Cartan-Sync: Fast and Global SE(d)-Synchronization

Jesus Briales Javier Gonzalez-Jimenez

Abstract— **This work addresses the fundamental problem of Pose Graph Optimization (PGO), which is pervasive in the context of SLAM, and widely known as SE(d)-Synchronization in the mathematical community. Our contribution is twofold. First, we provide a novel, elegant and compact matrix formulation of the Maximum Likelihood Estimation (MLE) for this problem, drawing interesting connections with the Connection Laplacian of a graph object. Secondly, even though the MLE problem is non-convex and computationally intractable in general, we exploit recent advances in convex relaxations of PGO and Riemannian techniques for low-rank optimization to yield an *a-posteriori certifiably globally optimal* algorithm [1] that is also *fast* and *scalable*.**

This work builds upon a fairly demanding mathematical machinery, but beyond the theoretical basis presented, we demonstrate its performance through extensive experimentation in common large-scale SLAM datasets. The proposed framework, `Cartan-Sync`, is up to one order of magnitude faster that the state-of-the-art `SE-Sync` [2] in some important scenarios (e.g. the KITTI dataset).

We make the code for `Cartan-Sync` available at bitbucket.org/jesusbriales/cartan-sync, along with some examples and guides for a friendly use by researchers in the field, hoping to promote further adoption and exploitation of these techniques in the robotics community.

I. Introduction

Pose Graph Optimization (PGO) has become through time an invaluable tool for Robotics, where it stands at the core of *Simultaneous Localization and Mapping* (SLAM) and multi-robot localization. The problem is also pervasive in many other fields including structure-from-motion, camera network calibration, sensor network localization and cryo-electron microscopy, where it is also referred to as the pose synchronization or SE(d)-synchronization problem.

From a formal standpoint, given a collection of unknown *poses* and a set of measured relative *transformations* between them, both seen as orientation-position pairs in the d-dimensional Euclidean space (elements of the *special Euclidean group* $\mathrm{SE}(d) \equiv \mathrm{SO}(d) \ltimes \mathbb{R}^d$), SE($d$)-synchronization seeks to maximize the consistency of the model with the available observations. The typical Maximum-Likelihood Estimation (MLE) formulation of PGO leads to a high-dimensional non-convex optimization problem, which is computationally intractable in general. Because of this, most approaches have traditionally resorted to *local* search techniques (*e.g*. Gauss-Newton, gradient descent or trust region methods [3]–[5]), which perform remarkably well and efficiently given a sufficiently good initial estimate. The *local* nature of these algorithms may result in convergence to (suboptimal) local minima though, which in practice translates to arbitrarily wrong estimates [6], [7].

At this point, we feel obliged to quote the following insightful lines by Rosen *et al*. [8]: "The increasingly widespread adoption of robotic technology in areas such as transportation, medicine, and disaster response has tremendous potential to increase productivity, alleviate suffering, and preserve life. At the same time, however, these high-impact applications often place autonomous systems in safety- and life-critical roles, where misbehavior or undetected failures can carry dire consequences [9]". Because of this, the development of *a-posteriori certifiably correct* methods [1] is an invaluable step towards the safe adoption of robotic technologies in society.

This paper addresses the optimization of the PGO problem, commonly standing at the back-end of different applications, providing a-posteriori *global optimality* guarantees for many practical problems. Note we intentionally leave out the problem of data association, which is per se a challenging task and should be considered elsewhere. Thus, we assume the pose graph provided by the front-end is free of outliers.

Related work: In the context of SLAM, the reliability issues resulting from typical local iterative solvers have been traditionally addressed in a number of ways, mainly focused either on the *initialization problem* to bootstrap the iterative solver or on a better understanding of the global structure of the problem. Recent findings have (*empirically*) shown that, upon an appropriate MLE formulation of the problem, in virtually all practical cases of interest the global solution of the original *hard non-convex* pose synchronization problem is closely related to that of a *tight convex* Semidefinite Program (SDP) relaxation of the same problem [6], [10]–[13]. Thus, when *strong duality* (tightness) holds, this connection has been successfully exploited providing effective means of *certifying* the global optimality of a given candidate solution and also of recovering the optimal solution through the resolution of the relaxed problem. This is a key feature that enables the development of *Probably Certifiably Correct* (PCC) algorithms [1] for PGO, that is, algorithms that with a high probability are able both to efficiently find globally optimal solutions within a restricted range of operation of interest of the generally intractable PGO problem, and to certify to have done so. The relaxed problem is a *semidefinite program* (SDP) though, and while there are mature off-the-shelf SDP solvers for it, in practice they can handle only problems involving up to a few hundred poses.

A closely related and simpler (also fundamental) problem is *rotation synchronization*. This problem has inspired its own extensive literature in many different fields, with

MAPIR-UMA Group. Dept. Ingenieria de Sistemas y Automatica. Instituto de Investigacion Biomedica de Malaga (IBIMA). Universidad de Malaga, 29071 Malaga, Spain. E-mail: jesusbriales@uma.es

similar results and findings to those mentioned above for the pose synchronization case. As for our problem, the MLE formulation of the rotation synchronization can be tightly relaxed to a SDP problem in many cases of interest [14].

In order to overcome the scalability issues imposed by the resolution of the SDP problem, recent works addressing the rotation case have focused on the development of practical algorithms which are able to solve this SDP problem fast and with strong theoretical optimality guarantees [15].

