# The Proxy Step-size Technique for Regularized Optimization on the Sphere Manifold

Fang Bai, *Member, IEEE,* Adrien Bartoli

**Abstract**—We give an effective solution to the regularized optimization problem $g(\boldsymbol{x}) + h(\boldsymbol{x})$, where $\boldsymbol{x}$ is constrained on the unit sphere $\|\boldsymbol{x}\|_2 = 1$. Here $g(\cdot)$ is a smooth cost with Lipschitz continuous gradient within the unit ball $\{\boldsymbol{x} : \|\boldsymbol{x}\|_2 \leq 1\}$ whereas $h(\cdot)$ is typically non-smooth but convex and absolutely homogeneous, *e.g.,* norm regularizers and their combinations. Our solution is based on the Riemannian proximal gradient, using an idea we call *proxy step-size* – a scalar variable which we prove is monotone with respect to the actual step-size within an interval. The proxy step-size exists ubiquitously for convex and absolutely homogeneous $h(\cdot)$, and decides the actual step-size and the tangent update in closed-form, thus the complete proximal gradient iteration. Based on these insights, we design a Riemannian proximal gradient method using the proxy step-size. We prove that our method converges to a critical point, guided by a line-search technique based on the $g(\cdot)$ cost only. The proposed method can be implemented in a couple of lines of code. We show its usefulness by applying nuclear norm, $\ell_1$ norm, and nuclear-spectral norm regularization to three classical computer vision problems. The improvements are consistent and backed by numerical experiments.

**Index Terms**—Proxy step-size, Riemannian proximal gradient, non-smooth optimization, regularization, computer vision

## 1 Introduction

We start with optimization problems on the unit sphere, *i.e.,* the sphere manifold:

$$\underset{\boldsymbol{x} \in \mathbb{R}^n}{\arg\min}\ g(\boldsymbol{x}) \quad \text{s.t.} \quad \|\boldsymbol{x}\|_2 = 1. \tag{1}$$

The sphere constraint $\|\boldsymbol{x}\|_2 = 1$ has been widely used to model scale invariant mathematical structures, *e.g.,* the fundamental matrix [1] and the dual absolute quadric [2] in geometric vision (see [3] for more such applications). In statistics, the likelihood is defined up to scale [4], thus often results in problems in the form (1), *e.g.,* the spectral correspondence association [5] and the wavelet density estimation [6]. Some other well-known applications of problem (1) include the $p$-harmonic energy minimization [7] and discretized Bose-Einstein condensates [8].

Other than being constrained on the sphere, some applications require $\boldsymbol{x}$ to possess additional structures, *e.g.,* low-rank (by reorganizing the elements of $\boldsymbol{x}$ in matrix form) or sparsity (*i.e.,* number of nonzeros), to favor its physical/geometric meaning. These additional properties can be often enforced by a dedicated regularization term $h(\boldsymbol{x})$, leading to the following regularized optimization:

$$\underset{\boldsymbol{x} \in \mathbb{R}^n}{\arg\min}\ g(\boldsymbol{x}) + h(\boldsymbol{x}) \quad \text{s.t.} \quad \|\boldsymbol{x}\|_2 = 1. \tag{2}$$

In general, the regularizer $h(\cdot)$ is non-smooth but convex and absolutely homogeneous. Typically $h(\cdot)$ are norm functions or their combinations, in particular, $\ell_1$ norm for sparsity and nuclear norm for low-rank [9].

Problem (2) is difficult, because of the entangling of the non-smooth cost $h(\cdot)$ and the non-convex manifold constraint $\|\boldsymbol{x}\|_2 = 1$. This inhibits the direct applicability of well-studied classical methods, *i.e.,* the Euclidean optimization techniques for non-smooth composite costs [10]–[14], and the Riemannian optimization techniques for smooth costs [8], [15], [16]. In the literature, researchers have explored several ideas to solve non-smooth optimization problems with non-convex constraints, *e.g.,* Riemannian subgradient methods [17]–[20], proximal point methods [21]–[23], operator-splitting methods [24]–[28], and more recently Riemannian proximal gradient methods [29], [30] (see Section 2 for a short review of these methods). Among them, the Riemannian proximal gradient methods show significant advantages over other methods, in terms of both convergence guarantees and convergence speed [29].

The Riemannian proximal gradient is quite a recent research topic, mainly due to Chen *et al.*'s [29] and Huang *et al.*'s [30] work on the Stiefel manifold. These methods [29], [30] are exact methods with convergence proofs to a critical point, while the others either lack such proofs or only have proofs for special forms of $h(\cdot)$ [31]. At its core, [29], [30] solve a non-smooth equation derived from the Karush–Kuhn–Tucker (KKT) system by an iterative semi-smooth Newton method (SSNM) [32]. This approach requires the generalized Clarke differential, which is difficult to obtain and only applicable to special forms, *e.g.,* the $\ell_1$ norm regularization used in [29], [30], [32]. Besides, the use of the generalized Clarke differential contradicts the spirit of proximal gradient, as the proximal is intended to avoid the computation of sub-gradients. It should be noted that the unit sphere is a special case of the Stiefel manifold, thus the result in [29], [30] applies to problem (2). However, due to the usage of the generalized Clarke differential, it is hard

• *Both the authors are with the ENCOV, IGT, Institut Pascal, Université Clermont Auvergne, CHU Clermont-Ferrand, France.*
*E-mail: fang.bai@yahoo.com; adrien.bartoli@gmail.com*
*Corresponding author: Fang Bai*
*The work was supported by the ANR project TOPACS and the CLARA project AIALO.*

*Code available at: https://bitbucket.org/FangBai/proxystepsize-pgs*

to implement $h(\cdot)$ even as the nuclear norm regularization, not to mention more advanced $h(\cdot)$.

In this work, we advance the proximal gradient method on the sphere manifold (PGS) by discovering a concept we call *proxy step-size*, for convex and absolute homogeneous $h(\cdot)$. Importantly, we prove that the proxy step-size is monotone with respect to the actual step-size in the working region and define one line-search iteration in closed-form. Thus the generalized Clarke differential is never required. Based on this novel insight, we control the optimization flow using the proxy step-size, and establish the convergence proof to a critical point (*i.e.,* a solution that satisfies the first-order optimality condition). Our final outputs are three PGS algorithms (of which two are accelerated algorithms with the Nesterov momentum technique) that retain the elegance of classical Euclidean proximal gradient methods. Our method is easy to implement and much faster than the SSNM based methods [29], [30]. Our main contributions are highlighted as follows:

- **Section 4.1.** We reveal the existence of the proxy step-size by exploiting the convexity and absolute homogeneity of the non-smooth cost $h(\cdot)$, and show how it decides both the actual step-size and the tangent update in closed-form.
- **Section 4.2.** We establish the monotonicity between the proxy step-size and the actual step-size, allowing one to control the actual step-size using the proxy step-size monotonically.
- **Section 4.3.** We establish the convergence proof to a critical point, using a line-search from the $g(\cdot)$ cost only, by mildly assuming $g(\cdot)$ has Lipschitz continuous gradient within the unit ball $\{\boldsymbol{x} : \|\boldsymbol{x}\| \leq 1\}$.
- **Section 5.** We present accelerated versions of our PGS algorithm using the Nesterov momentum technique [14], [33]. We empirically show that the accelerated algorithms converge much faster.
- **Section 6.** We demonstrate our algorithms with three applications, by applying nuclear norm, $\ell_1$ norm and nuclear-spectral norm regularization to three well-know computer vision problems.

We start with related work in Section 2, and necessary background in Section 3. Then we formally present our proxy step-size technique in Section 4, and the accelerated version in Section 5. The three example applications and experimental results are given in Section 6 and Section 7, respectively. Section 8 concludes the paper.

## 2 Related Work

Both optimizing a smooth cost on the manifold [8], [15], [16] and proximal gradient for non-smooth optimization in the Euclidean space [10], [11], [11], [12], [14] have been well studied in the literature. However, there exist only a few methods for solving non-smooth cost functions on the manifold. We review existing techniques in this regard.

### 2.1 Riemannian Subgradient Methods

Subgradient methods require one to evaluate the descent direction of the total cost directly. The descent direction, in the non-smooth setting, is characterized by the notion of generalized Clarke gradient, which is usually difficult to calculate even numerically in practice, *e.g.,* see [34], [35] for certain types of functions. Instead, researchers seek for approximations of the subgradient. A key concept in this regard is the $\epsilon$-subgradient [36]. Grohs *et al.* proposed two Riemannian $\epsilon$-subgradient methods based on the line-search [17] and trust-region techniques [18], with convergence guarantee to a critical point. Hosseini *et al.* [20] generalized the idea to the $\epsilon$-subgradient-oriented descent sequence which combines the idea of the BFGS algorithm. Hosseini *et al.* [19] gave a non-smooth Riemannian gradient sampling method with convergence analysis. Despite the hassle to handle the subgradient, the subgradient method is shown to have a slow convergence rate $O(1/\sqrt{k})$ [23], [37].

### 2.2 Proximal Point Methods

Ferreira *et al.* [21] proposed the proximal point algorithm on the Riemannian manifold, and Bento *et al.* [23] established the $O(1/k)$ convergence rate of the algorithm on the Hadamard manifold for convex cost functions. However since every smooth function that is geodesically convex on a compact Riemannian manifold is a constant [38], the analysis in [21], [23] does not apply to compact Riemannian manifolds (*e.g.,* the Stiefel manifold and the unit sphere). Bento *et al.* [22] gave a convergence analysis on the general Riemannian setting by assuming the cost function satisfies the Kurdyka–Lojasiewicz inequality. In terms of computation, the Riemannian proximal point algorithm [21] requires one to solve a subproblem to which an efficient solution does not exist within the current research. As a result, this line of research is largely restricted to theoretical interests at the moment.

### 2.3 Operator-splitting Methods

The hardness of problem (2) is caused by the composition of a non-smooth cost function and the non-convex manifold constraint. In the convex setting, the cost function and the constraint can be handled separately by the alternating direction methods of multipliers (ADMM) [39]. We recapitulate the major advancements of this technique in the non-smooth and non-convex setting. Lai *et al.* [24] explored the splitting of orthogonality constraints (SOC) method, which handles the orthogonality constraint and the cost function separately. Kovnatsky *et al.* [25] proposed the manifold ADMM (MADMM) method, which further exploits the composite structure of the smooth and non-smooth cost functions. However, both these methods, SOC and MADMM, lack convergence proofs in their original paper. A deeper exploitation in terms of the convergence study has been conducted by Wang *et al.* [28]. For special forms of $h(\cdot)$, the convergence of ADMM to a stationary point can be established, *e.g.,* see the stabilized ADMM (SADMM) for $\ell_1$ norm regularization on the Stiefel manifold [31]. More recently, Chen *et al.* [26] proposed PAMAL, the proximal alternating minimized augmented Lagrangian method. The PAMAL method enjoys the sub-sequence convergence property, and is noticeably faster than the SOC method by the experiments in [26]. Another variant, termed EPALAML, was proposed by Zhu *et al.* [27], based on the proximal alternating linearized minimization (PALM) method. Both

PAMAL [26] and EPALAML [27] minimize the augmented Lagrangian function approximately with different methods.

### 2.4 Proximal Gradient Methods

The development of proximal gradient on the Riemannian manifold is a rather new topic. It started with the landmark paper from Chen *et al.* [29] which developed a proximal gradient on the Stiefel manifold with convergence guarantee to a critical point. Experiments in [29] show that the Riemannian proximal gradient method is more efficient than operator-splitting methods such as SOC and PAMAL. More recently, Huang *et al.* [30] proposed another formulation, by using retractions on the non-smooth cost as well to define an iteration. In [30], a convergence rate analysis was also given based on this adaptation, which is $O(1/k)$ for the case without acceleration. In addition to convergence guarantee to a critical point, Riemannian proximal gradient methods allow the possibility to design accelerated algorithms to obtain even faster convergence, see [30], [40].

However, the subproblem of each iteration in [29], [30] is solved by a SSNM which is expensive. In addition, the SSNM method requires the generalized Clarke differential which is difficult for advanced $h(\cdot)$ [41] and contradicts the spirit of proximal gradient (which aims to avoid the generalized Clarke differential).

In this work, we propose the proxy step-size technique for proximal gradient on the sphere manifold by exploiting the convexity and absolute homogeneity of $h(\cdot)$. Our method does not require the generalized Clarke differential, thus is more applicable to advanced $h(\cdot)$. In addition, our method is much faster thanks to the closed-form evaluation.

## 3 Preliminaries

### 3.1 Absolute Homogeneous Function

A function $h(\cdot)$ is said to be *absolutely homogeneous* if $h(\alpha \boldsymbol{x}) = |\alpha| h(\boldsymbol{x})$ for any scalar $\alpha$ and vector $\boldsymbol{x}$.

**Lemma 1.** *If $h(\cdot)$ is convex and absolutely homogeneous, then:*

- $h(\mathbf{0}) = 0$;
- $h(\cdot)$ *is even, i.e.,* $h(-\boldsymbol{x}) = h(\boldsymbol{x})$ *for any* $\boldsymbol{x}$.
- $h(\cdot)$ *is non-negative, i.e.,* $h(\boldsymbol{x}) \geq 0$ *for any* $\boldsymbol{x}$.

*Proof.* The first two are obtained by setting $\alpha = 0$ and $\alpha = -1$ respectively in $h(\alpha \boldsymbol{x}) = |\alpha| h(\boldsymbol{x})$. The third is true because if $h(\cdot)$ is further convex:

$$\frac{h(\boldsymbol{x}) + h(-\boldsymbol{x})}{2} \geq h\left(\frac{\boldsymbol{x} + (-\boldsymbol{x})}{2}\right) = h(\mathbf{0}).$$

The proof is immediate by $h(-\boldsymbol{x}) = h(\boldsymbol{x})$ and $h(\mathbf{0}) = 0$. □

### 3.2 Proximal Operator

For a convex but possibly non-smooth function $h(\cdot)$, the proximal operator evaluates the solution of the following convex optimization problem at a given point $\boldsymbol{w}$:

$$\begin{aligned} \operatorname{prox}_{th}(\boldsymbol{w}) &= \arg\min_{\boldsymbol{x}} \left\{ th(\boldsymbol{x}) + \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{w}\|_2^2 \right\} \\ &= \arg\min_{\boldsymbol{x}} \left\{ h(\boldsymbol{x}) + \frac{1}{2t}\|\boldsymbol{x} - \boldsymbol{w}\|_2^2 \right\} \stackrel{\text{def}}{=} \boldsymbol{z}, \end{aligned} \tag{3}$$

where $t \geq 0$ is a given scalar. For $t = 0$, $\boldsymbol{z} = \boldsymbol{w}$.

Since the proximal is a convex problem, its solution is characterized by its first-order necessary condition:

$$\boldsymbol{z} = \operatorname{prox}_{th}(\boldsymbol{w}) \Leftrightarrow \mathbf{0} \in \partial h|_{\boldsymbol{z}} + \frac{1}{t}(\boldsymbol{z} - \boldsymbol{w}) \tag{4}$$

$$\Leftrightarrow \boldsymbol{w} \in t\partial h|_{\boldsymbol{z}} + \boldsymbol{z}. \tag{5}$$

The proximal satisfies firm non-expansiveness and non-expansiveness (see Appendix A). Additionally, we show the following statements hold true.

**Lemma 2.** *If $h(\cdot)$ is convex and absolutely homogeneous and $t \geq 0$, we have:*

- $\operatorname{prox}_{th}(\mathbf{0}) = \mathbf{0}$;
- $\|\operatorname{prox}_{th}(\boldsymbol{w})\|_2^2 \leq \langle \operatorname{prox}_{th}(\boldsymbol{w}),\, \boldsymbol{w} \rangle$;
- $\|\operatorname{prox}_{th}(\boldsymbol{w})\|_2 \leq \|\boldsymbol{w}\|_2$

*Proof.* See Appendix A. □

### 3.3 Proximal Gradient in the Euclidean Space

Let $g(\cdot)$ be smooth, and $h(\cdot)$ be convex. A proximal gradient step in the Euclidean space is defined as:

$$\boldsymbol{v}_k = \arg\min_{\boldsymbol{v} \in \mathbb{R}^n} \langle \nabla g|_{\boldsymbol{x}_k}, \boldsymbol{v} \rangle + \frac{1}{2t}\langle \boldsymbol{v}, \boldsymbol{v} \rangle + h(\boldsymbol{x}_k + \boldsymbol{v}) \tag{6}$$

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{v}_k, \tag{7}$$

where $t \geq 0$. Problem (6) is convex, whose solution is characterized by its first-order necessary condition:

$$\begin{aligned} &\mathbf{0} \in \nabla g|_{\boldsymbol{x}_k} + \frac{1}{t}\boldsymbol{v}_k + \partial h|_{\boldsymbol{x}_k + \boldsymbol{v}_k} \Leftrightarrow \\ &\boldsymbol{x}_k - t\nabla g|_{\boldsymbol{x}_k} \in (\boldsymbol{x}_k + \boldsymbol{v}_k) + t\partial h|_{\boldsymbol{x}_k + \boldsymbol{v}_k}. \end{aligned}$$

Therefore, from equation (5), we have:

$$\boldsymbol{x}_k + \boldsymbol{v}_k = \operatorname{prox}_{th}(\boldsymbol{x}_k - t\nabla g|_{\boldsymbol{x}_k}) = \boldsymbol{x}_{k+1}.$$

The update step differs from the classical gradient step for minimizing the $g(\cdot)$ cost only by a proximal operation, thus is termed as *proximal gradient*. Here $t$ works as the *step-size* in the standard gradient descent methods.

### 3.4 Sphere Manifold

The unit sphere, or the sphere manifold, is an embedded manifold:

$$\mathcal{S} = \{\boldsymbol{x} \in \mathbb{R}^n \,:\, \|\boldsymbol{x}\|_2 = 1\}. \tag{8}$$

The tangent space at a point $\boldsymbol{x} \in \mathcal{S}$ is:

$$\mathcal{T}_{\boldsymbol{x}}\mathcal{S} = \left\{ \boldsymbol{v} \in \mathbb{R}^n \,:\, \boldsymbol{x}^{\mathsf{T}}\boldsymbol{v} = 0 \right\}. \tag{9}$$

Let $g : \mathcal{S} \rightarrowtail \mathbb{R}$ be a function defined on the manifold. The Riemannian gradient of $g(\cdot)$ at $\boldsymbol{x} \in \mathcal{S}$, denoted by $\operatorname{grad} g|_{\boldsymbol{x}}$, is the unique tangent vector satisfying:

$$\mathrm{D}g(\boldsymbol{x})[\boldsymbol{v}] = \langle \operatorname{grad} g|_{\boldsymbol{x}},\, \boldsymbol{v} \rangle_{\boldsymbol{x}}, \quad \forall \boldsymbol{v} \in \mathcal{T}_{\boldsymbol{x}}\mathcal{S},$$

where $\mathrm{D}g(\boldsymbol{x})[\boldsymbol{v}]$ is the directional derivative of $g(\cdot)$ along the direction of $\boldsymbol{v}$. We shall use the induced Euclidean metric as the Riemannian metric, which means $\langle \boldsymbol{v}, \boldsymbol{v} \rangle_{\boldsymbol{x}} = \langle \boldsymbol{v}, \boldsymbol{v} \rangle = \boldsymbol{v}^{\mathsf{T}}\boldsymbol{v}$. Practically, the Riemannian gradient $\operatorname{grad} g|_{\boldsymbol{x}}$ can be obtained by orthogonally projecting the Euclidean gradient

$\nabla g|_{\boldsymbol{x}}$ in $\mathbb{R}^n$ into the tangent space at $\boldsymbol{x}$. Using the so-called orthogonal projector $\mathrm{proj}_{\mathcal{T}_{\boldsymbol{x}}\mathcal{S}}(\cdot)$, we can write:

$$\operatorname{grad} g|_{\boldsymbol{x}} = \mathrm{proj}_{\mathcal{T}_{\boldsymbol{x}}\mathcal{S}} \nabla g|_{\boldsymbol{x}} = \left(\boldsymbol{I} - \boldsymbol{x}\boldsymbol{x}^{\mathsf{T}}\right) \nabla g|_{\boldsymbol{x}} \tag{10}$$

$$= \nabla g|_{\boldsymbol{x}} - \langle \boldsymbol{x}, \nabla g|_{\boldsymbol{x}} \rangle \, \boldsymbol{x}. \tag{11}$$

We shall use the following *retraction* to bring an increment in the tangent space back to the manifold:

$$\mathcal{R}_{\boldsymbol{x}}(\boldsymbol{v}) \overset{\text{def}}{=} \frac{\boldsymbol{x}+\boldsymbol{v}}{\|\boldsymbol{x}+\boldsymbol{v}\|_2} \; : \; \mathcal{T}_{\boldsymbol{x}}\mathcal{S} \rightarrowtail \mathcal{S}. \tag{12}$$

For more details of these concepts, we refer to [15], [16].

