcal{L}(X_0)-\mathcal{L}^*}{\eta_0} + \frac{\eta_0 L_g B}{2}\right) \ .$$

■

B.13 Proof of Proposition 14

Proof We define X_k^i as the matrix such that $\Phi_k^i = \nabla f(X_k^i)$, i.e., the last iterate for which the memory corresponding to sample i has been updated. As in the classical SAGA analysis, we will form a merit function combining the merit function $\mathcal{L}$ and the distance to the memory, defined as

$$S_k = \frac{1}{N}\sum_{j=1}^{N}\mathbb{E}[\|X_k - X_k^j\|^2] \ .$$

We also define the variance of the landing direction as

$$V_k = \frac{1}{N}\sum_{j=1}^{N}\|\Lambda_k^j\|^2 \ .$$

For short, we let $\Lambda_k = \Lambda(X_k)$.

Control of the distance to the memory Looking at an individual term j in the sum of S_k, we have

$$\begin{aligned}
\mathbb{E}_i[\|X_{k+1} - X_{k+1}^j\|^2] &= \mathbb{E}_i[\|X_k - \eta\Lambda_k^i - X_{k+1}^j\|^2] && (84)\\
&= \frac{1}{N}\Big(\sum_{i\neq j}^{N}\|X_k - \eta\Lambda_k^i - X_k^j\|^2 + \eta^2\|\Lambda_k^j\|^2\Big) && (85)\\
&= \mathbb{E}_i[\|X_k - \eta\Lambda_k^i - X_k^j\|^2] - \frac{1}{N}\big(\|X_k - \eta\Lambda_k^j - X_k^j\|^2 - \eta^2\|\Lambda_k^j\|^2\big) \ . && (86)
\end{aligned}$$

On the one hand, we have for the first term:

$$\begin{aligned}
\mathbb{E}_i[\|X_k - \eta\Lambda_k^i - X_k^j\|^2] &= \|X_k - X_k^j\|^2 - 2\eta\langle\Lambda_k, X_k - X_k^j\rangle + \eta^2 V_k && (87)\\
&\leq (1+\eta\beta)\|X_k - X_k^j\|^2 + \eta\beta^{-1}\|\Lambda_k\|^2 + \eta^2 V_k, && (88)
\end{aligned}$$

where we introduce $\beta > 0$ from Young's inequality to control the scalar product. For the second term, we obtain

$$\|X_k - \eta\Lambda_k^j - X_k^j\|^2 - \eta^2\|\Lambda_k^j\|^2 = \|X_k - X_k^j\|^2 - 2\eta\langle\Lambda_k^j, X_k - X_k^j\rangle \tag{89}$$

$$\geq (1-\gamma\eta)\|X_k - X_k^j\|^2 - \gamma^{-1}\eta\|\Lambda_k^j\|^2, \tag{90}$$

where once again we introduce $\gamma > 0$ from Young's inequality. Taking all of these inequalities together gives

$$\mathbb{E}_i[\|X_{k+1}-X_{k+1}^j\|^2] \leq (1-\frac{1}{N}+\eta\beta+N^{-1}\gamma\eta)\|X_k-X_k^j\|^2+\eta\beta^{-1}\|\Lambda_k\|^2+\eta^2V_k+N^{-1}\gamma^{-1}\eta\|\Lambda_k^j\|^2,$$

and averaging these for $j = 1 \dots N$ gives

$$S_{k+1} \leq (1 - \frac{1}{N} + \eta\beta + N^{-1}\gamma\eta)S_k + \eta\beta^{-1}\|\Lambda_k\|^2 + (\eta^2 + N^{-1}\gamma^{-1}\eta)V_k \ . \tag{91}$$

We choose $\beta = (4N\eta)^{-1}$ and $\gamma = (4\eta)^{-1}$, which finally gives, assuming $N > 4$:

$$S_{k+1} \leq (1 - \frac{1}{2N})S_k + 4N\eta^2\|\Lambda_k\|^2 + 2\eta^2V_k \ . \tag{92}$$

Control of the merit function The smoothness of the merit function gives once again

$$\mathbb{E}_i[\mathcal{L}(X_{k+1})] \leq \mathcal{L}(X_k) - \eta\langle\Lambda_k, \nabla\mathcal{L}(X_k)\rangle + \eta^2\frac{L_g}{2}V_k \tag{93}$$

$$\leq \mathcal{L}(X_k) - \eta\rho\|\Lambda_k\|^2 + \eta^2\frac{L_g}{2}V_k, \tag{94}$$

where the last inequality comes from Proposition 7.