In order to exploit the availability of this fast globally optimal algorithmic approach for rotation synchronization in the pose synchronization problem, recent work by Rosen *et al.* [2], [8] proposes to reduce the SE(d)-synchronization instance to a rotation one by marginalizing the translational components of the problem. They call their algorithm `SE-Sync`, and prove experimentally that, for a wide range of problems, this approach is able to recover globally optimal solutions to pose synchronization and does so more than an order of magnitude faster than the Gauss-Newton-based approach at the basis of current state-of-the-art techniques.

Contribution: In the present work, we present a novel *fast* and *a-posteriori certifiably globally optimal* procedure to solve large-scale instances of SE(d)-synchronization. Unlike prior work [2], [8], the optimization framework is entirely developed within the natural domain of pose synchronization, without relying on any auxiliary marginalization: Rotations and translations are jointly optimized.

This work connects with some recent contributions regarding the Riemannian Staircase procedure [2], [16] and verification and global optimality through the SDP relaxation for SE(d)-synchronization [11], [13]. For questions referring these topics, we strongly suggest reading those materials. Overall, the current contribution provides the following novelties:

- An elegant matrix formulation for the pose synchronization problem (significantly simpler than previous ones [2], [6], [8], [11]–[13]), which links to the Connection Laplacian [17], is provided in Section II.
- The (empirically often) tight SDP relaxation, key to the recovery of a global solution for the non-convex MLE, which is adapted to the new formulation in Section III.
- Finally, in Section IV, we provide an extension of the *Riemannian Staircase* approach [16] which allows us to solve large-scale instances of the SDP relaxation underpinning SE(d)-synchronization.

The crux of our framework is the reformulation of the SDP problem as a Riemannian optimization problem on a very specific *Cartan motion group* [18]. Because of this, we refer to our SE(d)-synchronization method as `Cartan-Sync`.

Experimental evaluation on an extensive set of synthetic and real-world pose-graph SLAM datasets, both for the 2D and 3D case, shows that the computational efficiency of `Cartan-Sync` meets that of `SE-Sync` and, in certain common scenarios, it consistently outperforms `SE-Sync`.

A comprehensive overview of the notation, as well as additional mathematical developments and proofs, are available in the supplementary material [19, Sec. I]. Lastly, we make the code available at bitbucket.org/jesusbriales/cartan-sync in order to promote usability of the presented framework.

II. Connection Laplacian formulation

In this section we present the MLE formulation of SE(d)-synchronization and rewrite it in a compact form that emphasizes the inherent relation of the problem with a specific graph-theoretic object, the Connection Laplacian [17], [20]. The formulation presented in [11], although similar, lacks this straightforward relation with the Connection Laplacian.

The SE(d)-synchronization problem requires to estimate a model consisting of n poses $(\boldsymbol{R}_i, \boldsymbol{t}_i)$ (the unknowns) from m relative measurements $(\bar{\boldsymbol{R}}_{ij}, \bar{\boldsymbol{t}}_{ij})$ (the data), both in the Special Euclidean group SE(d). The set of available measurements can be identified with an undirected graph $G = (V, E)$, with $V = \{1, \dots, n\}$ and $E = \{(i_k, j_k)\}_{k=1}^m \subset V \times V$. Also, to make the formulation clearer, G is assumed *without loss of generality* connected and simple. From here on, we will represent poses through the usual matrix embedding $\boldsymbol{T} = \left[\boldsymbol{R}, \boldsymbol{t}\right] \subset \mathbb{R}^{d\times(d+1)}$, and use $\tilde{\boldsymbol{T}}$ to refer to its *homogeneous* matrix representation [19, Sec. I].

The best model maximizes the consistency with the observations, which yields the optimization problem

$$f^\star_{\text{ML}} = \min_{\{\boldsymbol{T}_i \in \text{SE}(d)\}_{i\in V}} \frac{1}{2} \sum_{(i,j)\in E} \omega^2_{\bar{\boldsymbol{R}}_{ij}} \|\boldsymbol{R}_j - \boldsymbol{R}_i \bar{\boldsymbol{R}}_{ij}\|_F^2 \qquad (1)$$
$$+\omega^2_{\bar{\boldsymbol{t}}_{ij}} \|\boldsymbol{t}_j - \boldsymbol{t}_i - \boldsymbol{R}_i \bar{\boldsymbol{t}}_{ij}\|_2^2.$$

This formulation is consistent with the Maximum-Likelihood Estimation under the assumption of isotropic Langevin and Gaussian distributions for $\bar{\boldsymbol{R}}_{ij}$ and $\bar{\boldsymbol{t}}_{ij}$, with information $\omega^2_{\bar{\boldsymbol{R}}_{ij}}$ and $\omega^2_{\bar{\boldsymbol{t}}_{ij}}$ [2], [6], [8]. For non-isotropic distributions we must *approximate* these by bounding isotropic distributions [7].

The MLE formulation (1) is equivalent to

$$f^\star_{\text{ML}} = \min_{\{\boldsymbol{T}_i \in \text{SE}(d)\}_{i\in V}} \frac{1}{2} \sum_{(i,j)\in E} \|\boldsymbol{T}_j - \boldsymbol{T}_i \tilde{\bar{\boldsymbol{T}}}_{ij}\|^2_{\boldsymbol{\Omega}_{ij}}, \qquad (2)$$

where we define the matrix norm $\|\boldsymbol{M}\|^2_{\boldsymbol{\Omega}} = \text{tr}(\boldsymbol{M}\boldsymbol{\Omega}\boldsymbol{M}^\top)$, with matrix weights $\boldsymbol{\Omega}_{ij} = \text{blkdiag}(\omega^2_{\bar{\boldsymbol{R}}_{ij}} \boldsymbol{I}_d,\ \omega^2_{\bar{\boldsymbol{t}}_{ij}})$.