### 3.5 Proximal Gradient on the Sphere Manifold

**Formulation in the tangent space.** Inspired by the proximal gradient in the Euclidean space, researchers have tried to formulate the update vector $\boldsymbol{v}_k$ in the tangent space of $\boldsymbol{x}_k$ as an extension to the manifold setting [29], [30]. In this work, we propose the following:

$$\boldsymbol{v}_k = \underset{\boldsymbol{v} \in \mathcal{T}_{\boldsymbol{x}_k}\mathcal{S}}{\arg\min} \, \langle \operatorname{grad} g|_{\boldsymbol{x}_k}, \boldsymbol{v} \rangle + \frac{1}{2t}\langle \boldsymbol{v}, \boldsymbol{v} \rangle + h(\boldsymbol{x}_k + \boldsymbol{v}) \tag{13}$$

$$\boldsymbol{x}_{k+1} = \mathcal{R}_{\boldsymbol{x}_k}(\boldsymbol{v}_k). \tag{14}$$

The subproblem (13) was first proposed by Chen *et al.* [29] in their work on the Stiefel manifold. However in Chen *et al.*'s work, the update equation (14) is $\boldsymbol{x}_{k+1} = \mathcal{R}_{\boldsymbol{x}_k}(\alpha_k \boldsymbol{v}_k)$ with $\alpha_k$ acting as another step-size to control the length of $\boldsymbol{v}_k$ similar to Riemannian subgradient methods. We shall see that this $\alpha_k$ is not required, as in accordance with equation (7) in the Euclidean proximal gradient.

**The solution from the KKT system.** Problem (13) is convex, thus its solution is uniquely characterized by the KKT system. We take the tangent space constraint explicitly, and write the Lagrange function as:

$$\mathcal{L}(\boldsymbol{v}, \mu) = \langle \operatorname{grad} g|_{\boldsymbol{x}_k}, \boldsymbol{v} \rangle + \frac{1}{2t}\langle \boldsymbol{v}, \boldsymbol{v} \rangle + h(\boldsymbol{x}_k + \boldsymbol{v}) + \mu \boldsymbol{x}_k^{\mathsf{T}} \boldsymbol{v}.$$

The KKT system is $\boldsymbol{0} \in \partial \mathcal{L}_{\boldsymbol{v}}$, $\boldsymbol{x}_k^{\mathsf{T}}\boldsymbol{v}_k = 0$. With some trivial calculations, the KKT system is reduced to the following (see Appendix B for details):

$$\begin{cases} \boldsymbol{v}_k = \mathrm{prox}_{th}\left((1-\mu t)\,\boldsymbol{x}_k - t \operatorname{grad} g|_{\boldsymbol{x}_k}\right) - \boldsymbol{x}_k & \text{(15a)} \\ \boldsymbol{x}_k^{\mathsf{T}} \mathrm{prox}_{th}\left((1-\mu t)\,\boldsymbol{x}_k - t \operatorname{grad} g|_{\boldsymbol{x}_k}\right) = 1. & \text{(15b)} \end{cases}$$

This KKT system can be solved by the SSNM method [32], by solving the non-smooth equation (15b) first to obtain the Lagrange multiplier $\mu$ and then computing the tangent update $\boldsymbol{v}_k$ by equation (15a). This approach has been discussed in [32] and used by Chen *et al.* in [29].

**Limitations of the existing solution.** While subproblem (13) seems a trivial extension from the Euclidean case (6), the existing solution is not as elegant as its Euclidean counterpart. First, to apply the SSNM method, as $h(\cdot)$ is non-smooth, the generalized Clarke differential of $h(\cdot)$ is required. This contradicts the spirit of proximal gradient, as the proximal operator is typically used to avoid the generalized differential. Second, the generalized Clarke differential is usually difficult to obtain and only applicable for special forms, *e.g.*, the $\ell_1$ norm used in [29], [30]. If $h(\cdot)$ is the nuclear norm or more advanced functions, the SSNM is hard to implement. Third, solving an inner loop by an iterative method (like SSNM) can degenerate the numerical accuracy and even the overall convergence.

In this work, we propose proxy step-size, an effective technique to handle subproblem (13). Instead of solving (13) directly, we aim to generate valid solutions to problem (13) in closed-form controlled by the proxy step-size. With this new technique, the generalized differential is never required, and we show that proximal gradient on the sphere can be elegantly formulated as a proximal gradient step with respect to the proxy step-size followed by normalizations.

**One iteration in Chen *et al.* [29].** After solving $\boldsymbol{v}_k$, a line-search process is used to ensure the descent of the total cost $f(\cdot) = g(\cdot) + h(\cdot)$. In general, they propose:

- given $t$, solve $\boldsymbol{v}_k$ using the SSNM method;
- set $\alpha_k \leftarrow 1$ and shrink $\alpha_k$ until the following line-search criterion is met:

$$f\left(\mathcal{R}_{\boldsymbol{x}_k}(\alpha \boldsymbol{v}_k)\right) \leq f(\boldsymbol{x}_k) - \frac{\alpha_k}{2t}\langle \boldsymbol{v}_k, \boldsymbol{v}_k \rangle; \tag{16}$$

- set $\boldsymbol{x}_{k+1} \leftarrow \mathcal{R}_{\boldsymbol{x}_k}(\alpha_k \boldsymbol{v}_k)$.

The validity of the line-search (16) is proved in [29].

**Convex toolbox for problem (13).** Problem (13) is convex, thus a naive idea is to simply call a convex toolbox. However, the computation is prohibitive for practical usage for high dimensional $\boldsymbol{x}$. Needless to say this is only one iteration, which we want to solve as efficiently as possible. Mathematically, this solution is not elegant.

## 4 The Proxy Step-size Technique

We now formally present our proxy step-size technique to solve one proximal gradient iteration, by assuming $h(\cdot)$ to be absolutely homogeneous. We term the proposed algorithm based on the proxy step-size technique as PGS (short for Proximal Gradient on the Sphere manifold).

### 4.1 Proxy Step-size

**Lemma 3.** *If $h(\cdot)$ is convex and absolutely homogeneous and $t \geq 0$, then $\mathrm{prox}_{th}(\alpha \boldsymbol{w}) = \alpha \mathrm{prox}_{\frac{t}{|\alpha|}h}(\boldsymbol{w})$ for any scalar $\alpha$.*

*Proof.* See Appendix C. □

Now we introduce $t' = \frac{t}{1-\mu t}$, which we term *proxy step-size*. We show that the update equations from $\boldsymbol{x}_k$ to $\boldsymbol{x}_{k+1}$ are completely determined by the proxy step-size $t'$. To that end, we rewrite equation (15a) as:

$$\begin{aligned} \boldsymbol{x}_k + \boldsymbol{v}_k &= \mathrm{prox}_{th}\left((1-\mu t)\,\boldsymbol{x}_k - t \operatorname{grad} g|_{\boldsymbol{x}_k}\right) \\ &= \mathrm{prox}_{th}\left((1-\mu t)\left(\boldsymbol{x}_k - \frac{t}{1-\mu t}\operatorname{grad} g|_{\boldsymbol{x}_k}\right)\right) \\ &= (1-\mu t)\,\mathrm{prox}_{\frac{t}{|1-\mu t|}h}\left(\boldsymbol{x}_k - \frac{t}{1-\mu t}\operatorname{grad} g|_{\boldsymbol{x}_k}\right) \\ &= \frac{t}{t'}\mathrm{prox}_{|t'|h}\left(\boldsymbol{x}_k - t' \operatorname{grad} g|_{\boldsymbol{x}_k}\right), \end{aligned}$$

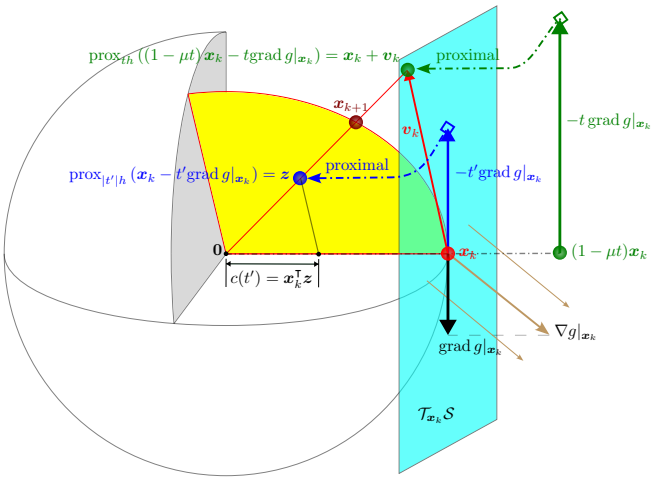



Fig. 1: The proxy step-size technique. In the KKT system (15), given $t$, the essence of the non-smooth equation (15b) is to decide a proper Lagrange multiplier $\mu$ that lands the point $\mathrm{prox}_{th}\left((1-\mu t)\,\boldsymbol{x}_k - t\mathrm{grad}\,g|_{\boldsymbol{x}_k}\right)$ into the tangent plane at $\boldsymbol{x}_k$. Such a process is difficult as it is hard to solve equation (15b) for advanced $h(\cdot)$. Instead, we propose to use the proxy step-size $t'$ to generate valid solutions to the KKT system (15). In specific, we first move in the tangent plane at $\boldsymbol{x}_k$ by $-t'\mathrm{grad}\,g|_{\boldsymbol{x}_k}$, and then apply the proximal to reach the point $\boldsymbol{z}=\mathrm{prox}_{|t'|h}\left(\boldsymbol{x}_k - t'\mathrm{grad}\,g|_{\boldsymbol{x}_k}\right)$. We see the point $\mathrm{prox}_{th}\left((1-\mu t)\,\boldsymbol{x}_k - t\mathrm{grad}\,g|_{\boldsymbol{x}_k}\right)$ is the intersection of the line $(\mathbf{0},\boldsymbol{z})$ and the tangent plane at $\boldsymbol{x}_k$, given as $\frac{1}{\boldsymbol{x}_k^\mathsf{T}\boldsymbol{z}}\boldsymbol{z}$.

where the third equality stems from Lemma 3. By $\boldsymbol{x}_k^\mathsf{T}\boldsymbol{v}_k=0$, we obtain $t=\frac{t'}{\boldsymbol{x}_k^\mathsf{T}\mathrm{prox}_{|t'|h}\left(\boldsymbol{x}_k - t'\mathrm{grad}\,g|_{\boldsymbol{x}_k}\right)}$. Therefore the KKT system (15) can be rewritten as:

$$\begin{cases} \boldsymbol{v}_k = \dfrac{\mathrm{prox}_{|t'|h}\left(\boldsymbol{x}_k - t'\mathrm{grad}\,g|_{\boldsymbol{x}_k}\right)}{\boldsymbol{x}_k^\mathsf{T}\mathrm{prox}_{|t'|h}\left(\boldsymbol{x}_k - t'\mathrm{grad}\,g|_{\boldsymbol{x}_k}\right)} - \boldsymbol{x}_k & \text{(17a)} \\ t = \dfrac{t'}{\boldsymbol{x}_k^\mathsf{T}\mathrm{prox}_{|t'|h}\left(\boldsymbol{x}_k - t'\mathrm{grad}\,g|_{\boldsymbol{x}_k}\right)} \overset{\text{def}}{=} \phi(t'). & \text{(17b)} \end{cases}$$

The new KKT system (17) can be considered as a reparameterization of the previous KKT system (15), using $t$ and $t'$. However, in the new KKT system, both $\boldsymbol{v}_k$ and $t$ are completely decided by the proxy step-size $t'$.

Therefore, problem (13) can be solved in closed-form with respect to a given proxy step-size $t'$ as follow:

$$\begin{cases} \boldsymbol{z} = \mathrm{prox}_{|t'|h}\left(\boldsymbol{x}_k - t'\mathrm{grad}\,g|_{\boldsymbol{x}_k}\right) \\ t = \frac{1}{\boldsymbol{x}_k^\mathsf{T}\boldsymbol{z}}t' \\ \boldsymbol{v}_k = \frac{1}{\boldsymbol{x}_k^\mathsf{T}\boldsymbol{z}}\boldsymbol{z} - \boldsymbol{x}_k. \end{cases} \tag{18}$$

Note that $t$ and $\boldsymbol{v}_k$ computed from $t'$ satisfy the KKT system (15), thus they are optimal for problem (13). An illustration of the proxy step-size technique is given in Fig. 1.

To design iterations based on the proxy step-size entirely, we need to reveal the relation between the proxy step-size $t'$ and the actual step-size $t$, and design a line-search process to govern the convergence.

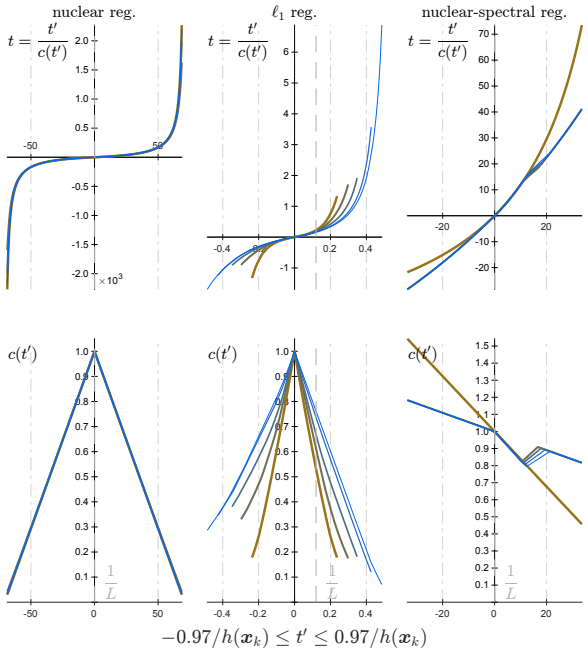



Fig. 2: Numerical examples for Proposition 2 and Theorem 1. If $h(\cdot)$ is convex and absolutely homogeneous, then for $|t'| < 1/h(\boldsymbol{x}_k)$, we have $c(t')=\boldsymbol{x}_k^\mathsf{T}\boldsymbol{z}>0$ and thus the mapping $t=\phi(t')=t'/c(t')$ is monotonically increasing within this interval. The reference proxy step-size $1/L$ to be described in Section 4.4.1 is plotted as vertical dashed lines. These examples are obtained from the first 5 iterations of: 1) fundamental matrix estimation with nuclear norm regularization, 2) correspondence association with $\ell_1$ norm regularization, and 3) self-calibration with nuclear-spectral norm regularization to be presented in Section 6.

### 4.2 The Mapping between Proxy Step-size and Actual Step-size

We denote $t=\phi(t')=t'/c(t')$ the mapping between $t'$ and $t$ defined by equation (17b), where we introduce $c(t')=\boldsymbol{x}_k^\mathsf{T}\boldsymbol{z}=\boldsymbol{x}_k^\mathsf{T}\mathrm{prox}_{|t'|h}\left(\boldsymbol{x}_k - t'\mathrm{grad}\,g|_{\boldsymbol{x}_k}\right)$.

**Lemma 4.** *Given arbitrary $t_1'\neq 0$, $t_2'\neq 0$, $t_1'\neq t_2'$, it can be shown that:*

$$\epsilon(t_1',t_2')=\left(\frac{1}{\phi(t_1')}-\frac{1}{\phi(t_2')}\right)\left(\frac{1}{t_1'}-\frac{1}{t_2'}\right)\geq 0, \tag{19}$$

*which means $1/\phi(t')$ is monotonically increasing with respect to $1/t'$.*

*Proof.* See Appendix D. □

Lemma 4 states the monotonicity between $1/t'$ and $1/t$. To establish the monotonicity between $t'$ and $t$ explicitly, it suffices to identify an interval where $c(t')>0$.

**Proposition 1.** *If $c(t')>0$ for $t'\in[l,u]$, then $\phi(t')$ is monotonically increasing with respect to $t'$ for $t'\in[l,u]$.*

*Proof.* Note that $\phi(t')=t'/c(t')$. We thus have:

$$t_1't_2'\phi(t_1')\phi(t_2')=(t_1't_2')^2/(c(t_1')c(t_2'))>0 \Leftrightarrow c(t_1')c(t_2')>0$$

Therefore if $c(t_1')c(t_2')>0$, inequality (19) is equivalent to:

$$\left(\phi(t_1')-\phi(t_2')\right)\left(t_1'-t_2'\right)\geq 0. \tag{20}$$

We see $c(t') > 0$ is sufficient for $c(t'_1)c(t'_2) > 0$. □

Now we characterize an interval in which $c(t') > 0$. For this purpose, we first prove the following global inequality for convex and absolutely homogeneous $h(\cdot)$:

**Lemma 5.** *Let $h(\cdot)$ be convex and absolutely homogeneous. Then for any $t$, $\boldsymbol{x}$ and $\boldsymbol{w}$, we have:*

$$\langle \boldsymbol{w} - \operatorname{prox}_{|t|h}(\boldsymbol{w}), \boldsymbol{x} \rangle \leq |t|\, h(\boldsymbol{x}). \tag{21}$$

*The inequality is tight for $\boldsymbol{x} = \operatorname{prox}_{|t|h}(\boldsymbol{w})$.*

*Proof.* See Appendix E. □

**Proposition 2.** *If $h(\cdot)$ is convex and absolutely homogeneous, then $c(t') \geq 1 - |t'|\, h(\boldsymbol{x}_k)$.*

*Proof.* In Lemma 5, we set $\boldsymbol{w} = \boldsymbol{x}_k - t' \operatorname{grad} g|_{\boldsymbol{x}_k}$ and $\boldsymbol{x} = \boldsymbol{x}_k$, which yields:

$$\langle \boldsymbol{x}_k - t' \operatorname{grad} g|_{\boldsymbol{x}_k} - \operatorname{prox}_{|t'|h}(\boldsymbol{x}_k - t' \operatorname{grad} g|_{\boldsymbol{x}_k}), \boldsymbol{x}_k \rangle \leq |t'|\, h(\boldsymbol{x}_k). \tag{22}$$

We reorganize inequality (22) to complete the proof:

$$c(t') = \boldsymbol{x}_k^{\mathsf{T}} \operatorname{prox}_{|t'|h}(\boldsymbol{x}_k - t' \operatorname{grad} g|_{\boldsymbol{x}_k}) \geq 1 - |t'|\, h(\boldsymbol{x}_k),$$

where we use $\langle \boldsymbol{x}_k, \boldsymbol{x}_k \rangle = 1$ and $\langle \operatorname{grad} g|_{\boldsymbol{x}_k}, \boldsymbol{x}_k \rangle = 0$. □

**Theorem 1.** *If $h(\cdot)$ is convex and absolutely homogeneous, then the mapping $t = \phi(t')$ from proxy step-size $t'$ to actual step-size $t$ is monotonically increasing for $|t'| < 1/h(\boldsymbol{x}_k)$, and $\phi(0) = 0$.*

*Proof.* By Proposition 2, we see $c(t') > 0$ for $|t'| < 1/h(\boldsymbol{x}_k)$. Then by Proposition 1, $\phi(t')$ is monotonically increasing with respect to $t'$ for $|t'| < 1/h(\boldsymbol{x}_k)$. By definition of the proximal operator in equation (3), we observe:

$$\lim_{t' \to 0} \operatorname{prox}_{|t'|h}(\boldsymbol{x}_k - t' \operatorname{grad} g|_{\boldsymbol{x}_k}) = \boldsymbol{x}_k, \tag{23}$$

from which we conclude $c(0) = 1$, thus $\phi(0) = 0$. □

Numerical examples to these results are given in Fig. 2. In practice, we require $t > 0$, thus we use $0 < t' < 1/h(\boldsymbol{x}_k)$. Theorem 1 suggests that we can control the step-size $t$ by the proxy step-size $t'$ within this interval.