Variance control We then control V_k. By the bias-variance decomposition, using the unbiasedness of Λ_k^j, we get

$$V_k = \|\Lambda_k\|^2 + \frac{1}{N}\sum_{j=1}^N \|\Lambda_k^j - \Lambda_k\|^2 \tag{95}$$

$$\leq \|\Lambda_k\|^2 + \frac{1}{N}\sum_{j=1}^N \|\mathrm{grad}f_i(X_k) - \mathrm{skew}(\Phi_i X_k^\top)X_k\|^2 \tag{96}$$

$$\leq \|\Lambda_k\|^2 + \frac{1}{N}\sum_{j=1}^N \|\mathrm{skew}((\nabla f(X_k) - \nabla f(X_k^i))X_k^\top)X_k\|^2 \tag{97}$$

$$\leq \|\Lambda_k\|^2 + L_f^2(1+\varepsilon)S_k, \tag{98}$$

where L_f is the smoothness constant of f.

Putting it together We get the two inequalities

$$S_{k+1} \leq (1 - \frac{1}{2N} + 2\eta^2 L_f^2(1+\varepsilon)^2)S_k + (4N+2)\eta^2\|\Lambda_k\|^2 \tag{99}$$

$$\mathbb{E}_i[\mathcal{L}(X_{k+1})] \leq \mathcal{L}(X_k) - (\eta\rho - \eta^2\frac{L_g}{2})\|\Lambda_k\|^2 + \eta^2\frac{L_g}{2}L_f^2(1+\varepsilon)^2 S_k \ . \tag{100}$$

The hypothesis on the step size simplifies these inequalities to

$$S_{k+1} \leq (1 - \frac{1}{4N})S_k + (4N+2)\eta^2\|\Lambda_k\|^2 \tag{101}$$

$$\mathbb{E}_i[\mathcal{L}(X_{k+1})] \leq \mathcal{L}(X_k) - \frac{1}{2}\eta\rho\|\Lambda_k\|^2 + \eta^2\frac{L_g}{2}L_f^2(1+\varepsilon)^2 S_k \ . \tag{102}$$

We now look for a decreasing quantity of the form $\mathcal{G}_k = \mathcal{L}(X_k) + cS_k$, and get

$$\mathbb{E}[G_{k+1}] \leq G_k - (\frac{1}{2}\eta\rho - c(4N+2)\eta^2)\|\Lambda_k\|^2 - (\frac{c}{4N} - \eta^2\frac{L_g}{2}L_f^2(1+\varepsilon)^2)S_k \ .$$

We take $c = 2N\eta^2 L_g L_f^2(1+\varepsilon)^2$ in order to cancel the last term. The term in front of $\|\Lambda_k\|^2$ becomes $\frac{1}{2}\eta\rho - 2N(4N+2)L_g L_f^2(1+\varepsilon)^2\eta^4$, which is lower bounded by $\frac{1}{4}\eta\rho$ with the last condition on the step size. We therefore get

$$\mathbb{E}[G_{k+1}] \leq G_k - \frac{1}{4}\eta\rho\|\Lambda_k\|^2 \ .$$

Taking expectations with respect to the whole past and summing yields

$$\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}[\|\Lambda_k\|^2] \leq \frac{4}{\eta\rho K}(\mathcal{L}(X_0) - \mathcal{L}^*),$$

which concludes the proof. ■

B.14 Proof of Proposition 15

Proof Let us write $X = QDW^\top$ a singular value decomposition of X. Then we find

$$g(X) = \langle U^\top QDW^\top V, \Sigma\rangle + \frac{\lambda}{4}\|D^2 - I_p\|^2$$

The factors Q and W appear only in the first term. Hence we will first consider its minimization. Letting $\tilde{U} = Q^\top U$ and $\tilde{V} = W^\top V$, Von Neumann's trace inequality gives that the first term verifies

$$\langle \tilde{U}^\top D\tilde{V}, \Sigma\rangle \geq -\langle D, \Sigma\rangle$$

This lower bound is attained e.g. for $\tilde{U} = -I_p$ and $\tilde{V} = I_p$, i.e. for $Q = -U$ and $W = V$. Note that any other choice of $\tilde{U}, \tilde{V}$ that minimizes the above inequality leads to the same solution X^*. We then fall back to the problem of minimizing

$$-\langle D, \Sigma\rangle + \frac{\lambda}{4}\|D^2 - I_p\|^2$$

with respect to D, which is a problem that is separable on the entries of D, so that the optimal D is $D = \mathrm{diag}(\sigma_i^*)$ with σ_i^* that minimizes the scalar function $x \to -\sigma_i x + \frac{\lambda}{4}(x^2-1)^2$. We then have $X^* = QDW^\top = -UDV^\top$.