Let us now define a convenient extension of the graph G, the *matrix* weighted graph $\mathcal{G}$ [17], as the triplet $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$ where $\mathcal{V} = \{\boldsymbol{T}_i\}_{i\in V}$ is the set of node poses, $\mathcal{E} = \{\bar{\boldsymbol{T}}_{ij}\}_{(i,j)\in E}$ is the list of relative pose measurements and $\mathcal{W} = \{\boldsymbol{\Omega}_{ij}\}_{(i,j)\in E}$ is the set of corresponding matrix weights. Then, after some linear algebra [19, Sec. II], we write the MLE problem (2) as

$$f^\star_{\text{ML}} = \min_{\boldsymbol{X}} \frac{1}{2} \text{tr}(\boldsymbol{X}^\top \boldsymbol{Q} \boldsymbol{X}), \quad \boldsymbol{X} \in \text{stack}(\{\text{SE}(d)\}^n)^\top, \quad (3)$$

where $\boldsymbol{X}$ is the transposed *stacked representation* of the node set $\mathcal{V}$, a *column* $[(d+1)\times d]$-block-vector

$$\underset{\substack{[(d+1)\times d] \\ (n\times 1)}}{\underline{\boldsymbol{X}}} = \text{stack}(\mathcal{V})^\top = \left[\,\boldsymbol{T}_1 \middle| \dots \middle| \boldsymbol{T}_n\,\right]^\top \subset \mathbb{R}^{n(d+1)\times d}.$$

The matrix $\boldsymbol{Q}$ in the compact objective form of (3) is core to the synchronization problem as it condenses all the information available in the problem. Interestingly, this matrix is not other than the *Connection Laplacian* [17], [20] of the matrix graph $\mathcal{G}$ defined above. This is a $[(d+1)\times(d+1)](n\times n)$ block-matrix characterized by the *Connection Incidence* matrix $\boldsymbol{A}$ and the *Connection Weight* matrix $\boldsymbol{\Omega}$, a convenient generalization of the common incidence and weight matrices fulfilling the otherwise classical relation $\boldsymbol{Q}(\mathcal{G}) = \boldsymbol{A}(\vec{\mathcal{G}})\boldsymbol{\Omega}(\mathcal{G})\boldsymbol{A}(\vec{\mathcal{G}})^\top$. We refer to [19, Sec. II] for the proof.

Let us adopt an arbitrary orientation[1] of the original *scalar* graph G to obtain the oriented graph $\vec{G}$. The Connection Incidence matrix $\boldsymbol{A}$ of the corresponding oriented *matrix* graph $\vec{\mathcal{G}}$ is the $[(d+1)\times(d+1)](n\times m)$ matrix whose k-th block-column is defined by the measurement in the k-th edge (i_k, j_k) as

$$\boldsymbol{A}(\vec{\mathcal{G}})_{[r,k]} = \begin{cases} -\tilde{\bar{\bar{\boldsymbol{T}}}}_{i_k j_k} & \text{if } r = i_k, \\ +\boldsymbol{I}_{d+1} & \text{if } r = j_k, \\ \mathbf{0}_{d+1} & \text{otherwise.} \end{cases} \tag{4}$$

The $[(d+1)\times(d+1)](m\times m)$ block-diagonal Connection Weight matrix is defined as

$$\boldsymbol{\Omega}(\mathcal{G}) = \text{blkdiag}(\boldsymbol{\Omega}_1, \dots, \boldsymbol{\Omega}_m). \tag{5}$$

III. Global Resolution through SDP relaxation

The hardness in solving the MLE problem arises from the high non-convexity of the domain, which fills the optimization problem with local minima. In this section we present a (*convex*) SDP relaxation, arising from Lagrangian duality, that circumvents the challenge of local minima *in many practical problems*. This relaxation, *if tight*, makes it possible to recover and certify the globally optimal solution of the MLE problem (3). Recent work proves that *tightness* is guaranteed as long as the magnitude of the noise corrupting the measurements $\bar{\boldsymbol{T}}_{ij}$ remains below a certain theoretical bound [2], [8]. The empirical evidence though is that the SDP relaxation remains tight with high probability for noise up to an order of magnitude greater[2] than what is encountered in typical robotics and computer vision applications [2], [6], [8], [12], [13], [21].

In this work the SDP relaxation is tailored to the specific formulation proposed in Section II. The usual algorithmic approach exploiting the relaxation is depicted in Alg. 1 [13].

Algorithm 1: Solving MLE via SDP relaxation

```
Input: Data matrix $\boldsymbol{Q}$ (Connection Laplacian)
Output: $\boldsymbol{X}$, suboptimality bound
/* solve SDP relaxation                         */
$\boldsymbol{\Lambda}^\star, f^\star_{\text{SDP}} \leftarrow$ SolveSDP($\boldsymbol{Q}$);        ▷ Problem (9)
/* recover $\boldsymbol{X}$ via duality theory              */
$\boldsymbol{\mathcal{N}} \leftarrow$ Get nullspace basis for null($\boldsymbol{Q} - \boldsymbol{\Lambda}^\star$);
$\boldsymbol{X} \leftarrow$ MetricUpgrade($\boldsymbol{\mathcal{N}}$);             ▷ See [13]
/* check optimality a-posteriori                */
$f^\star_{\text{ML}} - f^\star_{\text{SDP}} \leq f(\boldsymbol{X}) - f^\star_{\text{SDP}}$
```