**Proposition 3.** *If $h(\cdot)$ is convex and absolutely homogeneous and if $\|\operatorname{grad} g|_{\boldsymbol{x}_k}\|_2$ is bounded, then for any $t' \neq 0$, we have $t = \phi(t') \neq 0$.*

*Proof.* It can be shown (see Appendix F for details) that:

$$|t| \geq \frac{1}{\sqrt{(1/t')^2 + \|\operatorname{grad} g|_{\boldsymbol{x}_k}\|_2^2}}.$$

Thus if $\|\operatorname{grad} g|_{\boldsymbol{x}_k}\|_2$ is bounded, then $t \neq 0$ for $t' \neq 0$. □

We notice that $\|\operatorname{grad} g|_{\boldsymbol{x}_k}\|_2 \leq \|\boldsymbol{I} - \boldsymbol{x}_k \boldsymbol{x}_k^{\mathsf{T}}\|\, \|\nabla g|_{\boldsymbol{x}_k}\|_2 = \|\nabla g|_{\boldsymbol{x}_k}\|_2$, thus it suffices to have a bounded $\|\nabla g|_{\boldsymbol{x}_k}\|_2$. In particular, if the Euclidean gradient $\nabla g$ is Lipschitz continuous on the sphere, $\|\nabla g|_{\boldsymbol{x}_k}\|_2$ is bounded.

**Algorithm 1:** One line-search iteration.

**function** $(\boldsymbol{v}_k, t, t') \leftarrow$ **lineSearch** $(\boldsymbol{x}_k, t'_{\max})$
- $t' \leftarrow \min\{t'_{\max}, 1/h(\boldsymbol{x}_k)\}$
- $\boldsymbol{z} \leftarrow \operatorname{prox}_{|t'|h}(\boldsymbol{x}_k - t' \operatorname{grad} g|_{\boldsymbol{x}_k})$
- $\boldsymbol{v}_k \leftarrow \frac{1}{\boldsymbol{x}_k^{\mathsf{T}} \boldsymbol{z}} \boldsymbol{z} - \boldsymbol{x}_k$
- $t \leftarrow \frac{1}{\boldsymbol{x}_k^{\mathsf{T}} \boldsymbol{z}} t'$
- $Q_L = g(\boldsymbol{x}_k) + \langle \operatorname{grad} g|_{\boldsymbol{x}_k}, \boldsymbol{v}_k \rangle + \frac{1}{2t} \langle \boldsymbol{v}_k, \boldsymbol{v}_k \rangle$
- **if** $g(\mathcal{R}_{\boldsymbol{x}_k}(\boldsymbol{v}_k)) \leq Q_L$ **then return** $(\boldsymbol{v}_k, t, t')$
- **else** $t' \leftarrow 0.8t'$, goto step 3

**end**

### 4.3 Line-search and Convergence

To establish the convergence proof, we make the following mild assumption on the cost $g(\cdot)$.

**Assumption 1.** *We assume the pullback function $g(\mathcal{R}_{\boldsymbol{x}}(\boldsymbol{v}))$ with $\boldsymbol{v} \in \mathcal{T}_{\boldsymbol{x}}\mathcal{S}$ satisfies:*

$$g(\mathcal{R}_{\boldsymbol{x}}(\boldsymbol{v})) \leq g(\boldsymbol{x}) + \langle \operatorname{grad} g(\boldsymbol{x}), \boldsymbol{v} \rangle + \frac{L}{2} \langle \boldsymbol{v}, \boldsymbol{v} \rangle, \tag{24}$$

*for a (known or unknown) constant $L > 0$.*

Assumption 1 holds if the ambient Euclidean space function $g(\cdot)$ on $\mathbb{R}^n$ has Lipschitz continuous gradient $\nabla g$ within the convex hull of $\mathcal{S}$ (Lemma 3 in [42]), *i.e.,* within the unit ball $\{\boldsymbol{x} : \|\boldsymbol{x}\|_2 \leq 1\}$. In other words, this assumption is satisfied if $g(\cdot)$ is not changing radically within the ball (like going to infinity at some point). Thus in practice, this assumption is hardly violated.

Based on Assumption 1, we propose the following line-search.

**Line-search criterion.** Search for a proxy step-size $t'$, such that the actual step-size $t$ and the tangent update $\boldsymbol{v}_k$ solved from equation (18) satisfy:

$$g(\mathcal{R}_{\boldsymbol{x}_k}(\boldsymbol{v}_k)) \leq g(\boldsymbol{x}_k) + \langle \operatorname{grad} g|_{\boldsymbol{x}_k}, \boldsymbol{v}_k \rangle + \frac{1}{2t} \langle \boldsymbol{v}_k, \boldsymbol{v}_k \rangle. \tag{25}$$

By Assumption 1, the inequality (25) is satisfied for any $0 \leq t \leq 1/L$. This line-search criterion is in the same spirit as the one used in the classical Euclidean proximal gradient literature, where the Euclidean gradient $\nabla g|_{\boldsymbol{x}_k}$ is used instead of the Riemannian gradient here.

**Line-search process**. We start with an initial proxy step-size $t' < 1/h(\boldsymbol{x}_k)$ and reduce $t'$ until $t = \phi(t')$ satisfies $0 \leq t \leq 1/L$. This process is well-defined by the monotonicity of $t = \phi(t')$, as proved by Theorem 1. One line-search iteration is given in Algorithm 1.

We now show the line-search criterion guarantees a descent for the total cost $f(\boldsymbol{x}) = g(\boldsymbol{x}) + h(\boldsymbol{x})$ at each iteration. See the PGS curve in Fig. 4 for an illustration.

**Lemma 6.** *For retraction (12), if $h(\cdot)$ is absolutely homogeneous and $\boldsymbol{v}_k \in \mathcal{T}_{\boldsymbol{x}_k}\mathcal{S}$, we have $h(\mathcal{R}_{\boldsymbol{x}_k}(\boldsymbol{v}_k)) \leq h(\boldsymbol{x}_k + \boldsymbol{v}_k)$.*

*Proof.* See Appendix G. □

**Theorem 2.** *Let $f(\boldsymbol{x}) = g(\boldsymbol{x}) + h(\boldsymbol{x})$. If the line-search criterion (25) holds then:*

$$f(\boldsymbol{x}_{k+1}) = f(\mathcal{R}_{\boldsymbol{x}_k}(\boldsymbol{v}_k)) \leq f(\boldsymbol{x}_k) - \frac{1}{2t} \langle \boldsymbol{v}_k, \boldsymbol{v}_k \rangle. \tag{26}$$

*Proof.* The first-order necessary optimality condition of problem (13) states that:

$$\mathbf{0} \in \operatorname{grad} g|_{\boldsymbol{x}_k} + \frac{1}{t}\boldsymbol{v}_k + \operatorname{proj}_{\mathcal{T}_{\boldsymbol{x}_k}\mathcal{S}} \partial h|_{\boldsymbol{x}_k+\boldsymbol{v}_k} \tag{27}$$

$$\Leftrightarrow -\operatorname{grad} g|_{\boldsymbol{x}_k} - \frac{1}{t}\boldsymbol{v}_k \in \operatorname{proj}_{\mathcal{T}_{\boldsymbol{x}_k}\mathcal{S}} \partial h|_{\boldsymbol{x}_k+\boldsymbol{v}_k}. \tag{28}$$

Because $\boldsymbol{v}_k \in \mathcal{T}_{\boldsymbol{x}_k}\mathcal{S}$ then $\langle \boldsymbol{z}, \boldsymbol{v}_k\rangle = \langle \operatorname{proj}_{\mathcal{T}_{\boldsymbol{x}_k}\mathcal{S}}\boldsymbol{z}, \boldsymbol{v}_k\rangle$ for any $\boldsymbol{z} \in \mathbb{R}^n$. By the convexity of $h(\cdot)$ at $\boldsymbol{x}_k + \boldsymbol{v}_k$, we obtain:

$$h(\boldsymbol{x}_k + \boldsymbol{v}_k) \le h(\boldsymbol{x}_k) + \langle \partial h|_{\boldsymbol{x}_k+\boldsymbol{v}_k}, \boldsymbol{v}_k\rangle \tag{29}$$

$$= h(\boldsymbol{x}_k) + \langle \operatorname{proj}_{\mathcal{T}_{\boldsymbol{x}_k}\mathcal{S}} \partial h|_{\boldsymbol{x}_k+\boldsymbol{v}_k}, \boldsymbol{v}_k\rangle \tag{30}$$

$$\overset{(28)}{=} h(\boldsymbol{x}_k) + \langle -\operatorname{grad} g|_{\boldsymbol{x}_k} - \frac{1}{t}\boldsymbol{v}_k, \boldsymbol{v}_k\rangle. \tag{31}$$

By considering Lemma 6, we have the following inequality:

$$h(\mathcal{R}_{\boldsymbol{x}_k}(\boldsymbol{v}_k)) \le h(\boldsymbol{x}_k) + \langle -\operatorname{grad} g|_{\boldsymbol{x}_k} - \frac{1}{t}\boldsymbol{v}_k, \boldsymbol{v}_k\rangle. \tag{32}$$

Summing together inequalities (25) and (32), we obtain inequality (26). □

We denote $t'_k$ $(k = 0, 1, \ldots, K-1)$ to be the proxy step-size satisfying the line-search criterion (25) at iteration $k$. In the line-search process, we have:

$$0 < t'_k \le \min\{t'_{\max}, c/h(\boldsymbol{x}_k)\}, \quad \text{with } 0 < c < 1.$$

We further denote the corresponding actual step-sizes as $t_k$ where $t_k = \phi(t'_k) = t'_k/c(t'_k)$. From Theorem 1, we know $t_k \ge 0$. By Proposition 3, we know if $t'_k \ne 0$ then $t_k \ne 0$ as $\|\operatorname{grad} g|_{\boldsymbol{x}_k}\|_2$ is bounded by the assumption of Lipschitz continuous gradient $\nabla g$ within the unit ball. Therefore we conclude $t_k > 0$. To summarize, we have:

$$0 < t_{\min} \le t_k \le t_{\max},$$

where we denote $t_{\min} = \min\{t_k\}$, $t_{\max} = \max\{t_k\}$.

At last, we show the iterations guided by the line-search process converge to a critical point of problem (2).

**Proposition 4.** *Assume $f(\boldsymbol{x})$ is bounded from below on $\mathcal{S}$, i.e., the problem is well-posed. If $g(\cdot)$ has Lipschitz continuous gradient $\nabla g$ within the unit ball $\{\boldsymbol{x} : \|\boldsymbol{x}\| \le 1\}$, the line-search iterations converge to a critical point of problem (2).*

*Proof.* Let $f^\star$ be the optimal value, *i.e.,* $f^\star \le f(\boldsymbol{x})$ for any $\boldsymbol{x}$ on $\mathcal{S}$. Given $K$ iterations, inequality (26) implies:

$$f(\boldsymbol{x}_0) - f^\star \ge f(\boldsymbol{x}_0) - f(\boldsymbol{x}_K) = \sum_{k=0}^{K-1} (f(\boldsymbol{x}_k) - f(\boldsymbol{x}_{k+1}))$$

$$\ge \sum_{k=0}^{K-1} \frac{1}{2t_k}\langle \boldsymbol{v}_k, \boldsymbol{v}_k\rangle = \sum_{k=0}^{K-1} \frac{t_k}{2}\langle \boldsymbol{v}_k/t_k, \boldsymbol{v}_k/t_k\rangle.$$

Considering $0 < t_{\min} \le t_k \le t_{\max}$, we have:

$$f(\boldsymbol{x}_0) - f^\star \ge \frac{1}{2t_{\max}} \sum_{k=0}^{K-1} \|\boldsymbol{v}_k\|_2^2,$$

$$f(\boldsymbol{x}_0) - f^\star \ge \frac{t_{\min}}{2} \sum_{k=0}^{K-1} \|\boldsymbol{v}_k/t_k\|_2^2.$$

Denote $\epsilon_0 = f(\boldsymbol{x}_0) - f^\star$. Taking $K$ to the infinity, we obtain:

$$\lim_{K\to\infty} \sum_{k=0}^{K-1} \|\boldsymbol{v}_k\|_2^2 \le 2\epsilon_0 t_{\max}, \quad \lim_{K\to\infty} \sum_{k=0}^{K-1} \left\|\frac{\boldsymbol{v}_k}{t_k}\right\|_2^2 \le \frac{2\epsilon_0}{t_{\min}}.$$

Since $f(\boldsymbol{x})$ is bounded from below on $\mathcal{S}$, $\epsilon_0$ is a non-negative constant. Thus the right side of each inequality is bounded. Noting the left side is the summation of an infinite non-negative sequence, we have $\lim_{k\to\infty} \|\boldsymbol{v}_k\|_2 = 0$, $\lim_{k\to\infty} \|\boldsymbol{v}_k/t_k\|_2 = 0$, which means $\lim_{k\to\infty} \boldsymbol{v}_k = \mathbf{0}$, $\lim_{k\to\infty} \boldsymbol{v}_k/t_k = \mathbf{0}$. Lastly, we notice if $\boldsymbol{v}_k = \mathbf{0}$ and $\boldsymbol{v}_k/t_k = \mathbf{0}$, the first-order necessary optimality condition of problem (13), *i.e.,* equation (27), becomes:

$$\mathbf{0} \in \operatorname{grad} g|_{\boldsymbol{x}_k} + \operatorname{proj}_{\mathcal{T}_{\boldsymbol{x}_k}\mathcal{S}} \partial h|_{\boldsymbol{x}_k}, \tag{33}$$

which is exactly the first-order necessary optimality condition of problem (2) [29], [43]. □

Both $\|\boldsymbol{v}_k\|$ and $\|\boldsymbol{v}_k/t_k\|$ have linear convergence rates:

$$\min_{k=1,2,\ldots,K} \|\boldsymbol{v}_k\|_2^2 \le \frac{2\epsilon_0 t_{\max}}{K}, \quad \min_{k=1,2,\ldots,K} \left\|\frac{\boldsymbol{v}_k}{t_k}\right\|_2^2 \le \frac{\epsilon_0}{t_{\min}K},$$

with the constant decided by line-search strategies.

### 4.4 Algorithm

The final algorithm is to repeat the line-search until convergence. The pseudocode is provided in Algorithm 3, which unifies the PGS method in this section and the accelerated methods (A-PGS, AM-PGS) to be discussed shortly. Here we give additional components to complete the PGS method, before moving to its accelerated versions.

#### 4.4.1 Maximum proxy step-size $t'_{\max}$

While the constant $L$ in Assumption 1 (*i.e.,* inequality (24)) exists ubiquitously, it is often not clear how to obtain $L$ in closed-form except for certain types of $g(\cdot)$ *e.g.,* $g(\boldsymbol{x}) = \boldsymbol{x}^\mathsf{T}\boldsymbol{A}\boldsymbol{x}$ where $L = 2\sigma_{\max}(\boldsymbol{A})$ (see Section 6.1). Importantly, the existence of $L$ is used to establish proofs, but its actual value is never required to be known explicitly. Instead, to complete Algorithm 1, we need to decide $t'_{\max}$.

**Setting $t'_{\max}$ from known $L$.** In line-search criterion (25), $L$ establishes a lower-bound for the search, where if step-size $t < 1/L$ then the total cost is guaranteed to descend (Theorem 2). While we work with proxy step-size $t'$ and control step-size $t$ by $t = \phi(t')$ in its monotone region, the value $1/L$ provides a good reference for the maximum proxy step-size $t'_{\max}$. Practically, if $L$ is known ahead, we recommend using $t'_{\max} = 1/L$.

**Setting $t'_{\max}$ from line-search.** Nonetheless, if $L$ is unknown (which is the usual case), $t'_{\max}$ can be decided effectively from line-search. Such a line-search process is described in Algorithm 2, which is well-defined owing to the existence of $L$. Numerical examples are provided in Fig. 3. We compare the searched $t'_{\max}$ and the known reference proxy step-size $t' = 1/L$ using their ratio, and see that a proper $t'_{\max}$ close to $1/L$ can be found cheaply within 5 - 10 iterations.

**Adaptive maximum proxy step-size $t'_{\max}$.** We observe that typically the corresponding $t = \phi(t') = t'/c(t')$ is slightly larger than $t'$ for $t'$ around $1/L$, as $c(t')$ is slightly below

**Algorithm 2:** Search for proxy step-size $t'_{\max}$.

**function** $t'_{\max} \leftarrow$ **searchMaxProxyStepsize** $(\boldsymbol{x}_0)$
  $found \leftarrow false$, $ub \leftarrow 0.7/h(\boldsymbol{x}_0)$, $t' \leftarrow ub$
  $\boldsymbol{z} \leftarrow \operatorname{prox}_{|t'|h}(\boldsymbol{x}_0 - t' \operatorname{grad} g|_{\boldsymbol{x}_0})$
  $\boldsymbol{v}_0 \leftarrow \frac{1}{\boldsymbol{x}_0^\top \boldsymbol{z}} \boldsymbol{z} - \boldsymbol{x}_0$
  $t \leftarrow \frac{1}{\boldsymbol{x}_0^\top \boldsymbol{z}} t'$
  $Q_L = g(\boldsymbol{x}_0) + \langle \operatorname{grad} g|_{\boldsymbol{x}_0}, \boldsymbol{v}_0 \rangle + \frac{1}{2t} \langle \boldsymbol{v}_0, \boldsymbol{v}_0 \rangle$
  **if** $g(\mathcal{R}_{\boldsymbol{x}_0}(\boldsymbol{v}_0)) \leq Q_L$ **then**
    **if** $t' = ub$ **then return** $t'_{\max} \leftarrow ub$
    $found \leftarrow true$, $t' \leftarrow \min\{2t', ub\}$, goto step 3
  **else if** $found = true$ **then return** $t'_{\max} \leftarrow 0.5t'$
  **else** $t' \leftarrow 0.1t'$, goto step 3
**end**

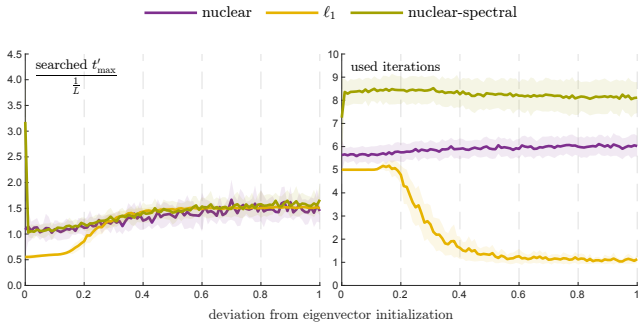



Fig. 3: Examples of line-searched maximum proxy step-size $t'_{\max}$, using different initializations $\boldsymbol{x}_0$ (detailed in Section 7.1.1). On the left we report the ratio $t'_{\max}/(1/L)$, and on the right the used iterations.

1 as seen in Fig. 2. Thus setting $t' = 1/L$ does not necessarily guarantee a line-search success from criterion (25) established on $t$. On the other hand, criterion (25) can also be satisfied for some $t$ greater than $1/L$. This motivates us to use an adaptive $t'_{\max}$, where we set $t'_{\max}$ to the working $t'$ obtained at the previous iteration. This choice is controlled by the *AdaptiveMaxProxyStepsize* flag in Algorithm 3. We shall see such a strategy is useful to reduce subsequent total line-search iterations if $t'_{\max}$ is initially obtained from the line-search in Algorithm 2 (see results in Fig. 5).

#### 4.4.2 Stop criteria for convergence

In general, we propose to monitor at least $\|\boldsymbol{v}_k\|_2$ and $\|\boldsymbol{v}_k/t_k\|_2$ as the stop criteria for convergence. Numerical examples are provided in Fig. 4. While other options are also possible, *e.g.*, by monitoring $f(\cdot)$, these two indicators are important for the following reasons.

**Convergence of estimates.** The convergence of the estimates can be determined from the distance of $\boldsymbol{x}_k$ and $\boldsymbol{x}_{k+1}$, *e.g.*, by using the chordal distance $d = \|\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\|_2$ or the angle $\theta = \operatorname{acos}(\boldsymbol{x}_{k+1}^\top \boldsymbol{x}_k)$. Here we propose to check the length of the tangent vector $\|\boldsymbol{v}_k\|_2$. In fact, $\boldsymbol{v}_k = \mathcal{R}_{\boldsymbol{x}_k}^{-1}(\boldsymbol{x}_{k+1})$ with $\mathcal{R}_{\cdot}^{-1}(\cdot)$ to be defined in equation (36), thus measuring the distance of $\boldsymbol{x}_k$ and $\boldsymbol{x}_{k-1}$.