The function $x \to -\sigma_i x + \frac{\lambda}{4}(x^2-1)^2$ is minimized for the value $\sigma_i^* > 1$ such that (cancelling its derivative) it holds

$$(\sigma_i^*)^3 - \sigma_i^* = \frac{\sigma_i}{\lambda} \ .$$

We see that as λ goes to $+\infty$ we have σ_i^* that goes to 1 (we recover that X^* tends to orthogonality), and using the implicit function theorem, we find the expansion

$$\sigma_i^* = 1 + \frac{\sigma_i}{2\lambda} + o(\frac{1}{\lambda}) \text{ for } \lambda \to +\infty.$$

Furthermore, letting $h(\sigma) = \sigma^3 - \sigma$, we have

$$h(1 + \frac{\sigma_i}{2\lambda}) = \frac{\sigma_i}{\lambda} + \frac{3\sigma_i^2}{4\lambda^2} + \frac{\sigma_i^3}{8\lambda^3} \geq \frac{\sigma_i}{\lambda}.$$

Since the function h is locally increasing around 1, we deduce that $\sigma_i^* = h^{-1}(\frac{\sigma_i}{\lambda}) \leq 1 + \frac{\sigma_i}{2\lambda}$. Hence, we find $\|X^* - X_{\mathrm{St}(p,n)}\| = \sqrt{\sum_{i=1}^p (\sigma_i^* - 1)^2} = \frac{1}{2\lambda}\sqrt{\sum_{i=1}^p \sigma_i^2} + o(\frac{1}{\lambda})$, which goes to 0 at a $1/\lambda$ rate.

Finally, the Hessian of g at the optimum is the linear operator H such that

$$\text{For all } E \in \mathbb{R}^{n\times p}, \ \ H[E] = \lambda \left(E((X^*)^\top X^* - I_p) + X^*(E^\top X^* + (X^*)^\top E) \right) \ .$$

Note that $(X^*)^\top X^* - I_p$ is a symmetric positive-definite matrix. Let us lower-bound the largest eigenvalue of H, by taking $E = X^*$. We find

$$\begin{aligned}
\langle X^*, H[X^*] \rangle &= \lambda \left(\langle X^*, X^*((X^*)^\top X^* - I_p) \rangle + 2 \langle X^*, X^*(X^*)^\top X^* \rangle \right) && (103)\\
&\geq 2\lambda \langle X^*, X^*(X^*)^\top X^* \rangle && (104)\\
&\geq 2\lambda (\|X^*\|^2 + \langle X^*, X^*((X^*)^\top X^* - I_p) \rangle) && (105)\\
&\geq 2\lambda \|X^*\|^2 && (106)
\end{aligned}$$

where for the first and third inequality, we use the positive-definiteness of $(X^*)^\top X^* - I_p$.

We conclude that $\lambda_{\max}(H) \geq \frac{\langle X^*, H[X^*] \rangle}{\|X^*\|^2} \geq 2\lambda$. Similarly, let us upper-bound the smallest eigenvalue of H, by taking any E such that $E^\top X^* + (X^*)^\top E = 0$. We find

$$\begin{aligned}
\langle E, H[E] \rangle &= \lambda \langle E, E((X^*)^\top X^* - I_p) \rangle && (107)\\
&\leq \lambda \|(X^*)^\top X^* - I_p\|_2 \|E\|^2 && (108)\\
&\leq (\sigma_p + \frac{\sigma_p^2}{4\lambda}) \|E\|^2 && (109)
\end{aligned}$$

where we use the inequality $1 \leq \sigma_i^* \leq 1 + \frac{\sigma_i}{2\lambda}$ in order to upper bound $\|(X^*)^\top X^* - I_p\|_2 = (\sigma_p^*)^2 - 1$ by $\frac{\sigma_p}{\lambda} + \frac{\sigma_p^2}{4\lambda^2}$. As a consequence, we have $\lambda_{\min}(H) \leq \frac{\langle E, H[E]\rangle}{\|E\|^2} \leq \sigma_p + \frac{\sigma_p^2}{4\lambda}$.

Combining the inequalities on the eigenvalues, we find that the conditioning of H is lower-bounded by:

$$\frac{\lambda_{\max}(H)}{\lambda_{\min}(H)} \geq \frac{2\lambda}{\sigma_p + \frac{\sigma_p^2}{4\lambda}} = \frac{2\lambda}{\sigma_p} + o(\lambda) \text{ for } \lambda \to +\infty \ .$$

■