A. Forming the SDP relaxation

1) Orthogonal relaxation: To begin with, we relax the condition that $\boldsymbol{T}_i \in \text{SE}(d)$ by the looser condition $\boldsymbol{T}_i \in \text{E}(d)$, with $\text{E}(d) \equiv \text{O}(d) \ltimes \mathbb{R}^d$ the Euclidean group. The *orthogonal relaxation* of the MLE problem (3) is then

$$f^\star_{\text{ort}} = \min_{\boldsymbol{X}} \frac{1}{2} \operatorname{tr}(\boldsymbol{X}^\top \boldsymbol{Q} \boldsymbol{X}), \quad \boldsymbol{X} \in \text{stack}(\{\text{E}(d)\}^n)^\top. \tag{6}$$

There exists both empirical [21] and theoretical [2, App. C] evidence supporting that, *in practical applications*, we can expect this relaxation to be tight so that $f^\star_{\text{ort}} = f^\star_{\text{ML}}$. This relaxation defines the equivalent Quadratically Constrained Quadratic Program (QCQP):

$$f^\star_{\text{ort}} = \min_{\boldsymbol{X}} \frac{1}{2} \operatorname{tr}(\boldsymbol{X}^\top \boldsymbol{Q} \boldsymbol{X}) \tag{7}$$

$$\text{s.t. } \boldsymbol{R}_i^\top \boldsymbol{R}_i = \boldsymbol{I}_d, \quad \forall i = 1, \dots, n. \tag{8}$$

2) Lagrangian relaxation: As a QCQP, the Lagrangian dual of the problem (7) is straightforward to obtain as a *semidefinite program* (SDP) [11], [22] in its dual form

$$f^\star_{\text{SDP}} = \max_{\boldsymbol{\Lambda}} \frac{1}{2} \operatorname{tr}(\boldsymbol{\Lambda}), \quad \text{s.t. } \boldsymbol{\Lambda}_i \in \mathbb{S}^d, \ \boldsymbol{Q} - \boldsymbol{\Lambda} \succcurlyeq \mathbf{0} \tag{9}$$

$$\boldsymbol{\Lambda} = \text{blkdiag}(\boldsymbol{\Lambda}_1, 0, \dots, \boldsymbol{\Lambda}_n, 0), \tag{10}$$

where the dual SDP variables $\boldsymbol{\Lambda}_i \in \mathbb{S}^d, \forall i \in V$ (with $\mathbb{S}^d$ standing for $d \times d$ symmetric matrices) correspond to Lagrange multipliers for the orthogonality constraints. See the supplementary material for more details [19, Sec.III].

B. Undoing the relaxation: MLE solution

The crux for recovering the global solution of the original MLE resides in the fact that, if the relaxations above are tight, the optimal solution $\boldsymbol{X}^\star$ must lie in the nullspace $\boldsymbol{\mathcal{N}}$ of $\boldsymbol{Q} - \boldsymbol{\Lambda}^\star$ (see [11], [13]), with $\boldsymbol{\Lambda}^\star$ the optimal solution to the dual SDP problem (9). Even if tightness does not hold, a good heuristic is that the optimal MLE solution $\boldsymbol{X}^\star$ should still be close to that nullspace [13].

In any case, once we get a basis for null$(\boldsymbol{Q} - \boldsymbol{\Lambda}^\star)$ the recovery of the globally optimal solution $\boldsymbol{X}^\star$ (or a remarkably good initialization $\boldsymbol{X}_0$, if non-tight) is straightforward by applying the *metric upgrade approach* presented in [13].

IV. Fast SDP resolution

The global resolution of the MLE problem through the SDP relaxation as proposed in the previous section (and depicted in Alg. 1) looks algorithmically clear and simple. However, it has an important caveat: The resolution of the SDP problem (9) by traditional off-the-shelf interior-point solvers scales poorly as they imply working with full matrices whose dimension grows with the problem size[3]. This rendered the application of Alg. 1 in previous works [10], [11], [13] unpractical for large-scale pose synchronization problems.

[1] We take edge orientation *from* left (-) *to* right (+) index in the pair (i, j).

[2] No simple condition exists here, but rotation noise seems the principal factor and tightness holds in practice as long as it is below 10 degrees.

[3] A conservative estimate of the complexity for interior point methods solving the SDP (9) is $O(n^{4.5} log(1/\epsilon))$ [23].

This same fact promoted the development of specialized scalable solvers for the SDP relaxation appearing in the related (but simpler) case of rotation synchronization [16]. Thus, Rosen *et al.* [8] built an efficient and scalable approach for pose synchronization by marginalizing this to an equivalent rotation synchronization instance, and then falling back to the corresponding solver in [16].

In this section, we provide and justify an algorithmic approach to *directly* solve large-scale instances of the SDP relaxation (15) in a global, fast, scalable way. This makes it possible to efficiently handle pose synchronization without resorting to any kind of marginalization. For that purpose, we first show in Section IV-A that there exists a complete hierarchy of partial (low-rank) relaxations between the original MLE problem (3) and the primal version of the SDP relaxation (15). Then, we will exploit the existence of such partial relaxations within an algorithmic approach, the Riemannian Staircase in Alg. 2, as explained in Section IV-B.

The resulting method we propose in the present work for solving the MLE problem (1) in SE(d)-synchronization, `Cartan-Sync`, can be found in Alg. 3.