**Optimality as critical points.** If $\boldsymbol{v}_k/t_k = \boldsymbol{0}$ and $\boldsymbol{v}_k = \boldsymbol{0}$, equation (27) becomes equation (33), thus $\boldsymbol{x}_k$ admits a critical point of problem (2) by satisfying the first-order

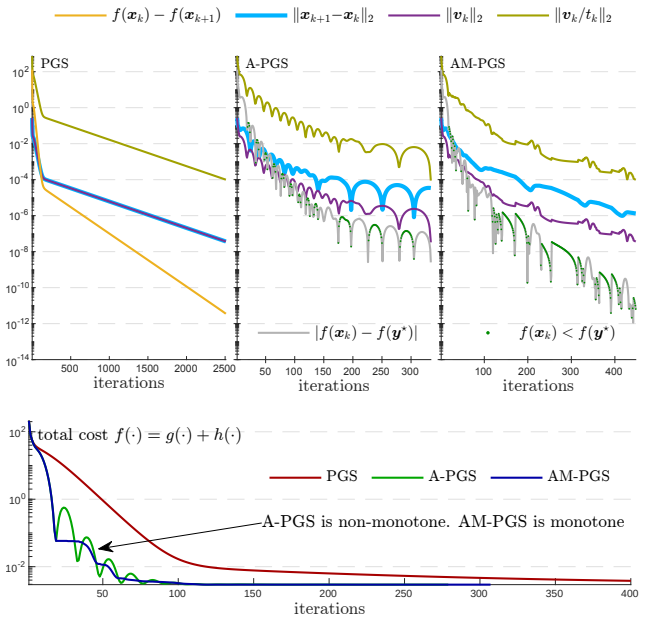



Fig. 4: The convergence behavior of the PGS, A-PGS and AM-PGS methods. The example is drawn from self-calibration with nuclear norm regularization. For each method, we report the convergence of the estimate as $\|\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\|_2$, of the tangent vector as $\|\boldsymbol{v}_k\|_2$ and of the first-order optimality as $\|\boldsymbol{v}_k/t_k\|_2$. The progression of the total cost $f(\boldsymbol{x}_k)$ for each method is plotted in the second figure. In the first figure, for PGS, we report the convergence of the total cost as $f(\boldsymbol{x}_k) - f(\boldsymbol{x}_{k+1})$ which is always positive from Theorem 2. For A-PGS and AM-PGS, we report the absolute value $|f(\boldsymbol{x}_k) - f(\boldsymbol{y}^\star)|$ and mark down cases where $f(\boldsymbol{x}_k) < f(\boldsymbol{y}^\star)$.

optimality condition. For a deeper understanding, we see from the KKT system (17):

$$\begin{aligned}\frac{\boldsymbol{v}_k}{t_k} &= \frac{1}{t'_k}(\boldsymbol{I} - \boldsymbol{x}_k \boldsymbol{x}_k^\top) \operatorname{prox}_{|t'_k|h}(\boldsymbol{x}_k - t'_k \operatorname{grad} g|_{\boldsymbol{x}_k}) \\ &= \operatorname{proj}_{\mathcal{T}_{\boldsymbol{x}_k}\mathcal{S}} \operatorname{prox}_h(\frac{1}{t'_k}\boldsymbol{x}_k - \operatorname{grad} g|_{\boldsymbol{x}_k}).\end{aligned}$$

Thus $\boldsymbol{v}_k/t_k$ is to project the proximal of $\frac{1}{t'_k}\boldsymbol{x}_k - \operatorname{grad} g|_{\boldsymbol{x}_k}$ into the tangent space at $\boldsymbol{x}_k$. A critical point is where this tangent projection goes to zero.

#### 4.4.3 Initialization $\boldsymbol{x}_0$

Typically, it is a good idea to initialize the regularized problem (2) from the solution of the original problem (1). However this choice is problem dependent and should be discussed specifically according to the problem at hand. For example, we initialize the regularized problems for fundamental matrix estimation and correspondence association with the solution of the Rayleigh quotient optimization (the original problem of these instances). However, for self-calibration, the regularized problems are initialized from the canonical DAQ after quasi-calibration, as the solution space of the original problem is likely to be ambiguous due to the critical motion sequence. Details are given in Section 6, and numerical validations provided in Section 7.1.

## 5 Acceleration using the Nesterov Momentum Technique

The conventional PGS method evaluates the gradient and proximal at the current estimate $\boldsymbol{x}_k$.

For the Nesterov momentum technique, the gradient and the proximal are instead evaluated at an auxiliary state $\boldsymbol{y}_k$ defined as a linear combination of the current estimate $\boldsymbol{x}_k$ and the previous estimate $\boldsymbol{x}_{k-1}$ [12], [14], [33]. In the Euclidean case, one accelerated iteration is defined as:

$$\begin{cases} \boldsymbol{x}_{k+1} = \boldsymbol{y}_k + \boldsymbol{v}_k \\ \boldsymbol{y}_{k+1} = \boldsymbol{x}_{k+1} + \frac{1-\alpha_k}{\alpha_{k+1}} \left( \boldsymbol{x}_k - \boldsymbol{x}_{k+1} \right), \quad \boldsymbol{y}_0 = \boldsymbol{x}_0, \end{cases} \tag{34}$$

where $\boldsymbol{v}_k$ is obtained by evaluating iteration (6) at $\boldsymbol{y}_k$ as:

$$\boldsymbol{v}_k = \underset{\boldsymbol{v} \in \mathbb{R}^n}{\arg\min} \, \langle \nabla g|_{\boldsymbol{y}_k}, \boldsymbol{v} \rangle + \frac{1}{2t} \langle \boldsymbol{v}, \boldsymbol{v} \rangle + h(\boldsymbol{y}_k + \boldsymbol{v}),$$

and the scalar sequence $\alpha_k$ $(k = 1, 2, \dots)$ is defined as:

$$\alpha_{k+1} = \frac{1 + \sqrt{1 + 4\alpha_k^2}}{2}, \quad \alpha_0 = 1. \tag{35}$$

The iteration defined above was first proposed by Nesterov for smooth optimization [33], and later extended to non-smooth composite optimization in [12], [14]. The original proof shows that the accelerated iteration attains quadratic convergence for convex cost functions. These results are also valid if the cost function is locally convex around a local minimum. Although Riemannian manifolds are typically non-convex, it has been shown that the Nesterov sequence can attain quadratic convergence rate for geodesically convex optimization problems on the manifold [44], [45].

To extend the above result to the sphere manifold, we need to evaluate the difference between $\boldsymbol{x}_{k+1}$ and $\boldsymbol{x}_k$ on the sphere. Inspired by [40], we define this difference as a vector $\Delta\boldsymbol{v}$ in the tangent space of $\boldsymbol{x}_{k+1}$, thus the first summation can be extended by the retraction at $\boldsymbol{x}_{k+1}$. Such $\Delta\boldsymbol{v}$ must satisfy $\mathcal{R}_{\boldsymbol{x}_{k+1}}(\Delta\boldsymbol{v}) = \boldsymbol{x}_k$. Abusing notations, we define the inverse of the retraction, $\Delta\boldsymbol{v} = \mathcal{R}^{-1}_{\boldsymbol{x}_{k+1}}(\boldsymbol{x}_k)$, which can be calculated in closed-form as:

$$\mathcal{R}^{-1}_{\boldsymbol{x}_{k+1}}(\boldsymbol{x}_k) \overset{\text{def}}{=} \frac{1}{\boldsymbol{x}_k^{\mathsf{T}} \boldsymbol{x}_{k+1}} \boldsymbol{x}_k - \boldsymbol{x}_{k+1}. \tag{36}$$

It can be easily verified that $\mathcal{R}_{\boldsymbol{x}_{k+1}}(\mathcal{R}^{-1}_{\boldsymbol{x}_{k+1}}(\boldsymbol{x}_k)) = \boldsymbol{x}_k$ and $\mathcal{R}^{-1}_{\boldsymbol{x}_{k+1}}(\boldsymbol{x}_k) \in \mathcal{T}_{\boldsymbol{x}_{x+1}}\mathcal{S}$ for any $\boldsymbol{x}_k$, $\boldsymbol{x}_{k+1} \in \mathcal{S}$.

Overall we extend the Nesterov sequence to the sphere manifold as:

$$\begin{cases} \boldsymbol{x}_{k+1} = \mathcal{R}_{\boldsymbol{y}_k}(\boldsymbol{v}_k) \\ \boldsymbol{y}_{k+1} = \mathcal{R}_{\boldsymbol{x}_{k+1}}\left( \frac{1-\alpha_k}{\alpha_{k+1}} \mathcal{R}^{-1}_{\boldsymbol{x}_{k+1}}(\boldsymbol{x}_k) \right), \quad \boldsymbol{y}_0 = \boldsymbol{x}_0. \end{cases} \tag{37}$$

The extension is to replace Euclidean addition and subtraction with Riemannian retraction $\mathcal{R}_{\cdot}(\cdot)$ and its inverse $\mathcal{R}^{-1}_{\cdot}(\cdot)$. Here $\boldsymbol{x}_k$ is obtained by evaluating iteration (13) at $\boldsymbol{y}_k$ as:

$$\boldsymbol{v}_k = \underset{\boldsymbol{v} \in \mathcal{T}_{\boldsymbol{y}_k}\mathcal{S}}{\arg\min} \, \langle \operatorname{grad} g|_{\boldsymbol{y}_k}, \boldsymbol{v} \rangle + \frac{1}{2t} \langle \boldsymbol{v}, \boldsymbol{v} \rangle + h(\boldsymbol{y}_k + \boldsymbol{v}),$$

and the scalar $\alpha_k$ is defined as in equation (35). The tangent update $\boldsymbol{v}_k$ can be solved in closed-form using the proxy step-size technique proposed in Section 4.

The estimates $\boldsymbol{x}_k$ $(k = 1, 2, \dots)$ generated from equation (37) do not guarantee the monotonicity of the total cost $f(\boldsymbol{x}_k) = g(\boldsymbol{x}_k) + h(\boldsymbol{x}_k)$ [46]. On the manifold setting, this can sometimes lead to divergence if $\boldsymbol{v}_k$ is computed based on the SSNM method [40]. To detect and recover from potential failures, the authors in [40] introduced a safeguard by monitoring the progression of the cost function within several iterations. One potential reason for the divergence is the inexact computation of each iteration [46]:

> "In our case, where the denoising subproblems are not solved exactly, monotonicity becomes an important issue. It might happen that due to the inexact computations of the denoising subproblems, the algorithm might become extremely non-monotone and in fact can even diverge!"

Practically in our experiments to be presented in Section 7, we did not observe the divergence of iterations by using the proxy step-size to obtain the tangent update $\boldsymbol{v}_k$. This may be due to the fact that we solve each iteration exactly (in closed-form) while the SSNM method used in [29], [30], [40] is iterative thus incurring inexact solutions.

Nonetheless, we propose a monotone algorithm for the sphere manifold based on Beck *et al.*'s [46] Euclidean version which has been proved to retain the quadratic convergence rate. Beck *et al.*'s [46] monotone algorithm is defined as:

$$\begin{cases} \boldsymbol{y}^\star = \boldsymbol{y}_k + \boldsymbol{v}_k \\ \boldsymbol{x}_{k+1} = \begin{cases} \boldsymbol{y}^\star & \text{if } f(\boldsymbol{y}^\star) < f(\boldsymbol{x}_k) \\ \boldsymbol{x}_k & \text{otherwise} \end{cases} \\ \boldsymbol{y}_{k+1} = \begin{cases} \boldsymbol{y}^\star + \frac{1-\alpha_k}{\alpha_{k+1}} (\boldsymbol{x}_k - \boldsymbol{y}^\star) & \text{if } f(\boldsymbol{y}^\star) < f(\boldsymbol{x}_k) \\ \boldsymbol{x}_k + \frac{\alpha_k}{\alpha_{k+1}} (\boldsymbol{y}^\star - \boldsymbol{x}_k) & \text{otherwise.} \end{cases} \end{cases}$$

The above iteration ensures the monotonicity of the total cost $f(\cdot)$ by leveraging between the new estimate $\boldsymbol{y}^\star$ and the previous estimate $\boldsymbol{x}_k$. We extend Beck *et al.*'s monotone algorithm to the sphere manifold as follow:

$$\begin{cases} \boldsymbol{y}^\star = \mathcal{R}_{\boldsymbol{y}_k}(\boldsymbol{v}_k) \\ \boldsymbol{x}_{k+1} = \begin{cases} \boldsymbol{y}^\star & \text{if } f(\boldsymbol{y}^\star) < f(\boldsymbol{x}_k) \\ \boldsymbol{x}_k & \text{otherwise} \end{cases} \\ \boldsymbol{y}_{k+1} = \begin{cases} \mathcal{R}_{\boldsymbol{y}^\star}\left( \frac{1-\alpha_k}{\alpha_{k+1}} \mathcal{R}^{-1}_{\boldsymbol{y}^\star}(\boldsymbol{x}_k) \right) & \text{if } f(\boldsymbol{y}^\star) < f(\boldsymbol{x}_k) \\ \mathcal{R}_{\boldsymbol{x}_k}\left( \frac{\alpha_k}{\alpha_{k+1}} \mathcal{R}^{-1}_{\boldsymbol{x}_k}(\boldsymbol{y}^\star) \right) & \text{otherwise.} \end{cases} \end{cases}$$

The accelerated PGS (A-PGS), *i.e.*, Nesterov sequence, and the accelerated monotone PGS (AM-PGS), *i.e.*, Beck's sequence, are implemented as pseudocode in Algorithm 3. An illustration of the convergence is given in Fig. 4.

## 6 Applications

### 6.1 Rayleigh Quotient Optimization

We consider $g(\boldsymbol{x}) = \boldsymbol{x}^{\mathsf{T}}\boldsymbol{A}\boldsymbol{x}$ with $\boldsymbol{A}$ being a symmetric matrix. The Euclidean gradient at $\boldsymbol{x} \in \mathbb{R}^n$ is $\nabla g|_{\boldsymbol{x}} = 2\boldsymbol{A}\boldsymbol{x}$. The Riemannian gradient at $x \in \mathcal{S}$ is:

$$\begin{aligned} \operatorname{grad} g(\boldsymbol{x}) &= \nabla g|_{\boldsymbol{x}} - \langle \boldsymbol{x}, \nabla g|_{\boldsymbol{x}} \rangle \, \boldsymbol{x} \\ &= 2\boldsymbol{A}\boldsymbol{x} - 2 \langle \boldsymbol{x}, \boldsymbol{A}\boldsymbol{x} \rangle \, \boldsymbol{x} = 2\boldsymbol{A}\boldsymbol{x} - 2(\boldsymbol{x}^{\mathsf{T}}\boldsymbol{A}\boldsymbol{x})\boldsymbol{x}. \end{aligned} \tag{38}$$

**Algorithm 3:** A unified implementation of the PGS, A-PGS and AM-PGS methods.

**input** : method = PGS or A-PGS or AM-PGS
**input** : $\boldsymbol{x}_0$
**if** *known Lipschitz constant* $L$ **then** $t'_{\max} \leftarrow 1/L$
**else** $t'_{\max} \leftarrow$ **searchMaxProxyStepsize** $(\boldsymbol{x}_0)$
$\boldsymbol{y}_0 \leftarrow \boldsymbol{x}_0$, $\alpha_0 \leftarrow 1$, $k \leftarrow 0$
**while** $k < maxIterations$ **do**
  $(\boldsymbol{v}_k, t, t') \leftarrow$ **lineSearch** $(\boldsymbol{y}_k, t'_{\max})$
  $\boldsymbol{y}^\star \leftarrow \mathcal{R}_{\boldsymbol{y}_k}(\boldsymbol{v}_k)$
  **if** *AdaptiveMaxProxyStepsize* **then** $t'_{\max} \leftarrow t'$
  **if** method = PGS **then**
    $\boldsymbol{x}_{k+1} \leftarrow \boldsymbol{y}^\star$
    $\boldsymbol{y}_{k+1} \leftarrow \boldsymbol{y}^\star$
  **else if** method = A-PGS *or* AM-PGS **then**
    $\alpha_{k+1} \leftarrow \frac{1+\sqrt{1+4\alpha_k^2}}{2}$
    $\boldsymbol{x}_{k+1} \leftarrow \boldsymbol{y}^\star$
    $\boldsymbol{y}_{k+1} \leftarrow \mathcal{R}_{\boldsymbol{y}^\star}\left(\frac{1-\alpha_k}{\alpha_{k+1}}\mathcal{R}^{-1}_{\boldsymbol{y}^\star}(\boldsymbol{x}_k)\right)$
    **if** method = AM-PGS **then**
      **if** $g(\boldsymbol{y}^\star) + h(\boldsymbol{y}^\star) > g(\boldsymbol{x}_k) + h(\boldsymbol{x}_k)$ **then**
        $\boldsymbol{x}_{k+1} \leftarrow \boldsymbol{x}_k$
        $\boldsymbol{y}_{k+1} \leftarrow \mathcal{R}_{\boldsymbol{x}_k}\left(\frac{\alpha_k}{\alpha_{k+1}}\mathcal{R}^{-1}_{\boldsymbol{x}_k}(\boldsymbol{y}^\star)\right)$
      **end**
    **end**
  **end**
  **if** $\|\boldsymbol{v}_k\|_2 < 1e-5$ *and* $\|\boldsymbol{v}_k/t\|_2 < 1e-3$ **then**
    **return** $\boldsymbol{x}_{k+1}$
  **else** $k \leftarrow k+1$
**end**

Using retraction (12), the Lipschitz-type constant of $g(\mathcal{R}_{\boldsymbol{x}}(\boldsymbol{v}))$ is $L = 2\sigma_{\max}(\boldsymbol{A})$ (see Appendix H) where $\sigma_{\max}(\boldsymbol{A})$ denotes the largest singular value of $\boldsymbol{A}$.

Minimizing $g(\boldsymbol{x}) = \boldsymbol{x}^\mathsf{T}\boldsymbol{A}\boldsymbol{x}$ on the sphere manifold is called Rayleigh quotient optimization:

$$\boldsymbol{x}_0 = \underset{\boldsymbol{x}\in\mathbb{R}^n}{\arg\min}\ \boldsymbol{x}^\mathsf{T}\boldsymbol{A}\boldsymbol{x} \qquad \text{s.t.} \quad \|\boldsymbol{x}\|_2 = 1. \tag{39}$$

The solution to this problem is given in closed-form, which is the bottom eigenvector of $\boldsymbol{A}$ (*i.e.*, the eigenvector associated with the smallest eigenvalue), which often gives as a good initialization for its regularized versions.

### 6.2 Fundamental Matrix Estimation

#### 6.2.1 Problem Statement

The fundamental matrix is a key algebraic model of the two-view geometry [1], [3], [47]. We denote $\boldsymbol{p}_i \leftrightarrow \boldsymbol{p}'_i$ $(i = [1:m])$ the homogeneous coordinates of corresponding points in two images. In the noise-free case, the epipolar constraint holds as:

$$\boldsymbol{p}_i'^\mathsf{T}\boldsymbol{F}\boldsymbol{p}_i = 0, \quad i = [1:m]. \tag{40}$$

The matrix $\boldsymbol{F} \in \mathbb{R}^{3\times3}$ is called the *fundamental matrix*, defined up to scale, thus we seek for $\boldsymbol{F}$ on the unit sphere such that $\|\boldsymbol{F}\|_F = 1$. Importantly, $\mathrm{rank}(\boldsymbol{F}) = 2$ is required. A related concept is the essential matrix for normalized calibration, for which we refer to a recent work [48].

To estimate $\boldsymbol{F}$, we can formulate a cost function based on the algebraic error as:

$$\begin{aligned}\varphi(\boldsymbol{F}) &= \frac{1}{m}\sum_{i=1}^{m}\left(\boldsymbol{p}_i'^\mathsf{T}\boldsymbol{F}\boldsymbol{p}_i\right)^2 = \frac{1}{m}\sum_{i=1}^{m}\left(\left(\boldsymbol{p}_i^\mathsf{T}\otimes\boldsymbol{p}_i'^\mathsf{T}\right)\mathrm{vec}\left(\boldsymbol{F}\right)\right)^2 \\ &= \frac{1}{m}\left\|\boldsymbol{H}\mathrm{vec}\left(\boldsymbol{F}\right)\right\|_2^2,\end{aligned}$$

where the $i$-th row of $\boldsymbol{H}$ is $\boldsymbol{p}_i^\mathsf{T}\otimes\boldsymbol{p}_i'^\mathsf{T}$. We denote $\boldsymbol{x} = \mathrm{vec}\left(\boldsymbol{F}\right)$ and $\boldsymbol{F} = \mathrm{mat}\left(\boldsymbol{x}\right)$, where $\mathrm{vec}(\cdot)$ is the standard column-wise matrix vectorization and $\mathrm{mat}(\cdot)$ is its inverse operation. Defining $\boldsymbol{A} = \frac{1}{m}\boldsymbol{H}^\mathsf{T}\boldsymbol{H}$, we see the fundamental matrix problem is an instance of problem (39).