A. Low-rank SDP relaxation

1) Lifting relaxation: Whereas the *dual* SDP problem (9) stems from the Lagrangian dual problem definition applied on the orthogonal relaxation (7), the corresponding *primal* SDP problem (dual of the dual SDP relaxation) is closely related in turn to the *lifted* formulation of the same explicitly constrained problem (7). Exploiting the fact that $\operatorname{tr}(\boldsymbol{X}^\top \boldsymbol{Q}\boldsymbol{X}) = \operatorname{tr}(\boldsymbol{Q}\boldsymbol{X}\boldsymbol{X}^\top)$:

$$f^\star_{\text{ort}} = \min_{\boldsymbol{X}} \frac{1}{2}\operatorname{tr}(\boldsymbol{Q}\boldsymbol{Z}), \quad \boldsymbol{X} \in \mathbb{R}^{n(d+1)\times d} \tag{11}$$

$$\text{s.t. } \boldsymbol{Z}_{[i,i](1:d,1:d)} = \boldsymbol{I}_d, \quad \forall i = 1,\dots,n, \tag{12}$$

$$\boldsymbol{Z} = \boldsymbol{X}\boldsymbol{X}^\top, \tag{13}$$

where the *lifted* variable $\boldsymbol{Z}$ is a $[(d+1)\times(d+1)](n\times n)$ block-matrix (the same dimensions as the data matrix $\boldsymbol{Q}$). The *lifting* constraint (13) making this formulation equivalent to the original (non-lifted) problem (7) can be written as

$$\boldsymbol{Z} = \boldsymbol{X}\boldsymbol{X}^\top \iff \begin{cases} \boldsymbol{Z} \succcurlyeq \boldsymbol{0}, \\ \boldsymbol{Z} \in \mathbb{S}^{n(d+1)}, \\ \operatorname{rank}(\boldsymbol{Z}) = d. \end{cases} \tag{14}$$

We can see that if we relax the non-convex rank constraint above, the remaining problem is the primal SDP, dual of the dual SDP (9),

$$f^\star_{\text{SDP}} = \min_{\boldsymbol{Z}} \frac{1}{2}\operatorname{tr}(\boldsymbol{Q}\boldsymbol{Z}), \quad \boldsymbol{Z} \in \mathbb{S}^{n(d+1)} \tag{15}$$

$$\text{s.t. } \boldsymbol{Z}_{[i,i](1:d,1:d)} = \boldsymbol{I}_d, \quad \forall i = 1,\dots,n, \tag{16}$$

$$\boldsymbol{Z} \succcurlyeq \boldsymbol{0}. \tag{17}$$

and the lifted unknown $\boldsymbol{Z}$ is actually the primal SDP variable.

2) Partial lifting relaxation: It is clear from the results above that the rank constraint $\operatorname{rank}(\boldsymbol{Z}) = d$ in the lift constraint is fully responsible for the non-convexity of the synchronization problem and dropping it yields the usual primal SDP relaxation (15). However, in practice, the solution $\boldsymbol{Z}$ to the primal SDP (15) turns to be *low-rank*. In fact, in the common circumstance that the relaxation is tight, $\operatorname{rank}(\boldsymbol{Z}) = d$. Because of this, similarly to recent works [8], [16] addressing the SDP relaxation of rotation synchronization, we propose to use the aproach by Burer and Monteiro [24], [25]: We relax the rank constraint in (14) to a softer constraint $\operatorname{rank}(\boldsymbol{Z}) \leq p$, where $d \leq p \ll n(d+1)$. In that case it follows that $\boldsymbol{Z} = \hat{\boldsymbol{X}}\hat{\boldsymbol{X}}^\top$, where $\hat{\boldsymbol{X}} \in \mathbb{R}^{n(d+1)\times p}$. This partially-lifted relaxation closely resembles the original lifted formulation (11),

$$f^\star_{\text{SDPLR}} = \min_{\hat{\boldsymbol{X}}} \frac{1}{2}\operatorname{tr}(\boldsymbol{Q}\boldsymbol{Z}), \quad \hat{\boldsymbol{X}} \in \mathbb{R}^{n(d+1)\times p} \tag{18}$$

$$\text{s.t. } \boldsymbol{Z}_{[i,i](1:d,1:d)} = \boldsymbol{I}_d, \quad \forall i = 1,\dots,n, \tag{19}$$

$$\boldsymbol{Z} = \hat{\boldsymbol{X}}\hat{\boldsymbol{X}}^\top. \tag{20}$$

Note this relaxation remains *non-convex*.

3) Unconstrained Riemannian optimization: Consider the block structure that $\hat{\boldsymbol{X}}$ inherits from $\boldsymbol{X}$ through the lifting process (see details in [19, Fig. 1]),

$$\underset{(n\times 1)}{\underset{[(d+1)\times p]}{\hat{\boldsymbol{X}}}} = \left[\hat{\boldsymbol{T}}_1 \middle| \dots \middle| \hat{\boldsymbol{T}}_n\right]^\top = \operatorname{stack}(\{\hat{\boldsymbol{T}}_i\}_{i\in V})^\top, \tag{21}$$

where each *lifted pose* $\hat{\boldsymbol{T}}_i = \left[\hat{\boldsymbol{R}}_i, \hat{\boldsymbol{t}}_i\right] \subset \mathbb{R}^{p\times(d+1)}$ involves the lifting of its components $\hat{\boldsymbol{R}}_i \subset \mathbb{R}^{p\times d}, \hat{\boldsymbol{t}}_i \in \mathbb{R}^p$.

The block diagonal constraints in (19) lead then to the equivalent constraint $\hat{\boldsymbol{R}}_i^\top \hat{\boldsymbol{R}}_i = \boldsymbol{I}_d$ on each $\hat{\boldsymbol{R}}_i$. This implies that the columns of $\hat{\boldsymbol{R}}_i$ are orthonormal or, otherwise stated, $\hat{\boldsymbol{R}}_i$ belongs to a Stiefel manifold, $\hat{\boldsymbol{R}}_i \in St(p,d)$ [26]. It follows that the lifted pose $\hat{\boldsymbol{T}}_i$ lies in another manifold, the Cartan motion group $St(p,d) \ltimes \mathbb{R}^p$ [18].

Let us define the manifold

$$\mathcal{M}_p \equiv \operatorname{stack}(\{St(p,d) \ltimes \mathbb{R}^p\}^n)^\top, \tag{22}$$

where $p \geq d$ parameterizes the inner partial lift dimension of the Burer-Monteiro relaxation. Then the partially lifted problem (18) is equivalent to the *unconstrained* Riemannian optimization problem

$$f^\star_{\text{SDPLR}} = \min_{\hat{\boldsymbol{X}} \in \mathcal{M}_p} \frac{1}{2}\operatorname{tr}(\hat{\boldsymbol{X}}^\top \boldsymbol{Q}\hat{\boldsymbol{X}}). \tag{23}$$

B. The Riemannian Staircase

Akin to the rotation synchronization case [15], [16], we exploit the availability of the Riemannian formulation (23) to provide a *Riemannian Staircase* procedure that solves the primal SDP relaxation (15) in a fast and scalable way.

Optimality guarantees: Before introducing the approach, let us show the following results (adapted from Boumal *et al.* [15]) which justify the theoretical basis for the procedure:

Proposition 1 (Global optima of Riemannian lift (23))**.** *Provided that the primal SDP problem* (15) *has a compact search space and the manifold* $\mathcal{M}_p$ *in* (23) *is smooth:*

1) *If* $\hat{\boldsymbol{X}} \in \mathcal{M}_p$ *is a rank-deficient second-order critical point*[4] *of the Riemannian problem* (23)*, then* $\hat{\boldsymbol{X}}$ *is a global minimizer of* (23) *and* $\boldsymbol{Z}^\star = \hat{\boldsymbol{X}}\hat{\boldsymbol{X}}^\top$ *is a solution of the primal SDP problem* (15)*.*
2) *If* $p \geq \left\lceil \sqrt{(d+1)(d+2)n} \right\rceil$*, then with probability* 1 *every first-order critical point of the Riemannian problem* (23) *is rank-deficient.*

Further details regarding Proposition 1 may be found in [15]. Note that, even though the search space of the SDP problem (15) is not compact, the behavior described above is consistent with that empirically observed for the pose synchronization case. A complete theoretical proof that justifies Proposition 1 for the non-compact domain of SE(d)-synchronization seems plausible, but is left for future work.

The algorithm: Proposition 1 supports the *Riemannian Staircase* procedure [8], [16]. To reach the optimal solution of the SDP problem (15), we solve successively higher partial liftings (with increasing p) in the Riemannian formulation (23) until a globally optimal solution is found. This should happen for $p \leq \left\lceil \sqrt{(d+1)(d+2)n} \right\rceil$, according to Prop. 1.2 and, in fact, the actual number of steps required in the Riemannian staircase is much lower in practice, usually just one or two. Intuitively, the good behavior of the approach hints that the relatively few extra d.o.f. gained in the partially lifted relaxation (18) are enough to circumvent local minima in the higher dimensional search space of $\mathcal{M}_p$.

The remainder of this section will be devoted then to provide a detailed exposition of the steps involved in the Riemannian staircase procedure. These appear also summarized in Alg. 2.

1) Solving the Riemannian problem (23)*:* Similarly to previous works [8], [16], we will use the *truncated-Newton Riemannian trust-region* (RTR) algorithm [27], [28] as implemented in the *Manopt toolbox* [29]. This solver offers superlinear convergence and is appropriate for large-scale optimization problems, allowing for the scalability of the method with the size of the synchronization problem.

In order to apply RTR we need to define several operations related to the Riemannian geometry of the problem (23). Current state-of-the-art Riemannian optimization suites such as *Manopt* abstract most of this complexity allowing to setup the optimization with a minimum amount of mathematical effort. Because of this, we will show here the minimal amount of information regarding the use of *Manopt*. A more comprehensive description of the Riemannian geometry of the problem and related operators may be found in the supplementary material [19, Sec.V].

Algorithm 2: Riemannian Staircase Algorithm

```
   Input: Initial estimate[5] $\hat{X}_0 \in \{\mathrm{SE}(d)\}^n$, $p_0$
   Input: Problem data $Q$, $\mathrm{chol}(Q)$
   Output: Optimal SDP solution $(\hat{X}^\star, f^\star_{\text{SDP}})$
 1 $\mathcal{M} \leftarrow$ Setup manifold $\{St(p_0, d) \ltimes \mathbb{R}^{p_0}\}^n$;
 2 $\hat{X}, f \leftarrow$ RTR($\mathcal{M}$,$\hat{X}_0$);
 3 $(\lambda_{\min}, v) \leftarrow$ Get optimality certificate [19, Sec. VIII];
 4 if $\lambda_{\min} < 0$ then
       /* suboptimal point [19, Sec. IX]                */
 5     $p = p + 1$;        ▷ Increase partial lifting
 6     $\mathcal{M} \leftarrow$ Setup manifold $\{St(p, d) \ltimes \mathbb{R}^p\}^n$;
 7     $\hat{X}_0 \leftarrow$ EscapeSaddle($\mathcal{M}$, $\hat{X}$, $v$); ▷ See [16]
 8     go to line 2;                 ▷ Repeat algorithm
 9 end
   /* reached global solution                          */
10 return $\hat{X}^\star \leftarrow \hat{X}$, $f^\star_{SDP} \leftarrow f$
```

That said, for the resolution in *Manopt* we identify the appropriate manifold structure in the toolbox, which in our case is the n-power manifold of the product manifold $St(p, d) \times \mathbb{R}^p$. Secondly, we provide the cost function $f(\hat{\boldsymbol{X}}) = \frac{1}{2}\operatorname{tr}(\hat{\boldsymbol{X}}^\top \boldsymbol{Q}\hat{\boldsymbol{X}})$ and its (Euclidean) derivatives (in the ambient space)

$$\nabla f(\hat{\boldsymbol{X}}) = \boldsymbol{Q}\hat{\boldsymbol{X}}, \quad \nabla^2 f(\hat{\boldsymbol{X}})[\boldsymbol{U}] = \boldsymbol{Q}\boldsymbol{U}. \tag{24}$$

Finally, we pass this information to the builtin RTR solver.