The solution $\boldsymbol{F}_0 = \mathrm{mat}(\boldsymbol{x}_0)$ solved from problem (39) is usually of rank 3. A remedy is to subsequently round the solution $\boldsymbol{F}_0$ using the rank-2 approximation via the Singular Value Decomposition (SVD). This two-stage solution is the eight-point algorithm. Instead, we give a low-rank solution using the nuclear norm regularization.

#### 6.2.2 Nuclear Norm Regularization

The nuclear norm of a matrix $\|\boldsymbol{X}\|_*$, defined as the summation of its singular values, is the tightest convex envelop of the rank function within the unit ball $\{\boldsymbol{X}\in\mathbb{R}^{m\times n} : \|\boldsymbol{X}\|_2 \le 1\}$ [9]. Nuclear norm regularization has been widely used as a technique to promote low-rank in Euclidean optimization problems [13], while a direct deployment to the manifold setting seems to be obscure with the results in [29], [32], mainly due to the challenge incurred in evaluating the generalized Jacobian matrix of a non-smooth function. In contrast, our technique can handle nuclear norm regularization with no effort. We apply nuclear norm regularization to problem (39) as:

$$\underset{\boldsymbol{x}\in\mathbb{R}^n}{\arg\min}\ \boldsymbol{x}^\mathsf{T}\boldsymbol{A}\boldsymbol{x} + \lambda\|\mathrm{mat}\left(\boldsymbol{x}\right)\|_* \quad \text{s.t.} \quad \|\boldsymbol{x}\|_2 = 1, \tag{41}$$

where $\lambda \ge 0$ is a given constant controlling the strength of regularization.

In problem (41), the regularization term is $h(\boldsymbol{x}) = \lambda\left\|\mathrm{mat}\left(\boldsymbol{x}\right)\right\|_*$, which is convex but non-smooth. Besides, $h(\cdot)$ is absolutely homogeneous, *i.e.*, $h(\alpha\boldsymbol{x}) = |\alpha|\,h(\boldsymbol{x})$, because:

$$\lambda\left\|\mathrm{mat}\left(\alpha\boldsymbol{x}\right)\right\|_* = \lambda\left\|\alpha\mathrm{mat}\left(\boldsymbol{x}\right)\right\|_* = \lambda\left|\alpha\right|\left\|\mathrm{mat}\left(\boldsymbol{x}\right)\right\|_*. \tag{42}$$

The first equality holds because $\mathrm{mat}\left(\boldsymbol{x}\right)$ is a linear operator, and the second because norms are absolutely homogeneous.

Let $\boldsymbol{X} = \mathrm{mat}\left(\boldsymbol{x}\right)$ and denote its SVD as $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\mathsf{T}$. For the nuclear norm function $\lambda\|\boldsymbol{X}\|_*$, the proximal is given in closed-form [49], where $\mathrm{prox}_{t\lambda\|\cdot\|_*}(\boldsymbol{X}) = \boldsymbol{U}(\boldsymbol{\Sigma} - t\lambda\boldsymbol{I})_+\boldsymbol{V}^\mathsf{T}$. Here $\boldsymbol{A}_+ \overset{\text{def}}{=} \max\{\boldsymbol{A}, \boldsymbol{0}\}$ proceeds element-wise. Thus we obtain the proximal of $h(\cdot)$ as:

$$\mathrm{prox}_{th}\left(\boldsymbol{x}\right) = \mathrm{vec}\left(\boldsymbol{U}\left(\boldsymbol{\Sigma} - t\lambda\boldsymbol{I}\right)_+\boldsymbol{V}^\mathsf{T}\right). \tag{43}$$

### 6.3 Correspondence Association

#### 6.3.1 Problem Statement

The correspondence association problem using pairwise constraints can be formulated as a Rayleigh quotient optimization as well [5]. We denote the association hypothesis that a point $i$ in the point-cloud $\mathcal{Q}$ is matched with a point

$i'$ in the point-cloud $\mathcal{Q}'$ as $\mathfrak{h}_{ii'}$. The correspondence problem is to estimate the likelihood $p(\mathfrak{h}_{ii'})$ ($i \in \mathcal{Q}$, $i' \in \mathcal{Q}'$) of all possible association hypotheses, collected as components of the state vector $\boldsymbol{x}$.

To that end, we can design an adjacency matrix $\boldsymbol{M}$ from the pairwise consistency of hypotheses [5]. A common practice is based on the change of distance:

$$\boldsymbol{M}(\mathfrak{h}_{ii'}, \mathfrak{h}_{jj'}) = \begin{cases} 4.5 - \frac{(d_{ij} - d_{i'j'})^2}{2\delta_d^2} & \text{if } |d_{ij} - d_{i'j'}| < 3\delta_d \\ 0 & \text{otherwise,} \end{cases} \tag{44}$$

where $d_{ij}$ is the Euclidean distance between the points $i$ and $j$ in $\mathcal{Q}$, $d_{i'j'}$ the distance between the points $i'$ and $j'$ in $\mathcal{Q}'$, and $\delta_d$ a tuning parameter.

The resulting problem is formalized as maximizing the overall consistency $\boldsymbol{x}^\mathsf{T}\boldsymbol{M}\boldsymbol{x}$ on the unit sphere, as an instance of problem (39) by letting $\boldsymbol{A} = -\boldsymbol{M}$. The estimate of $\boldsymbol{x}$ is further used to decide the final correspondences, based on various assumptions, *e.g.,* one point in $\mathcal{Q}$ can only be matched with one point in $\mathcal{Q}'$ [5].

The match hypotheses solved this way are dense, while many of them present with contradictions or low probabilities. It is thus favorable to have a sparse $\boldsymbol{x}$, where some unlikely hypotheses and contradictions are pruned away. We give such a sparse solution by $\ell_1$ norm regularization.

#### 6.3.2 $\ell_1$ Norm Regularization

It has been known that $\ell_1$ norm regularization can favor sparsity in Euclidean [12] and manifold optimization [29], [32]. We apply $\ell_1$ norm regularization to problem (39) as:

$$\underset{\boldsymbol{x} \in \mathbb{R}^n}{\arg\min}\ \boldsymbol{x}^\mathsf{T}\boldsymbol{A}\boldsymbol{x} + \lambda\|\boldsymbol{x}\|_{\ell_1} \quad \text{s.t.} \quad \|\boldsymbol{x}\|_2 = 1, \tag{45}$$

where $\lambda \geq 0$ is a given constant controlling the strength of regularization. In this case, $h(\boldsymbol{x}) = \lambda\|\boldsymbol{x}\|_{\ell_1}$, which is convex and absolutely homogeneous. The proximal of $h(\boldsymbol{x})$, often termed soft shrinkage operator, is given element-wise as:

$$\mathrm{prox}_{th}\left(\boldsymbol{x}\right)_i = \mathrm{sgn}(\boldsymbol{x}_i)(|\boldsymbol{x}_i| - t\lambda)_+. \tag{46}$$

### 6.4 Camera Self-calibration

#### 6.4.1 Problem Statement

In projective reconstruction, we obtain a set of projective cameras $\tilde{\boldsymbol{P}}_i \in \mathbb{R}^{3\times4}$:

$$\tilde{\boldsymbol{P}}_i \propto \boldsymbol{P}_i\boldsymbol{H}^{-1} \quad (i \in [1:n]), \tag{47}$$

which differ from the Euclidean cameras $\boldsymbol{P}_i \in \mathbb{R}^{3\times4}$ by a common projective transformation $\boldsymbol{H} \in \mathbb{R}^{4\times4}$. The camera self-calibration problem is to infer $\boldsymbol{H}$ from $\tilde{\boldsymbol{P}}_i$ [3].

The key algebraic model to this task is the *Dual Absolute Quadric (DAQ)*, a rank-3 symmetric matrix in $\mathbb{R}^{4\times4}$ defined up to scale [2]. In specific, the DAQ in the Euclidean space, termed the canonical DAQ, takes the form $\Omega_\infty^* = \mathrm{diag}(1,1,1,0)$. We denote the DAQ in the projective space (where $\tilde{\boldsymbol{P}}_i$ is defined) by $\boldsymbol{Q}_\infty^* = \boldsymbol{H}\Omega_\infty^*\boldsymbol{H}^\mathsf{T}$. The image of the DAQ, denoted by $\omega_i^*$, is invariant under $\boldsymbol{H}$:

$$\omega_i^* \propto \tilde{\boldsymbol{P}}_i\boldsymbol{Q}_\infty^*\tilde{\boldsymbol{P}}_i^\mathsf{T} \propto \boldsymbol{P}_i\Omega_\infty^*\boldsymbol{P}_i^\mathsf{T} \propto \boldsymbol{K}_i\boldsymbol{K}_i^\mathsf{T}, \tag{48}$$

where $\boldsymbol{K}_i \in \mathbb{R}^{3\times3}$ is the intrinsic matrix of $\boldsymbol{P}_i$. At its core, the self-calibration problem is to estimate $\boldsymbol{Q}_\infty^*$ from equation (48) using various constraints on $\boldsymbol{K}_i$.

Here we consider a linear approach developed for cameras with varying focal lengths [50]. In this case, we have $\boldsymbol{K}_i\boldsymbol{K}_i^\mathsf{T} \propto \mathrm{diag}(f_i^2, f_i^2, 1)$. Based on equation (48), we have:

$$\begin{cases} \tilde{\boldsymbol{a}}_i^\mathsf{T}\boldsymbol{Q}_\infty^*\tilde{\boldsymbol{a}}_i = \tilde{\boldsymbol{b}}_i^\mathsf{T}\boldsymbol{Q}_\infty^*\tilde{\boldsymbol{b}}_i & \text{(49a)} \\ \tilde{\boldsymbol{a}}_i^\mathsf{T}\boldsymbol{Q}_\infty^*\tilde{\boldsymbol{b}}_i = 0,\ \tilde{\boldsymbol{a}}_i^\mathsf{T}\boldsymbol{Q}_\infty^*\tilde{\boldsymbol{c}}_i = 0,\ \tilde{\boldsymbol{b}}_i^\mathsf{T}\boldsymbol{Q}_\infty^*\tilde{\boldsymbol{c}}_i = 0, & \text{(49b)} \end{cases}$$

where $\tilde{\boldsymbol{P}}_i^\mathsf{T} = \begin{bmatrix} \tilde{\boldsymbol{a}}_i & \tilde{\boldsymbol{b}}_i & \tilde{\boldsymbol{c}}_i \end{bmatrix}$. Equation (49) is linear in $\boldsymbol{Q}_\infty^*$ thus can be rewritten as:

$$\boldsymbol{M}_i\,\mathrm{vec_t}\left(\boldsymbol{Q}_\infty^*\right) = \boldsymbol{M}_i\,\boldsymbol{x} = \boldsymbol{0}, \tag{50}$$

where we have defined $\mathrm{vec_t}\left(\boldsymbol{Q}_\infty^*\right) = \boldsymbol{x} \in \mathbb{R}^{10}$ comprising of the upper triangular elements of $\boldsymbol{Q}_\infty^*$, and $\mathrm{mat_t}\left(\boldsymbol{x}\right) = \boldsymbol{Q}_\infty^*$ its inverse operation. Since $\boldsymbol{Q}_\infty^*$ is defined up to scale so is $\boldsymbol{x}$ and we minimize the cost $\varphi(\boldsymbol{x}) = \frac{1}{n}\sum_{i=1}^n \|\boldsymbol{M}_i\,\boldsymbol{x}\|_2^2$ on the unit sphere. Upon defining $\boldsymbol{A} = \frac{1}{n}\sum_{i=1}^n \boldsymbol{M}_i^\mathsf{T}\boldsymbol{M}_i$, we see it is another instance of problem (39).

Just as the case of the fundamental matrix estimation, the DAQ estimated from solving problem (39) is typically of rank $4$ instead of rank $3$, thus an SVD based rounding process is used subsequently.

Once obtaining a rank-3 estimate of $\boldsymbol{Q}_\infty^*$, we can recover $\boldsymbol{H}$ up to a similarity transformation. This is usually done by the eigen decomposition of $\boldsymbol{Q}_\infty^*$. Let $\boldsymbol{Q}_\infty^* = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\mathsf{T}$, with $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1, \lambda_2, \lambda_3, 0)$. We then set $\boldsymbol{H} \propto \boldsymbol{U}\,\mathrm{diag}(\sqrt{\lambda_1}, \sqrt{\lambda_2}, \sqrt{\lambda_3}, 1)$. The estimate of $\boldsymbol{H}$ determines camera $\boldsymbol{P}_i$, and $\boldsymbol{K}_i$ afterwards by decomposing $\boldsymbol{P}_i$.

#### 6.4.2 Nuclear Norm Regularization

One issue regarding self-calibration is the critical motion sequences (CMS) [51]. The CMSs are camera configurations where self-calibration is ambiguous due to the lack of sufficient constraints. Among which, we consider the artificial CMS that can be resolved by enforcing the rank deficiency of the DAQ during the estimation rather than a posteriori. For the linear self-calibration described in equation (49), one such CMS is that - all the cameras' principal axes intersect at a fixed point, *i.e.,* all the cameras look towards a common point. In this case, there only exist two rank deficient solutions [51]: one rank-3 solution (desired) and one rank-1 solution (undesired).

In analogy to the fundamental matrix estimation, we use nuclear norm regularization to promote low rank. This case is similar to what we have discussed in Section 6.2.2. We omit the details as they can be readily derived by replacing $\mathrm{mat}(\cdot)$ with $\mathrm{mat}_t(\cdot)$ and $\mathrm{vec}(\cdot)$ with $\mathrm{vec}_t(\cdot)$ in problem (41).

#### 6.4.3 Nuclear-Spectral Norm Regularization

The aforementioned nuclear norm regularization resolves the CMS only partly due to the existence of the rank-1 solution. We propose to avoid the rank-1 solution by additionally including a spectral norm to penalize the largest singular value. The nuclear-spectral norm regularizer is:

$$h(\boldsymbol{x}) = \lambda_1\|\mathrm{mat_t}\left(\boldsymbol{x}\right)\|_* + \lambda_2\|\mathrm{mat_t}\left(\boldsymbol{x}\right)\|_2, \tag{51}$$

where $\lambda_1$ and $\lambda_2$ control the regularization strength of each part. We apply this regularizer to problem (39) as:

$$\begin{aligned} &\underset{\boldsymbol{x} \in \mathbb{R}^n}{\arg\min}\ \boldsymbol{x}^\mathsf{T}\boldsymbol{A}\boldsymbol{x} + \lambda_1\|\mathrm{mat_t}\left(\boldsymbol{x}\right)\|_* + \lambda_2\|\mathrm{mat_t}\left(\boldsymbol{x}\right)\|_2 \\ &\quad \text{s.t.} \quad \|\boldsymbol{x}\|_2 = 1. \end{aligned} \tag{52}$$

Denote the SVD of $\mathrm{mat}_{\mathrm{t}}(\boldsymbol{x})$ as $\mathrm{mat}_{\mathrm{t}}(\boldsymbol{x}) = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathsf{T}}$. Since both the nuclear and spectral norm are orthogonal invariant, we can derive the proximal of $h(\cdot)$ in a similar manner as the derivation used for the nuclear norm. The proximal of $h(\cdot)$ in equation (51) is given as follow:

$$\mathrm{prox}_{th}(\boldsymbol{x}) = \mathrm{vec}_{\mathrm{t}}\left(\boldsymbol{U}\left(\boldsymbol{\Sigma} - t\lambda_1\boldsymbol{I} - t\lambda_2\boldsymbol{E}_1\right)_+\boldsymbol{V}^{\mathsf{T}}\right), \quad (53)$$

where $\boldsymbol{E}_1 = \mathrm{diag}(1, 0, \cdots, 0)$ is a diagonal matrix where the top-left element is 1 and the rests are all-zeros.

## 7 Experimental Results

### 7.1 The Proposed PGS Algorithm

We first provide an evaluation of the PGS methods, *i.e.*, PGS, A-PGS and AM-PGS methods in Algorithm 3.

#### 7.1.1 Experiment Setup

**Numerical instances.** We experiment with the Rayleigh quotient optimization, and draw numerical examples from different applications which essentially form different $\boldsymbol{A}$ matrices in problem (39):

- nuclear norm reg. — fundamental matrix estimation with nuclear norm regularization,
- $\ell_1$ norm reg. — correspondence association with $\ell_1$ norm regularization,
- nuclear-spectral norm reg. — self-calibration with nuclear-spectral norm regularization.

In this section, we distinguish these instances by the type of regularization used, *i.e.*, nuclear, $\ell_1$ and nuclear-spectral.

**Initialization.** The numerical examples used in this section are special forms ot the Rayleigh quotient optimization problem (39), and its optimal solution is known to be the bottom eigenvector of matrix $\boldsymbol{A}$ which we denote by $\boldsymbol{\xi}_{\boldsymbol{A}}$. To examine the convergence behavior with respect to different initializations, we create a range of initial values $\boldsymbol{x}_0$ by adding independent zero-mean Gaussian noise element-wisely to $\boldsymbol{\xi}_{\boldsymbol{A}}$. For the $k$-th vector element, we let:

$$\boldsymbol{\xi}[k] \leftarrow \boldsymbol{\xi}_{\boldsymbol{A}}[k] + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \delta_{\mathrm{init}}), \quad (54)$$

and normalize $\boldsymbol{\xi}$ to get the initial value $\boldsymbol{x}_0 = \boldsymbol{\xi}/\|\boldsymbol{\xi}\|_2$. Intuitively, $\delta_{\mathrm{init}}$ controls the deviation from the eigenvector initialization $\boldsymbol{\xi}_{\boldsymbol{A}}$, and if $\delta_{\mathrm{init}}$ is large, the above process simulates random initialization.

**Proxy step-size strategies.** We examine the following proxy step-size strategies.

- LipschitzFixed — $t'_{\max} = 1/L$ initially, and $t'_{\max}$ is kept fixed in the following iterations;
- LipschitzAdaptive — $t'_{\max} = 1/L$ initially, and at each iteration $t'_{\max}$ is updated to the previous working proxy step-size $t'$;
- SearchedFixed — $t'_{\max}$ is obtained from Algorithm 2 initially, and $t'_{\max}$ is kept fixed in the following iterations;
- SearchedAdaptive — $t'_{\max}$ is obtained from Algorithm 2, and at each iteration $t'_{\max}$ is updated to the previous working proxy step-size $t'$.

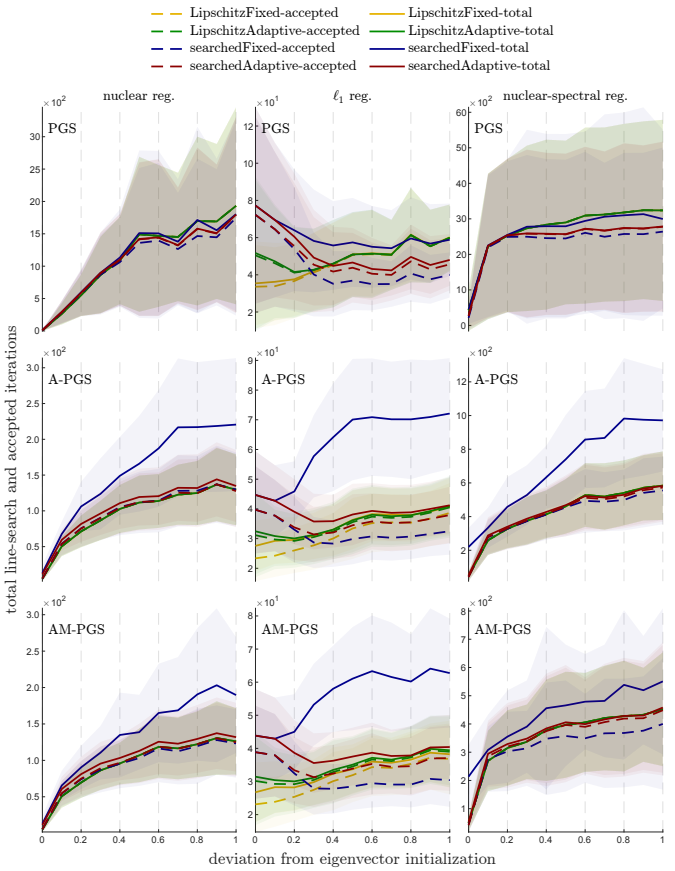



Fig. 5: The convergence of the PGS, A-PGS and AM-PGS methods by using different proxy step-size strategies. Results are reported as the total line-search and the accepted iterations with respect to different initialization.