Even though the information above is in principle enough to apply RTR, and it is the way RTR is applied in the recent `SE-Sync` method [2], it is insufficient for the *efficient* resolution of the Riemannian problem (23) in the `Cartan-Sync` framework. As we will see in the experiments of Section V, this naïve version of RTR is not practical as it is too slow to converge. To understand this, note that a key step in the employed RTR algorithm is the resolution of the inner Trust Region (TR) subproblem by the *truncated Conjugate Gradient* (tCG) method [30]. As for classical Conjugate Gradient, the performance of tCG is strongly connected to the conditioning of the problem at hand. This motivates the search for an appropriate Hessian preconditioner.

Riemannian Hessian preconditioner: The *Riemannian* preconditioner at a specific point $\hat{\boldsymbol{X}}$ must be a linear, symmetric, positive definite operator from $T_{\hat{\boldsymbol{X}}}\mathcal{M}_p$ to $T_{\hat{\boldsymbol{X}}}\mathcal{M}_p$, where $T_{\hat{\boldsymbol{X}}}\mathcal{M}_p$ denotes the tangent space of the manifold at point $\hat{\boldsymbol{X}}$ [27]. Ideally, it should approximate the inverse of the Riemannian Hessian well while being also fast to compute. That is, given a vector $\boldsymbol{H}$ in $T_{\hat{\boldsymbol{X}}}\mathcal{M}_p$, the returned tangent vector $\boldsymbol{U} = \mathrm{Precon}(\hat{\boldsymbol{X}})[\boldsymbol{H}]$ should fulfill $\mathrm{Hess}(\hat{\boldsymbol{X}})[\boldsymbol{U}] \approx \boldsymbol{H}$. The Riemannian Hessian involves both

[4] A second-order critical point satifies the *Riemannian* gradient is zero and the *Riemannian* Hessian is positive-semidefinite [27].

[5] Please, notice that the initialization does *not* compromise the global convergence, but it may improve substantially the convergence speed of the iterative method.

the Euclidean Hessian and the Euclidean gradient[6]

$$\operatorname{Hess} f(\hat{\boldsymbol{X}})[\boldsymbol{U}] = \operatorname{proj}_{T_{\hat{\boldsymbol{X}}}\mathcal{M}_p}(\nabla^2 f(\hat{\boldsymbol{X}})[\boldsymbol{U}]) + O(\nabla f(\hat{\boldsymbol{X}})),$$

where $\operatorname{proj}_{T_{\hat{\boldsymbol{X}}}\mathcal{M}_p}(\boldsymbol{U})$ is the *orthogonal projection operator* from the ambient Euclidean space onto $T_{\hat{\boldsymbol{X}}}\mathcal{M}_p$ [27],

$$\operatorname{proj}_{T_{\hat{\boldsymbol{X}}}\mathcal{M}_p} : \mathbb{R}^{n(d+1)\times p} \to T_{\hat{\boldsymbol{X}}}\mathcal{M}_p. \tag{25}$$

We assume the second term is relatively small near critical points to get an approximate Riemannian Hessian $\operatorname{proj}_{\hat{\boldsymbol{X}}}(\nabla^2 f(\hat{\boldsymbol{X}})[\boldsymbol{U}])$. However, applying the inverse of this approximate Hessian as a preconditioner amounts to solve $\boldsymbol{U} \in T_{\hat{\boldsymbol{X}}}\mathcal{M}_p$ in the linear problem defined by

$$\operatorname{proj}_{\hat{\boldsymbol{X}}}(\boldsymbol{Q}\boldsymbol{U}) = \boldsymbol{H}, \quad \boldsymbol{H} \in T_{\hat{\boldsymbol{X}}}\mathcal{M}_p, \tag{26}$$

which is not straightforward or cheap to solve either. Instead,

$$\boldsymbol{U} = \operatorname{Precon}(\hat{\boldsymbol{X}})[\boldsymbol{H}] = \operatorname{proj}_{\hat{\boldsymbol{X}}}(\boldsymbol{Q}^{-1}\boldsymbol{H}) \tag{27}$$

is used. This heuristic alternative fulfills all the necessary conditions for a preconditioner as shown in the supplementary material [19, Sec.VI] and, despite its heuristic nature, performs quite well in practice as we will see in the experiments.

In the current implementation of Cartan-Sync, we cache a Cholesky factorization of $\boldsymbol{Q}$ with approximate minimum degree permutation (that promotes sparsity in the Cholesky factor) at the beginning of the optimization to speed up the recurrent computation of $\boldsymbol{Q}^{-1}\boldsymbol{H}$. Check the supplementary material [19, Sec.VII] for details.

2) Testing global optimality: Due to the local nature of the algorithms employed to solve the Riemannian problem (23), the prior resolution step may still converge to a suboptimal point. This unwanted situation can be detected testing the global optimality of the critical point $\hat{\boldsymbol{X}}$ using a similar verification procedure to that in [11]. A detailed overview is provided in the supplementary material [19, Sec. VIII].