#### 7.1.2 Results

We report the results with a 20 run Monte-Carlo simulation for each PGS method and each proxy step-size strategy with respect to different initializations.

**Convergence by used iterations.** We examine the convergence by the used iterations in Fig. 5. In particular, we distinguish the total line-search iterations (the overall iterations processed in Algorithm 1) and the accepted iterations (the iterations used in Algorithm 3). The final verdicts is as follows: a) In general, we observe no significant differences for the accepted iterations across different proxy step-size strategies. b) However, the proxy step-size strategy SearchedFixed is not recommended, as it often leads to line-search failures especially when the initialization is bad, as reflected by the number of total line-search iterations. Therefore, if $t'_{\max}$ is obtained from line-search in Algorithm 2, we recommend at each iteration updating $t'_{\max}$ to the previous working proxy step-size $t'$. c) If $t'_{\max}$ is initialized as $1/L$ from the Lipschitz constant $L$, both strategies LipschitzFixed and LipschitzAdaptive give similar results. d) We observe accelerated methods A-PGS and AM-PGS converge much faster than the unaccelerated PGS method, thus are generally recommended.

**Variation of the optimal costs.** We compare the optimal costs of each Monte-Carlo run in Fig. 6. In specific, in the

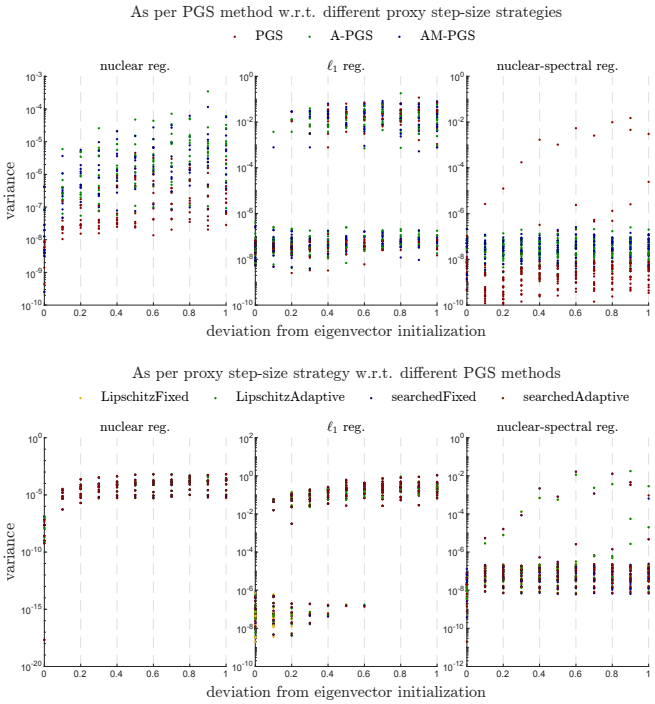



Fig. 6: The variation of the optimal costs for each case.

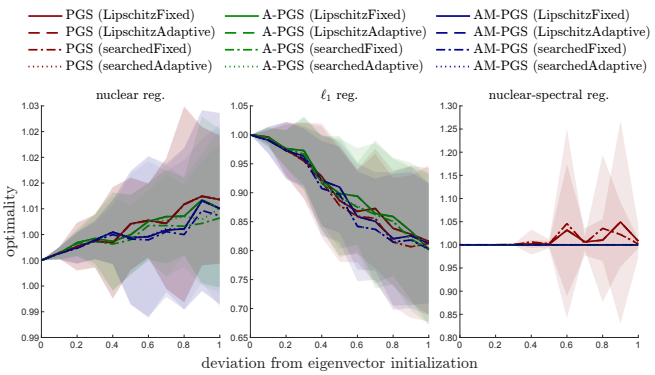



Fig. 7: The impact of different initializations.

top figure, for each PGS method (*i.e.*, PGS/A-PGS/AM-PGS), we compare the costs obtained from different proxy step-size strategies. Likewise, in the bottom figure, for each proxy step-size strategy, we compare the costs obtained from different PGS methods. The difference is evaluated as the variation of the optimal costs. From Fig. 6, we see that the optimal costs are mostly the same if $\delta_{\text{init}}$ in equation (54) is small, or otherwise stated if the initialization $\boldsymbol{x}_0$ is close to the eigenvector initialization $\boldsymbol{\xi}_{\boldsymbol{A}}$ where we simulate good initializations. As $\delta_{\text{init}}$ grows where we simulate bad initializations, the differences grow as different PGS methods and proxy step-size strategies can lead to the convergence to different local minima. This phenomenon is extremely clear for the $\ell_1$ norm regularized instances, where $\boldsymbol{x}$ is valued mostly below 0.5 (see the example in Fig. 10) and $\delta_{\text{init}} > 0.5$ sets $\boldsymbol{x}_0$ almost to random.

**Optimality of the optimal costs.** We evaluate the optimal cost obtained from the initialization $\boldsymbol{x}_0$ by comparing with the optimal cost obtained from the eigenvector initialization

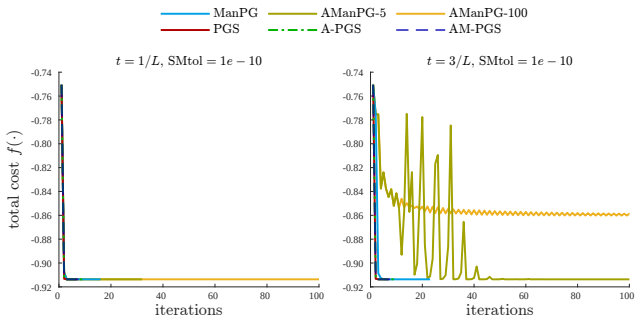



Fig. 8: The convergence of ManPG and AManPG using the line-search criterion (16) in comparison to the proposed one.

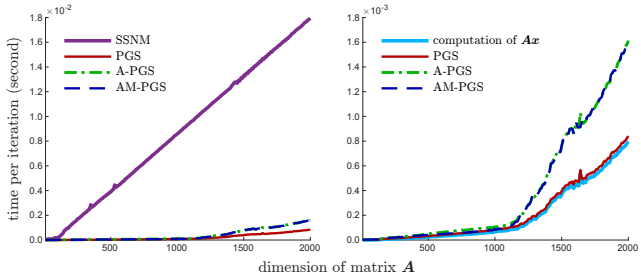



Fig. 9: The computational time per iteration by using the SSNM method [32] (used in [29], [30], [40]) and the proposed proxy step-size technique (used in the PGS, A-PGS and AM-PGS methods). The matrix-vector multiplication $\boldsymbol{Ax}$ is computed by the level-2 BLAS routine "dsymv".

$\boldsymbol{\xi}_{\boldsymbol{A}}$. For each Monte-Carlo run, we define the optimality:

$$\text{optimality} = \frac{\text{optimal cost initialized from } \boldsymbol{x}_0}{\text{optimal cost initialized from } \boldsymbol{\xi}_{\boldsymbol{A}}},$$

to benchmark the influence of different initializations. If the optimality metric is close to 1, then the optimal cost is close to the one obtained from the eigenvector initialization, and otherwise if this metric deviates from 1 then the computed solution is considered to be suboptimal. Intuitively, the optimality curve defines the robustness against bad initializations. The statistics for each tested case are plotted in Fig. 7. It is worth noting that the costs of the $\ell_1$ norm regularized instances are negative (as $\boldsymbol{A} = -\boldsymbol{M}$), thus the optimality metric is below 1. For the problem instances used in this paper, the PGS, A-PGS and AM-PGS methods are in general robust to a large range of initializations, while the $\ell_1$ norm regularized instances are more sensitive to initializations.

### 7.2 Comparison with ManPG [29] and AManPG [40]

We compare with the ManPG method [29] and its accelerated version AManPG [40]. Due to the difficulties of implementing the generalized Clarke differential for nuclear norm regularization, the comparison is only performed by applying $\ell_1$ regularization to problem (39). We use the C++ implementation of ManPG and AManPG released in [40]. AManPG uses a safeguard mechanism every $N$ iterations, where we set $N = 5$ and $N = 100$ and term the resulting methods as AManPG-5 and AManPG-100. The error-tolerance of the SSNM method [32] is set to $1e-10$.

We implemented our methods PGS, A-PGS, and AM-PGS in C++ as well for a fair comparison.

| | $e_{dist}$ (pixels)↓ | | | | | $e_{rep}$ (pixels)↓ | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 8pt | $PGS_5$ | $PGS_{10}$ | PGS | Gp | 8pt | $PGS_5$ | $PGS_{10}$ | PGS | Gp |
| Chapel(0,1) | 0.386 | 0.380 | *0.380* | 0.379 | **0.376** | 0.260 | 0.256 | *0.256* | 0.255 | **0.254** |
| Keble(0,3) | 0.248 | 0.247 | *0.247* | 0.247 | **0.247** | 0.175 | 0.175 | *0.175* | 0.175 | **0.175** |
| Desktop(C,D) | 0.574 | 0.336 | *0.303* | 0.288 | **0.268** | 0.406 | 0.238 | *0.214* | 0.204 | **0.190** |
| Library(1,3) | 0.428 | 0.409 | *0.405* | 0.405 | **0.400** | 0.302 | 0.289 | *0.286* | 0.286 | **0.282** |
| Merton1(1,3) | 0.308 | 0.295 | *0.291* | 0.288 | **0.277** | 0.217 | 0.209 | *0.205* | 0.203 | **0.196** |
| Merton2(1,3) | 0.596 | 0.528 | *0.498* | 0.472 | **0.404** | 0.421 | 0.373 | *0.352* | 0.334 | **0.286** |
| Arch | 0.304 | 0.299 | *0.299* | 0.299 | **0.298** | 0.215 | 0.211 | *0.211* | 0.211 | **0.211** |
| Yard | 0.433 | 0.429 | *0.429* | 0.428 | **0.426** | 0.306 | 0.303 | *0.303* | 0.302 | **0.301** |
| Slate | 0.246 | 0.184 | *0.177* | 0.170 | **0.163** | 0.129 | 0.097 | *0.093* | 0.089 | **0.085** |
| Ben1 | 0.203 | 0.144 | *0.128* | 0.102 | **0.101** | 0.142 | 0.101 | *0.089* | 0.071 | **0.070** |
| Ben2 | 0.139 | 0.086 | *0.063* | 0.050 | **0.048** | 0.097 | 0.060 | *0.044* | 0.035 | **0.033** |

TABLE 1: Fundamental matrix estimation with nuclear norm regularization (with data in [52]).

#### 7.2.1 Line-search Criteria and Convergence

**Theoretical justification.** In ManPG's line-search criterion (16), we first need to assign $t$ and then perform line-search for $\alpha_k$ to ensure the descent of the total cost $f(\cdot)$. From Theorem 2 of our work, we see that if $t \leq 1/L$ with $L$ being the Lipschitz constant, it suffices to set $\alpha_k = 1$ in the ManPG's line-search criterion (16). The authors in [29], [40] assume known Lipschitz constant $L$ and suggest to use $t = 1/L$ as a reference. Although for arbitrary $t > 0$, the existence of $\alpha_k$ for criterion (16) is proved in [29], we observe a proper $t$ in ManPG/AManPG is required.

**Numerical validation.** On the left of Fig. 8, we see by setting $t = 1/L$, ManPG's line-search (16) works almost the same way as the proposed PGS line-search. On the right of Fig. 8, we run the same numerical instance again by setting $t = 3/L$ in ManPG and AManPG. In this case, ManPG converges slower as seen from its curve being slightly shifted right. With some fluctuations, the accelerated method AManPG-5 manages to converge while the convergence is even slower than ManPG. AManPG-10 simply does not converge.

In practice, if the Lipschitz constant $L$ is unknown, it is expected that an approximated $L$ may cause a lot of trouble in ManPG's line-search criterion (16) as used in [29], [40]. In the proposed line-search, this is never a problem. Intuitively, we fix $\alpha_k = 1$ in criterion (16) and use proxy step-size $t'$ to find a working $t$ from the line-search criterion (25). This process is well-defined by the Lipschitz type assumption (*i.e.,* Assumption 1), and ensures the descent of the total cost $f(\cdot)$ by Theorem 2.

#### 7.2.2 Computational Complexity

The proposed PGS methods are much faster than ManPG and AManPG. Both ManPG and AManPG rely on the SSNM method [32] to solve the non-smooth KKT system, thus we compare the computation time per iteration of the proposed proxy step-size technique with that of the SSNM method. We report the timing statistics per iteration in Fig. 9, with respect to the dimension of the $\boldsymbol{A}$ matrix in problem (39). It is clearly seen that the proposed proxy step-size technique is substantially faster than the SSNM method. An ablation study shows that the computation time per iteration of the PGS, A-PGS and AM-PGS methods is mostly decided by the matrix-vector multiplication $\boldsymbol{A}\boldsymbol{x}_k$ used in evaluating the Euclidean gradient $\nabla g|_{\boldsymbol{x}} = 2\boldsymbol{A}\boldsymbol{x}_k$ and the cost function $g(\boldsymbol{x}_k) = \boldsymbol{x}_k^{\mathsf{T}}\boldsymbol{A}\boldsymbol{x}_k$. The A-PGS and AM-PGS methods have the same per iteration complexity, while being twice more expensive than the PGS method due to an extra evaluation at the auxiliary state $\boldsymbol{y}_k$.

### 7.3 Fundamental Matrix Estimation

We find $\lambda = 0.01$ works well in general, after normalizing the image points [3]. For this problem, we find that the PGS algorithm converges mostly within 20 iterations. For most of the cases, 10 iterations or even 5 are sufficient to reduce the last singular value $\sigma_3$ close enough to zero. Therefore aside from the PGS method (with full convergence), aiming for an efficient engineering design, we propose the following two truncated PGS algorithms:

- $PGS_5$. 5 PGS iterations, followed by a rank-2 rounding by setting $\sigma_3 = 0$.
- $PGS_{10}$. 10 PGS iterations, followed by a rank-2 rounding by setting $\sigma_3 = 0$.

We use two benchmark algorithms: a) the normalized eight point algorithm (denoted as 8pt), b) the global polynomial optimization [52] (denoted as Gp) with a formulation based on $\det(\boldsymbol{F}) = 0$.

We run the 8pt, $PGS_5$, $PGS_{10}$, PGS and Gp methods on a list of standard benchmarks, and report in Table 1 a) $e_{dist}$ the distance between the epipolar line and the corresponding image feature point [53], and b) $e_{rep}$ the reprojection error of the triangulated 3D points. Overall, the $e_{dist}$ and $e_{rep}$ statistics decrease consistently over the 8pt, $PGS_5$, $PGS_{10}$, PGS, and Gp methods. Table 1 shows that the $PGS_5$ and $PGS_{10}$ methods, with a close performance towards the PGS method, consistently outperform the 8pt method, and they can give almost the same accuracy as the global method Gp.

### 7.4 Correspondence Association

The regularization strength is set as $\lambda = -\sigma_{\min}(\boldsymbol{A})/(\sqrt{n}-1) = \sigma_{\max}(\boldsymbol{M})/(\sqrt{n}-1)$ with $n$ the dimension of $\boldsymbol{A}$. This choice is motivated by the canonical basis vector $\boldsymbol{e}_k^{\mathsf{T}}$. Since the diagonal elements of $\boldsymbol{A}$ are zero by construction, the total cost at $\boldsymbol{e}_k$ is $f(\boldsymbol{e}_k) = \boldsymbol{e}_k\boldsymbol{A}\boldsymbol{e}_k^{\mathsf{T}} + \lambda\|\boldsymbol{e}_k\|_{\ell_1} = \lambda$. Noting that $1 \leq \|\boldsymbol{x}\|_{\ell_1} \leq \sqrt{n}$, we thus have the following relation:

$$f(\boldsymbol{x}) < f(\boldsymbol{e}_k) \Leftrightarrow \boldsymbol{x}^{\mathsf{T}}\boldsymbol{A}\boldsymbol{x} + \lambda\|\boldsymbol{x}\|_{\ell_1} \leq \boldsymbol{x}^{\mathsf{T}}\boldsymbol{A}\boldsymbol{x} + \lambda\sqrt{n} < \lambda$$
$$\Leftrightarrow \lambda \leq -\frac{\boldsymbol{x}^{\mathsf{T}}\boldsymbol{A}\boldsymbol{x}}{\sqrt{n}-1} \leq -\frac{\sigma_{\min}(\boldsymbol{A})}{\sqrt{n}-1},$$

where the used $\lambda$ is chosen as the largest possible value.

Our experimental setup is similar to the one used in Section 5.1 of [5]. We simulate a 2-dimensional point-cloud

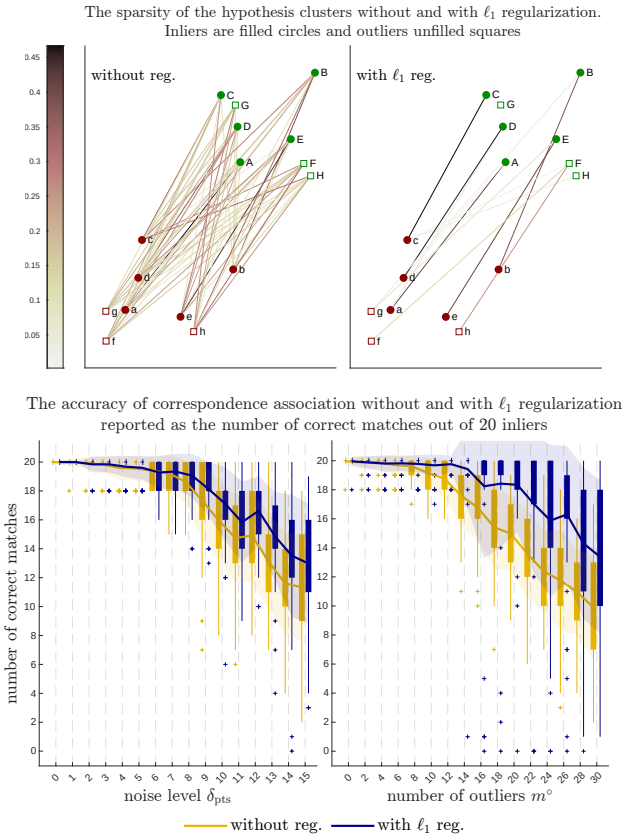



Fig. 10: Correspondence association with $\ell_1$ norm regularization. With $\ell_1$ regularization, high probability hypotheses are enhanced while low probability hypotheses are trimmed off, resulting in a sparser and more consistent cluster of association hypotheses. Therefore the correct correspondences (*i.e.,* inliers) can be identified more robustly.

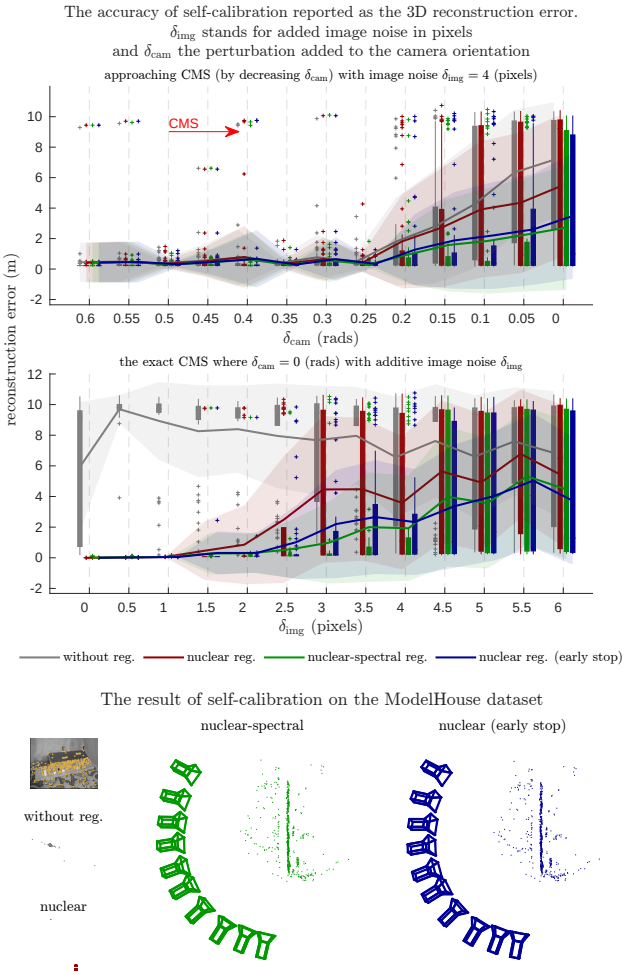



Fig. 11: Self-calibration with the nuclear norm and the nuclear-spectral norm regularization. We show the results with the critical motion sequence (CMS) where the camera principal axes pass approximately through the geometric center of the observed object (case R4 of Table 1 in [51]).

$\mathcal{Q}$ of $m$ points, with $x-y$ coordinates uniformly distributed in $[0,\, 256\sqrt{m/10}]$. Then we add zero-mean Gaussian noise with standard deviation $\delta_{\text{pts}}$ to each point in $\mathcal{Q}$, and rotate and translate the whole $\mathcal{Q}$ to obtain $\mathcal{Q}'$. We generate $m^\circ$ outliers in both $\mathcal{Q}$ and $\mathcal{Q}'$ uniformly in the same region. We use $\delta_d = 5$ in equation (44) as in [5].

The results are reported in Fig. 10. In the first figure, we give an illustration (using $m = 5$) that with $\ell_1$ norm regularization, many unlikely hypotheses are trimmed off, thus yielding a sparse hypothesis cluster. In the second figure, we set $m = 20$ and report the number of correct matches with respect to different noise levels and with respect to different outlier ratios. For each case, we use a 50 run Monte-Carlo simulation. It is clear that the association accuracy is consistently improved by using $\ell_1$ norm regularization.

### 7.5 Linear Self-calibration

#### 7.5.1 Implementation

We find normalization is essential to obtain stable self-calibration results. The key points of our implementation are sketched as follows:

1) Image point normalization. Let $(p_x, p_y)$ be the principal point of the camera. If this is unknown, we approximate $(p_x, p_y) = (I_x/2, I_y/2)$, where $I_x$ and $I_y$ are the width and height of the image. The average distance of all image points to $(p_x, p_y)$ is denoted by $s$. We normalize all image points by a common transformation $\boldsymbol{T}^{-1}$:

$$\boldsymbol{T} = \begin{pmatrix} s & 0 & p_x \\ 0 & s & p_y \\ 0 & 0 & 1 \end{pmatrix}, \; \boldsymbol{T}^{-1} = \begin{pmatrix} 1/s & 0 & -p_x/s \\ 0 & 1/s & -p_y/s \\ 0 & 0 & 1 \end{pmatrix}.$$

2) Projective reconstruction using projective bundle-adjustment from the normalized image points.
3) Quasi-Euclidean rectification [54]. We approximate the intrinsic matrix of each camera computed in step 2) as $\boldsymbol{K}_i = \text{diag}(f_i, f_i, 1)$, with $f_i = 2\sqrt{m_x^2 + m_y^2}$ where $m_x$ and $m_y$ are the maximum range in the $x$- and $y$- coordinates. Using this approximate $\boldsymbol{K}_i$, we compute an approximate estimate of $\boldsymbol{Q}_\infty^*$ from the DAQ constraint (48) *i.e.,* $\tilde{\boldsymbol{P}}_i \boldsymbol{Q}_\infty^* \tilde{\boldsymbol{P}}_i^\mathsf{T} \propto \boldsymbol{K}_i \boldsymbol{K}_i^\mathsf{T}$, and rectify the projective reconstruction approximately.
4) Linear self-calibration from the quasi-Euclidean rectification in step 3). We initialize the PGS methods from the canonical DAQ $\Omega_\infty^* = \text{diag}(1, 1, 1, 0)$.

5) Transforming cameras $\boldsymbol{P}_i$ obtained from step $4$) by $\boldsymbol{T}$. Lastly, $\boldsymbol{T}\boldsymbol{P}_i$ are the final estimate of Euclidean cameras.

We use $\lambda = 0.01$ for the nuclear norm regularization, and $\lambda_1 = 0.01$, $\lambda_2 = 2\lambda_1$ for the nuclear-spectral norm regularization.

#### 7.5.2 Simulated Data

We simulate a scene comprising: a) $50$ points spreading randomly in a diameter of $3$ meters; b) $7$ cameras circularly distributed $30$ meters away from the point-cloud. All cameras are oriented towards the centroid of the point-cloud, thus the simulated scenario is an artificial CMS whose ambiguity can be removed using the rank deficiency of the DAQ. We evaluate the performance of each method with respect to the perturbation of camera orientations $\delta_{\text{cam}}$ and the noise of image points $\delta_{\text{img}}$. We use the 3D reconstruction error as the evaluation metric, which is computed by the similarity Procrustes analysis between the ground-truth point-cloud and the estimated point-cloud.

We conduct two sets of experiments. First, we use the fixed image noise $\delta_{\text{img}} = 4$ pixels and decrease $\delta_{\text{cam}}$ to gradually bring the camera configuration to the CMS. Second, we set the camera configuration to the exact CMS where $\delta_{\text{cam}} = 0$ and then test the performance with respect to different image noise $\delta_{\text{img}}$. We report the reconstruction error with a $50$ run Monte-Carlo simulation in Fig. 11.

As shown in Fig. 11, when facing the CMS, the classical method without regularization fails, and the method with nuclear-spectral norm regularization is more robust than the one with nuclear norm regularization *e.g.*, in case of the exact CMS where $\delta_{\text{cam}} = 0$ and $\delta_{\text{img}} > 3$. It is interesting to see that the nuclear-norm regularization works well for less noisy scenarios of the CMS, *e.g.*, when $\delta_{\text{img}} < 2$. For these cases, it seems that the solution of the nuclear-norm regularized problem is well-trapped at the local minimum (the rank-3 DAQ), while the gradient is not large enough to go to the global minimum (the rank-1 DAQ). To examine this hypothesis, we use an early-stop trick (by setting the maximum PGS iterations to $1000$) in the nuclear norm regularization, and observe that with the early-stop trick the nuclear-norm regularized method mostly performs well.

#### 7.5.3 Real Data

A qualitative example of the CMS is given in Fig. 11 using a real dataset called ModelHouse where all cameras look towards the geometric center of a model house[1]. In this scenario, matrix $\boldsymbol{A}$ has two eigenvalues close to zero. The classic method fails because the solution space is ambiguous. The nuclear norm regularized method fails by converging to the rank-1 solution. The nuclear-spectral norm regularized method can find the correct rank-3 solution thus recover the correct Euclidean geometry. The nuclear norm regularized method with the early stop trick also works. Intuitively, the spectral norm in the nuclear-spectral norm regularization prevents the algorithm from gliding to the rank-1 solution, and the early stop trick has the similar functionality.

1. https://www.robots.ox.ac.uk/~vgg/data/mview/

## 8 Conclusion

We have proposed the proxy step-size technique, and presented an effective solution to problem (2) for convex and absolutely homogeneous $h(\cdot)$. The proposed solution is: exact (satisfying the first-order necessary optimality condition), elegant (simple and in closed-form), and easily applicable (to nuclear norm regularization etc.). Future work includes extending the proxy-step size technique to the oblique and the Stiefel manifolds, and analyzing the convergence rate of the accelerated methods in Algorithm 3.

## References


[1] O. Faugeras and B. Mourrain, "On the geometry and algebra of the point and line correspondences between $n$ images," in *International Conference on Computer Vision*, 1995.

[2] B. Triggs, "Autocalibration and the absolute quadric," in *Computer Vision and Pattern Recognition*, 1997.

[3] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, ISBN: 0521540518, 2004.

[4] A. Etz, "Introduction to the concept of likelihood and its applications," *Advances in Methods and Practices in Psychological Science*, vol. 1, no. 1, pp. 60–69, 2018.

[5] M. Leordeanu and M. Hebert, "A spectral technique for correspondence problems using pairwise constraints," in *Computer Vision, IEEE International Conference on*, vol. 2. IEEE Computer Society, 2005, pp. 1482–1489.

[6] A. M. Peter and A. Rangarajan, "Maximum likelihood wavelet density estimation with applications to image and shape matching," *IEEE Transactions on Image Processing*, vol. 17, no. 4, pp. 458–468, 2008.

[7] R. Lai, Z. Wen, W. Yin, X. Gu, and L. M. Lui, "Folding-free global conformal mapping for genus-0 surfaces by harmonic energy minimization," *Journal of Scientific Computing*, vol. 58, no. 3, pp. 705–725, 2014.

[8] J. Hu, X. Liu, Z.-W. Wen, and Y.-X. Yuan, "A brief introduction to manifold optimization," *Journal of the Operations Research Society of China*, vol. 8, no. 2, pp. 199–248, 2020.

[9] M. Fazel, "Matrix rank minimization with applications," Ph.D. dissertation, Stanford University, 2002.

[10] Y. Nesterov, *Introductory lectures on convex optimization: A basic course*. Springer Science & Business Media, 2003, vol. 87.

[11] A. Beck, *First-order methods in optimization*. SIAM, 2017.

[12] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM Journal on Imaging Sciences*, vol. 2, no. 1, pp. 183–202, 2009.

[13] M. Fazel, T. K. Pong, D. Sun, and P. Tseng, "Hankel matrix rank minimization with applications to system identification and realization," *SIAM Journal on Matrix Analysis and Applications*, vol. 34, no. 3, pp. 946–977, 2013.

[14] Y. Nesterov, "Gradient methods for minimizing composite functions," *Mathematical Programming*, vol. 140, no. 1, pp. 125–161, 2013.

[15] P.-A. Absil, R. Mahony, and R. Sepulchre, *Optimization algorithms on matrix manifolds*. Princeton University Press, 2009.

[16] N. Boumal, "An introduction to optimization on smooth manifolds," *Available online, May*, 2020.

[17] P. Grohs and S. Hosseini, "$\varepsilon$-subgradient algorithms for locally Lipschitz functions on Riemannian manifolds," *Advances in Computational Mathematics*, vol. 42, no. 2, pp. 333–360, 2016.

[18] ——, "Nonsmooth trust region algorithms for locally Lipschitz functions on Riemannian manifolds," *IMA Journal of Numerical Analysis*, vol. 36, no. 3, pp. 1167–1192, 2016.

[19] S. Hosseini and A. Uschmajew, "A Riemannian gradient sampling algorithm for nonsmooth optimization on manifolds," *SIAM Journal on Optimization*, vol. 27, no. 1, pp. 173–189, 2017.

[20] S. Hosseini, W. Huang, and R. Yousefpour, "Line search algorithms for locally Lipschitz functions on Riemannian manifolds," *SIAM Journal on Optimization*, vol. 28, no. 1, pp. 596–619, 2018.

[21] O. Ferreira and P. Oliveira, "Proximal point algorithm on Riemannian manifolds," *Optimization*, vol. 51, no. 2, pp. 257–270, 2002.

[22] G. de Carvalho Bento, J. X. da Cruz Neto, and P. R. Oliveira, "A new approach to the proximal point method: convergence on general Riemannian manifolds," *Journal of Optimization Theory and Applications*, vol. 168, no. 3, pp. 743–755, 2016.

[23] G. C. Bento, O. P. Ferreira, and J. G. Melo, "Iteration-complexity of gradient, subgradient and proximal point methods on Riemannian manifolds," *Journal of Optimization Theory and Applications*, vol. 173, no. 2, pp. 548–562, 2017.

[24] R. Lai and S. Osher, "A splitting method for orthogonality constrained problems," *Journal of Scientific Computing*, vol. 58, no. 2, pp. 431–449, 2014.

[25] A. Kovnatsky, K. Glashoff, and M. M. Bronstein, "MADMM: a generic algorithm for non-smooth optimization on manifolds," in *European Conference on Computer Vision*. Springer, 2016, pp. 680–696.

[26] W. Chen, H. Ji, and Y. You, "An augmented Lagrangian method for $\ell_1$-regularized optimization problems with orthogonality constraints," *SIAM Journal on Scientific Computing*, vol. 38, no. 4, pp. B570–B592, 2016.

[27] H. Zhu, X. Zhang, D. Chu, and L.-Z. Liao, "Nonconvex and nonsmooth optimization with generalized orthogonality constraints: An approximate augmented Lagrangian method," *Journal of Scientific Computing*, vol. 72, no. 1, pp. 331–372, 2017.

[28] Y. Wang, W. Yin, and J. Zeng, "Global convergence of ADMM in nonconvex nonsmooth optimization," *Journal of Scientific Computing*, vol. 78, no. 1, pp. 29–63, 2019.

[29] S. Chen, S. Ma, A. Man-Cho So, and T. Zhang, "Proximal gradient method for nonsmooth optimization over the Stiefel manifold," *SIAM Journal on Optimization*, vol. 30, no. 1, pp. 210–239, 2020.

[30] W. Huang and K. Wei, "Riemannian proximal gradient methods," *Mathematical Programming*, pp. 1–43, 2021.

[31] M. Tan, Z. Hu, Y. Yan, J. Cao, D. Gong, and Q. Wu, "Learning sparse pca with stabilized admm method on stiefel manifold," *IEEE Transactions on Knowledge and Data Engineering*, vol. 33, no. 3, pp. 1078–1088, 2019.

[32] X. Xiao, Y. Li, Z. Wen, and L. Zhang, "A regularized semi-smooth Newton method with projection steps for composite convex programs," *Journal of Scientific Computing*, vol. 76, no. 1, pp. 364–389, 2018.

[33] Y. E. Nesterov, "A method for solving the convex programming problem with convergence rate $o(1/k^2)$," *Dokl. akad. nauk Sssr*, vol. 269, pp. 543–547, 1983.

[34] G. Dirr, U. Helmke, and C. Lageman, "Nonsmooth Riemannian optimization with applications to sphere packing and grasping," in *Lagrangian and Hamiltonian methods for nonlinear control 2006*. Springer, 2007, pp. 29–45.

[35] P. B. Borckmans, S. E. Selvan, N. Boumal, and P.-A. Absil, "A Riemannian subgradient algorithm for economic dispatch with valve-point effect," *Journal of computational and applied mathematics*, vol. 255, pp. 848–866, 2014.

[36] A. Goldstein, "Optimization of Lipschitz continuous functions," *Mathematical Programming*, vol. 13, no. 1, pp. 14–22, 1977.

[37] H. Zhang and S. Sra, "First-order methods for geodesically convex optimization," in *Conference on Learning Theory*. PMLR, 2016, pp. 1617–1638.

[38] R. L. Bishop and B. O'Neill, "Manifolds of negative curvature," *Transactions of the American Mathematical Society*, vol. 145, pp. 1–49, 1969.

[39] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Machine Learning*, vol. 3, no. 1, pp. 1–122, 2010.

[40] W. Huang and K. Wei, "An extension of fast iterative shrinkage-thresholding algorithm to riemannian optimization for sparse principal component analysis," *Numerical Linear Algebra with Applications*, p. e2409, 2021.

[41] G. A. Watson, "Characterization of the subdifferential of some matrix norms," *Linear algebra and its applications*, vol. 170, no. 0, pp. 33–45, 1992.

[42] N. Boumal, P.-A. Absil, and C. Cartis, "Global rates of convergence for nonconvex optimization on manifolds," *IMA Journal of Numerical Analysis*, vol. 39, no. 1, pp. 1–33, 2019.

[43] W. H. Yang, L.-H. Zhang, and R. Song, "Optimality conditions for the nonlinear programming problems on Riemannian manifolds," *Pacific Journal of Optimization*, vol. 10, no. 2, pp. 415–434, 2014.

[44] Y. Liu, F. Shang, J. Cheng, H. Cheng, and L. Jiao, "Accelerated first-order methods for geodesically convex optimization on Riemannian manifolds." in *NIPS*, 2017, pp. 4868–4877.

[45] H. Zhang and S. Sra, "An estimate sequence for geodesically convex optimization," in *Conference On Learning Theory*. PMLR, 2018, pp. 1703–1723.

[46] A. Beck and M. Teboulle, "Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems," *IEEE transactions on image processing*, vol. 18, no. 11, pp. 2419–2434, 2009.

[47] Y. Zheng, S. Sugimoto, and M. Okutomi, "A practical rank-constrained eight-point algorithm for fundamental matrix estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 1546–1553.

[48] J. Zhao, "An efficient solution to non-minimal case essential matrix estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[49] N. Parikh and S. Boyd, "Proximal algorithms," *Foundations and Trends in optimization*, vol. 1, no. 3, pp. 127–239, 2014.

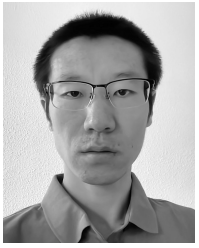
[50] M. Pollefeys, R. Koch, and L. Van Gool, "Self-calibration and metric reconstruction inspite of varying and unknown intrinsic camera parameters," *International Journal of Computer Vision*, vol. 32, no. 1, pp. 7–25, 1999.

[51] P. Gurdjos, A. Bartoli, and P. Sturm, "Is dual linear self-calibration artificially ambiguous?" in *International Conference on Computer Vision*, 2009.

[52] F. Bugarin, A. Bartoli, D. Henrion, J.-B. Lasserre, J.-J. Orteu, and T. Sentenac, "Rank-constrained fundamental matrix estimation by polynomial global optimization versus the eight-point algorithm," *Journal of Mathematical Imaging and Vision*, vol. 53, no. 1, pp. 42–60, 2015.

[53] Z. Zhang, "Determining the epipolar geometry and its uncertainty: A review," *International Journal of Computer Vision*, vol. 27, no. 2, pp. 161–195, 1998.

[54] P. A. Beardsley, A. Zisserman, and D. W. Murray, "Sequential updating of projective and affine structure from motion," *International journal of computer vision*, vol. 23, no. 3, pp. 235–259, 1997.



**Fang Bai** Fang Bai was born in Ningxia Province, China, in 1988. He received the B.Sc. degree in computer science and technology from Nankai University, China, in 2010, and the Ph.D. degree in robotics from University of Technology Sydney, Australia, in 2020. His research has been focused on mathematical abstractions in robotics and computer vision. He has conducted several fundamental breakthroughs on related topics as the first-author, e.g., the cycle based pose graph optimization, the equation to predict the change of optimal values, and the closed-form solution for template-free deformable Procrustes analysis.

**Adrien Bartoli** Adrien Bartoli has held the position of Professor of Computer Science at Université Clermont Auvergne since fall 2009 and has been a member of Institut Universitaire de France since 2016. He is currently on leave as research scientist at the University Hospital of Clermont-Ferrand and as Chief Scientific Officer at SurgAR. He leads the Endoscopy and Computer Vision (EnCoV) research group at the University and Hospital of Clermont-Ferrand. His main research interests are in computer vision, including image registration and Shape-from-X for deformable environments, and their application to computer-aided medical interventions.

# APPENDIX A
# PROPERTIES OF PROXIMAL

## A.1 Convexity

If $h(\cdot)$ is convex, then for any $\boldsymbol{x}$ and $\boldsymbol{x}_0$, we have:

$$h(\boldsymbol{x}) \geq h(\boldsymbol{x}_0) + \langle \partial h|_{\boldsymbol{x}_0},\, \boldsymbol{x} - \boldsymbol{x}_0 \rangle.$$

## A.2 Firm non-expansiveness and non-expansiveness

For a convex $h(\cdot)$, let $\boldsymbol{z}_1 = \operatorname{prox}_{th}(\boldsymbol{w}_1)$ and $\boldsymbol{z}_2 = \operatorname{prox}_{th}(\boldsymbol{w}_2)$ with $t \geq 0$. Then the following holds:

- Firm non-expansiveness:

$$\|\boldsymbol{z}_1 - \boldsymbol{z}_2\|_2^2 \leq \langle \boldsymbol{z}_1 - \boldsymbol{z}_2,\, \boldsymbol{w}_1 - \boldsymbol{w}_2 \rangle; \tag{55}$$

- Non-expansiveness:

$$\|\boldsymbol{z}_1 - \boldsymbol{z}_2\|_2^2 \leq \|\boldsymbol{w}_1 - \boldsymbol{w}_2\|_2^2. \tag{56}$$

*Proof.* The solution of the proximal is characterized by its first-order necessary condition:

$$\mathbf{0} \in \partial h|_{\boldsymbol{z}_1} + \frac{1}{t}(\boldsymbol{z}_1 - \boldsymbol{w}_1) \Leftrightarrow \frac{1}{t}(\boldsymbol{w}_1 - \boldsymbol{z}_1) \in \partial h|_{\boldsymbol{z}_1}, \tag{57}$$

$$\mathbf{0} \in \partial h|_{\boldsymbol{z}_2} + \frac{1}{t}(\boldsymbol{z}_2 - \boldsymbol{w}_2) \Leftrightarrow \frac{1}{t}(\boldsymbol{w}_2 - \boldsymbol{z}_2) \in \partial h|_{\boldsymbol{z}_2}. \tag{58}$$

By the convexity of $h(\cdot)$, we have:

$$\begin{cases} h(\boldsymbol{z}_1) \geq h(\boldsymbol{z}_2) + \langle \partial h|_{\boldsymbol{z}_2},\, \boldsymbol{z}_1 - \boldsymbol{z}_2 \rangle & \text{(59a)} \\ h(\boldsymbol{z}_2) \geq h(\boldsymbol{z}_1) + \langle \partial h|_{\boldsymbol{z}_1},\, \boldsymbol{z}_2 - \boldsymbol{z}_1 \rangle. & \text{(59b)} \end{cases}$$

Substituting equations (57) and (58) into (59), we have:

$$\begin{cases} h(\boldsymbol{z}_1) \geq h(\boldsymbol{z}_2) + \langle \frac{1}{t}(\boldsymbol{w}_2 - \boldsymbol{z}_2),\, \boldsymbol{z}_1 - \boldsymbol{z}_2 \rangle & \text{(60a)} \\ h(\boldsymbol{z}_2) \geq h(\boldsymbol{z}_1) + \langle \frac{1}{t}(\boldsymbol{w}_1 - \boldsymbol{z}_1),\, \boldsymbol{z}_2 - \boldsymbol{z}_1 \rangle. & \text{(60b)} \end{cases}$$

Summing together inequalities (60a) and (60b), we have:

$$\langle \boldsymbol{w}_2 - \boldsymbol{z}_2 - \boldsymbol{w}_1 + \boldsymbol{z}_1,\, \boldsymbol{z}_1 - \boldsymbol{z}_2 \rangle \leq 0.$$

Expanding the above, we obtain the firm non-expansiveness (55). By the Cauchy–Schwarz inequality, we obtain:

$$|\langle \boldsymbol{z}_1 - \boldsymbol{z}_2,\, \boldsymbol{w}_1 - \boldsymbol{w}_2 \rangle| \leq \|\boldsymbol{z}_1 - \boldsymbol{z}_2\|_2 \|\boldsymbol{w}_1 - \boldsymbol{w}_2\|_2.$$

After canceling $\|\boldsymbol{z}_1 - \boldsymbol{z}_2\|_2$ in the firm non-expansiveness (55), we obtain non-expansiveness (56).

□

## A.3 Proof of Lemma 2

By definition of the proximal, we write:

$$\operatorname{prox}_{th}(\mathbf{0}) = \arg\min_{\boldsymbol{x}} \left\{ th(\boldsymbol{x}) + \frac{1}{2}\|\boldsymbol{x}\|^2 \right\}, \quad t \geq 0 \tag{61}$$

Since $h(\cdot)$ is convex and absolutely homogeneous, from Lemma 1, we have $h(\boldsymbol{x}) \geq 0$ and $h(\mathbf{0}) = 0$. Thus, we observe that the optimal cost of problem (61) is 0 which is attained at $\boldsymbol{x} = \mathbf{0}$ as $h(\mathbf{0}) = 0$ and $\|\boldsymbol{x}\| = 0$. This concludes $\operatorname{prox}_{th}(\mathbf{0}) = \mathbf{0}$ if $h(\cdot)$ is convex and absolutely homogeneous. The inequalities $\|\operatorname{prox}_{th}(\boldsymbol{w})\|_2^2 \leq \langle \operatorname{prox}_{th}(\boldsymbol{w}),\, \boldsymbol{w} \rangle$ and $\|\operatorname{prox}_{th}(\boldsymbol{w})\|_2 \leq \|\boldsymbol{w}\|_2$ follow from firm non-expansiveness (55) and non-expansiveness (56), by setting $\boldsymbol{w}_1 = \boldsymbol{w}$ and $\boldsymbol{w}_2 = \mathbf{0}$.

# APPENDIX B
# DERIVATION OF KKT SYSTEM (15)

We write $\mathbf{0} \in \partial \mathcal{L}_{\boldsymbol{v}_k}$ as follow:

$$\begin{aligned} &\mathbf{0} \in \operatorname{grad} g|_{\boldsymbol{x}_k} + \frac{1}{t}\boldsymbol{v}_k + \partial h|_{\boldsymbol{x}_k + \boldsymbol{v}_k} + \mu \boldsymbol{x}_k \\ &\Leftrightarrow (1 - \mu t)\, \boldsymbol{x}_k - t \operatorname{grad} g|_{\boldsymbol{x}_k} \in t \partial h|_{\boldsymbol{x}_k + \boldsymbol{v}_k} + \boldsymbol{x}_k + \boldsymbol{v}_k \\ &\Leftrightarrow \boldsymbol{x}_k + \boldsymbol{v}_k = \operatorname{prox}_{th}\left((1 - \mu t)\, \boldsymbol{x}_k - t \operatorname{grad} g|_{\boldsymbol{x}_k}\right). \end{aligned}$$

Therefore:

$$\boldsymbol{v}_k = \operatorname{prox}_{th}\left((1 - \mu t)\, \boldsymbol{x}_k - t \operatorname{grad} g|_{\boldsymbol{x}_k}\right) - \boldsymbol{x}_k,$$

which is equation (15a). Considering $\boldsymbol{x}_k^{\mathsf{T}} \boldsymbol{v}_k = 0$ with equation (15a), we obtain equation (15b).

# APPENDIX C
# PROOF OF LEMMA 3

By definition of the proximal, we write:

$$\operatorname{prox}_{th}(\alpha \boldsymbol{w}) = \arg\min_{\boldsymbol{x}} \left\{ h(\boldsymbol{x}) + \frac{1}{2t}\|\boldsymbol{x} - \alpha \boldsymbol{w}\|^2 \right\} \overset{\text{def}}{=} \boldsymbol{z}.$$

Since $h(\cdot)$ is absolutely homogeneous, $h(\boldsymbol{x}) = |\alpha|\, h\left(\frac{1}{\alpha}\boldsymbol{x}\right)$. The above equation can thus be written as:

$$\begin{aligned} \boldsymbol{z} &= \arg\min_{\boldsymbol{x}} \left\{ |\alpha|\, h\left(\frac{1}{\alpha}\boldsymbol{x}\right) + \frac{\alpha^2}{2t} \left\| \frac{1}{\alpha}\boldsymbol{x} - \boldsymbol{w} \right\|^2 \right\} \\ &= \arg\min_{\boldsymbol{x}} \left\{ h\left(\frac{1}{\alpha}\boldsymbol{x}\right) + \frac{1}{2\frac{t}{|\alpha|}} \left\| \frac{1}{\alpha}\boldsymbol{x} - \boldsymbol{w} \right\|^2 \right\} \\ &\Leftrightarrow \frac{1}{\alpha}\boldsymbol{z} = \operatorname{prox}_{\frac{t}{|\alpha|}h}(\boldsymbol{w}). \end{aligned}$$

Therefore $\boldsymbol{z} = \operatorname{prox}_{th}(\alpha \boldsymbol{w}) = \alpha \operatorname{prox}_{\frac{t}{|\alpha|}h}(\boldsymbol{w})$.

# APPENDIX D
# PROOF OF LEMMA 4

From equation (17b), we obtain:

$$\begin{aligned} \frac{1}{\phi(t')} = \frac{1}{t} &= \frac{1}{t'} \boldsymbol{x}_k^{\mathsf{T}} \operatorname{prox}_{|t'|h}\left( t' \left( \frac{1}{t'}\boldsymbol{x}_k - \operatorname{grad} g|_{\boldsymbol{x}_k} \right) \right) \\ &= \boldsymbol{x}_k^{\mathsf{T}} \operatorname{prox}_h\left( \frac{1}{t'}\boldsymbol{x}_k - \operatorname{grad} g|_{\boldsymbol{x}_k} \right). \end{aligned}$$

The last equality is due to Lemma 3. We denote $\boldsymbol{w}_1 = \frac{1}{t_1'}\boldsymbol{x}_k - \operatorname{grad} g|_{\boldsymbol{x}_k}$ and $\boldsymbol{w}_2 = \frac{1}{t_2'}\boldsymbol{x}_k - \operatorname{grad} g|_{\boldsymbol{x}_k}$. Then $1/t_1' = \boldsymbol{x}_k^{\mathsf{T}} \boldsymbol{w}_1$ and $1/t_2' = \boldsymbol{x}_k^{\mathsf{T}} \boldsymbol{w}_2$. Moreover:

$$\boldsymbol{x}_k \boldsymbol{x}_k^{\mathsf{T}} (\boldsymbol{w}_1 - \boldsymbol{w}_2) = \frac{1}{t_1'}\boldsymbol{x}_k - \frac{1}{t_2'}\boldsymbol{x}_k = \boldsymbol{w}_1 - \boldsymbol{w}_2. \tag{62}$$

The proof is given by summarizing the above facts as:

$$\begin{aligned} \epsilon(t_1', t_2') &= \langle \boldsymbol{x}_k^{\mathsf{T}} \operatorname{prox}_h(\boldsymbol{w}_1) - \boldsymbol{x}_k^{\mathsf{T}} \operatorname{prox}_h(\boldsymbol{w}_2),\, \boldsymbol{x}_k^{\mathsf{T}}(\boldsymbol{w}_1 - \boldsymbol{w}_2) \rangle \\ &= \langle \operatorname{prox}_h(\boldsymbol{w}_1) - \operatorname{prox}_h(\boldsymbol{w}_2),\, \boldsymbol{x}_k \boldsymbol{x}_k^{\mathsf{T}}(\boldsymbol{w}_1 - \boldsymbol{w}_2) \rangle \\ &= \langle \operatorname{prox}_h(\boldsymbol{w}_1) - \operatorname{prox}_h(\boldsymbol{w}_2),\, \boldsymbol{w}_1 - \boldsymbol{w}_2 \rangle \\ &\geq \|\operatorname{prox}_h(\boldsymbol{w}_1) - \operatorname{prox}_h(\boldsymbol{w}_2)\|_2^2, \end{aligned}$$

where the inequality is due to the firm non-expansiveness of $\operatorname{prox}_h(\cdot)$.

# APPENDIX E
# PROOF OF LEMMA 5

## E.1 A General Inequality by Convexity

**Lemma 7.** *Let $h(\cdot)$ be convex. Then for any $t$, $\boldsymbol{x}$ and $\boldsymbol{w}$, we have:*

$$\langle \boldsymbol{w} - \operatorname{prox}_{|t|h}(\boldsymbol{w}),\, \operatorname{prox}_{|t|h}(\boldsymbol{w}) - \boldsymbol{x}\rangle \geq |t| \left(h(\operatorname{prox}_{|t|h}(\boldsymbol{w})) - h(\boldsymbol{x})\right). \quad (63)$$

*Proof.* Let $\boldsymbol{z} = \operatorname{prox}_{|t|h}(\boldsymbol{w})$. By definition of the proximal, we write:

$$\boldsymbol{z} = \arg\min_{\boldsymbol{y}} \left\{ h(\boldsymbol{y}) + \frac{1}{2|t|}\|\boldsymbol{y} - \boldsymbol{w}\|^2 \right\}. \quad (64)$$

The first-order necessary condition of problem (64) states:

$$-\frac{1}{|t|}(\boldsymbol{z} - \boldsymbol{w}) \in \partial h|_{\boldsymbol{z}}. \quad (65)$$

By the convexity of $h(\cdot)$ at $\boldsymbol{z}$, the following inequality holds for any $\boldsymbol{x}$:

$$h(\boldsymbol{x}) \geq h(\boldsymbol{z}) + \langle \partial h|_{\boldsymbol{z}},\, \boldsymbol{x} - \boldsymbol{z}\rangle.$$

Therefore for any $\boldsymbol{x}$, we have the following inequality:

$$h(\boldsymbol{x}) \geq h(\boldsymbol{z}) - \frac{1}{|t|}\langle \boldsymbol{z} - \boldsymbol{w},\, \boldsymbol{x} - \boldsymbol{z}\rangle.$$

Reorganizing this inequality, we obtain inequality (63). □

## E.2 Proof of Lemma 5

Following Lemma 7, in inequality (63), we let $\boldsymbol{x} = \alpha \operatorname{prox}_{|t|h}(\boldsymbol{w})$:

$$\langle \boldsymbol{w} - \operatorname{prox}_{|t|h}(\boldsymbol{w}),\, (1-\alpha)\operatorname{prox}_{|t|h}(\boldsymbol{w})\rangle \geq |t|\,(1-|\alpha|)h(\operatorname{prox}_{|t|h}(\boldsymbol{w})). \quad (66)$$

If $0 \leq \alpha < 1$, then $1 - |\alpha| = 1 - \alpha > 0$:

$$\langle \boldsymbol{w} - \operatorname{prox}_{|t|h}(\boldsymbol{w}),\, \operatorname{prox}_{|t|h}(\boldsymbol{w})\rangle \geq |t|\, h(\operatorname{prox}_{|t|h}(\boldsymbol{w})). \quad (67)$$

If $\alpha > 1$, then $1 - |\alpha| = 1 - \alpha < 0$:

$$\langle \boldsymbol{w} - \operatorname{prox}_{|t|h}(\boldsymbol{w}),\, \operatorname{prox}_{|t|h}(\boldsymbol{w})\rangle \leq |t|\, h(\operatorname{prox}_{|t|h}(\boldsymbol{w})). \quad (68)$$

By considering inequalities (67) and (68) together, we obtain:

$$\langle \boldsymbol{w} - \operatorname{prox}_{|t|h}(\boldsymbol{w}),\, \operatorname{prox}_{|t|h}(\boldsymbol{w})\rangle = |t|\, h(\operatorname{prox}_{|t|h}(\boldsymbol{w})). \quad (69)$$

Subtracting equation (69) in inequality (63), we obtain inequality (21).

# APPENDIX F
# PROOF OF PROPOSITION 3

It can be shown that $t = t'/c(t')$ satisfies:

$$\begin{aligned}
|t| &= \frac{|t'|}{\left|\boldsymbol{x}_k^{\mathsf{T}} \operatorname{prox}_{|t'|h}(\boldsymbol{x}_k - t'\operatorname{grad} g|_{\boldsymbol{x}_k})\right|} \\
&\geq \frac{|t'|}{\|\boldsymbol{x}_k\|_2 \left\|\operatorname{prox}_{|t'|h}(\boldsymbol{x}_k - t'\operatorname{grad} g|_{\boldsymbol{x}_k})\right\|_2} \quad \text{(Cauchy–Schwarz)} \\
&\geq \frac{|t'|}{\|\boldsymbol{x}_k\|_2 \|\boldsymbol{x}_k - t'\operatorname{grad} g|_{\boldsymbol{x}_k}\|_2} \quad (\textbf{Lemma } 2) \\
&= \frac{|t'|}{\|\boldsymbol{x}_k\|_2 \sqrt{\|\boldsymbol{x}_k\|_2^2 + \|t'\operatorname{grad} g|_{\boldsymbol{x}_k}\|_2^2}} \quad (\text{since } \boldsymbol{x}_k^{\mathsf{T}}\operatorname{grad} g|_{\boldsymbol{x}_k} = 0) \\
&= \frac{1}{\sqrt{(1/t')^2 + \|\operatorname{grad} g|_{\boldsymbol{x}_k}\|_2^2}} \quad (\text{since } \|\boldsymbol{x}_k\|_2 = 1).
\end{aligned}$$

# APPENDIX G
# PROOF OF LEMMA 6

Since $\langle \boldsymbol{x}_k, \boldsymbol{v}_k\rangle = 0$ as $\boldsymbol{v}_k \in \mathcal{T}_{\boldsymbol{x}_k}\mathcal{S}$, the norm satisfies:

$$\begin{aligned}
\|\boldsymbol{x}_k + \boldsymbol{v}_k\|_2 &= \sqrt{\|\boldsymbol{x}_k\|_2^2 + \|\boldsymbol{v}_k\|_2^2 + 2\langle \boldsymbol{x}_k, \boldsymbol{v}_k\rangle} \\
&= \sqrt{\|\boldsymbol{x}_k\|_2^2 + \|\boldsymbol{v}_k\|_2^2} \geq \sqrt{\|\boldsymbol{x}_k\|_2^2} = 1.
\end{aligned}$$

By retraction (12), since $h(\cdot)$ is absolutely homogeneous, we write:

$$\begin{aligned}
h\left(\mathcal{R}_{\boldsymbol{x}_k}(\boldsymbol{v}_k)\right) &= h\left(\frac{\boldsymbol{x}_k + \boldsymbol{v}_k}{\|\boldsymbol{x}_k + \boldsymbol{v}_k\|_2}\right) \\
&= \frac{1}{\|\boldsymbol{x}_k + \boldsymbol{v}_k\|_2} h\left(\boldsymbol{x}_k + \boldsymbol{v}_k\right) \leq h\left(\boldsymbol{x}_k + \boldsymbol{v}_k\right).
\end{aligned}$$

# APPENDIX H
# LIPSCHITZ-TYPE CONSTANT $L$

It is well-known in the Rayleigh quotient literature that:

$$\max_{\boldsymbol{v}} \frac{\boldsymbol{v}^{\mathsf{T}}\boldsymbol{A}\boldsymbol{v}}{\boldsymbol{v}^{\mathsf{T}}\boldsymbol{v}} = \sigma_{\max}(\boldsymbol{A}),$$

where $\sigma_{\max}(\boldsymbol{A})$ is the largest singular value of $\boldsymbol{A}$. Therefore for any $\boldsymbol{v}$, we have:

$$\frac{\boldsymbol{v}^{\mathsf{T}}\boldsymbol{A}\boldsymbol{v}}{\boldsymbol{v}^{\mathsf{T}}\boldsymbol{v}} \leq \sigma_{\max}(\boldsymbol{A}) \Leftrightarrow \boldsymbol{v}^{\mathsf{T}}\boldsymbol{A}\boldsymbol{v} \leq \sigma_{\max}(\boldsymbol{A})\|\boldsymbol{v}\|_2^2. \quad (70)$$

From equation (38) and the fact that $\boldsymbol{x}^{\mathsf{T}}\boldsymbol{v} = 0$, we have:

$$\langle \operatorname{grad} g(\boldsymbol{x}),\, \boldsymbol{v}\rangle = \langle 2\boldsymbol{A}\boldsymbol{x} - 2(\boldsymbol{x}^{\mathsf{T}}\boldsymbol{A}\boldsymbol{x})\boldsymbol{x},\, \boldsymbol{v}\rangle = 2\boldsymbol{x}^{\mathsf{T}}\boldsymbol{A}\boldsymbol{v}. \quad (71)$$

Recall that $\frac{1}{\|\boldsymbol{x}+\boldsymbol{v}\|_2^2} = \frac{1}{1+\|\boldsymbol{v}\|_2^2} \leq 1$. For $g(\boldsymbol{x}) = \boldsymbol{x}^{\mathsf{T}}\boldsymbol{A}\boldsymbol{x}$ and retraction (12), we write

$$\begin{aligned}
g\left(\mathcal{R}_{\boldsymbol{x}}(\boldsymbol{v})\right) &= \left(\frac{\boldsymbol{x}+\boldsymbol{v}}{\|\boldsymbol{x}+\boldsymbol{v}\|_2}\right)^{\mathsf{T}} \boldsymbol{A} \left(\frac{\boldsymbol{x}+\boldsymbol{v}}{\|\boldsymbol{x}+\boldsymbol{v}\|_2}\right) \\
&= \frac{1}{\|\boldsymbol{x}+\boldsymbol{v}\|_2^2}\left(\boldsymbol{x}^{\mathsf{T}}\boldsymbol{A}\boldsymbol{x} + 2\boldsymbol{x}^{\mathsf{T}}\boldsymbol{A}\boldsymbol{v} + \boldsymbol{v}^{\mathsf{T}}\boldsymbol{A}\boldsymbol{v}\right) \\
&\leq \boldsymbol{x}^{\mathsf{T}}\boldsymbol{A}\boldsymbol{x} + 2\boldsymbol{x}^{\mathsf{T}}\boldsymbol{A}\boldsymbol{v} + \boldsymbol{v}^{\mathsf{T}}\boldsymbol{A}\boldsymbol{v} \\
&\leq g(\boldsymbol{x}) + \langle \operatorname{grad} g(\boldsymbol{x}),\, \boldsymbol{v}\rangle + \frac{2\sigma_{\max}(\boldsymbol{A})}{2}\|\boldsymbol{v}\|_2^2.
\end{aligned}$$

Therefore we obtain $L = 2\sigma_{\max}(\boldsymbol{A})$.