3) Escaping suboptimal minima: If the critical point $\hat{\boldsymbol{X}}$ turns to be suboptimal, we increase the rank p of the partial lift (Sec. IV-A) and keep on optimizing on the extended search space. The specific procedure is provided in the supplementary material [19, Sec. IX].

4) Recovery of feasible estimate: The *global* solution $\hat{\boldsymbol{X}}^\star$ of the Riemannian problem (23) must fulfill the complementary slackness condition

$$(\boldsymbol{Q} - \boldsymbol{\Lambda}^\star)\hat{\boldsymbol{X}}^\star = \boldsymbol{0} \Rightarrow \operatorname{span}(\hat{\boldsymbol{X}}^\star) \equiv \operatorname{null}(\boldsymbol{Q} - \boldsymbol{\Lambda}^\star). \tag{28}$$

As a result, we can directly apply the MLE recovery method of Section III-B on $\hat{\boldsymbol{X}}^\star$ as a basis for $\operatorname{null}(\boldsymbol{Q} - \boldsymbol{\Lambda}^\star)$.

V. Experimental results

In order to evaluate the goodness of the approach proposed in this work we use an extensive list of large-scale problems, both in 2D and 3D, drawn from the motivating application of pose-graph SLAM. We thoroughly compare our approach to the state-of-the-art RTR-based method SE-Sync [2]. The traditional Gauss-Newton-based approach, GN, is included in the comparative for completeness as well. For all methods we use the state-of-the-art chordal initialization [31].

Algorithm 3: The Cartan-Sync Algorithm

```
Input: Initial estimate[5] $\boldsymbol{X}_0 \in \{\mathrm{SE}(d)\}^n$, $p_0 \geq d$
Output: Estimate $\boldsymbol{X} \in \{\mathrm{SE}(d)\}^n$, bound $f^\star_{\mathrm{SDP}} \leq f^\star_{\mathrm{ML}}$
$\boldsymbol{Q}, \operatorname{chol}(\boldsymbol{Q}) \leftarrow$ Build Conn. Laplacian and Cholesky;
$\hat{\boldsymbol{X}}^\star, f^\star_{\mathrm{SDP}} \leftarrow$ RiemannStaircase(...);   ▷ Alg. 2
$\boldsymbol{X} \leftarrow$ MetricUpgrade($\hat{\boldsymbol{X}}^\star$);            ▷ See [13]
return $\boldsymbol{X}, d^\star_{SDP}$
```

Note that the main focus of the performed experimentation is on measuring computational performance when solving the MLE problem (1). All experiments have been conducted on an Intel Core i5-6600 3.30GHz CPU. On the other hand, we skip here an extensive analysis on tightness for the SDP relaxation, which is already provided elsewhere [2], [6], [8], [10], [11], [13]. Specifically, we refer to [13] for a deeper examination of the effectiveness of the SDP relaxation for specially challenging problem regimes where tightness does not hold in general.

A comprehensive summary of the experimental results is contained in Tab. I: For each dataset, the number of poses (n) and relative measurements (m) as well as dimensionality (d) are provided. A simple measure of connectivity is given as the ratio of loop closures ($r_{\mathrm{LC}} = (m - m_{\mathrm{odom}})/\binom{n}{2}$), followed by the optimal objective value $f^\star_{\mathrm{ML}}$ according to the MLE formulation (1), which is attained by all the methods in the given datasets. For all the evaluated datasets the SDP relaxation is tight, so we can certify global optimality a-posteriori. For each method we show the convergence time from chordal initialization, as well as the required number of steps. For the RTR-based methods both the number of outer (TR) and total inner (tCG) steps are displayed [28].

Datasets: The 3D datasets employed are the same as in [31], of which sphere, torus and grid are synthetic whereas garage, cubicle and rim are real-world examples. The 2D datasets comprise customary evaluation datasets, such as CSAIL, manhattan, intel and ais2klinik as well as a subset of the KITTI datasets with loop closures, as provided in [32]. For those datasets which were not originally isotropic we applied the same bounding approximation with isotropic covariances employed in previous works [2], [7].

Computational performance: As discussed in Section IV, the numerical resolution of the Riemannian problem (23) within Cartan-Sync requires a preconditioner to converge fast. This is evidenced by Fig. 1(b), as in most cases RTR with no preconditioning provides the slowest convergence.

The reference method SE-Sync does not apply any preconditioner either, yet it provides in general faster convergence than the naive Cartan-Sync w/o preconditioning. The potential explanation for this lies at the marginalization step at the core of SE-Sync, as this entails some kind of *data normalization* and makes the condition number of the

[6] Check the supplementary material [19, Sec.V] for a full characterization of the Riemannian Hessian.

						Chordal+GN		SE-Sync [2]		Cartan-Sync [ours]	
Dataset	d	n	m	r_{LC}	$f^\star_{\text{ML}}$	Time [s]	#iter	Time [s]	#outer (#inner)	Time [s]	#outer (#inner)
sphere	3	2500	4949	7.8×10^{-4}	8.435×10^{2}	17.04	8	0.54	4 (64)	1.60	4 (69)
sphere-a	3	2200	8647	2.7×10^{-3}	1.481×10^{6}	67.13	13	0.66	4 (92)	0.79	8 (19)
torus	3	5000	9048	3.2×10^{-4}	1.211×10^{4}	37.95	6	1.64	3 (84)	0.81	3 (15)
cube	3	8000	22236	4.4×10^{-4}	4.216×10^{4}	146.80	4	5.26	4 (98)	6.68	4 (32)
garage	3	1661	6275	3.3×10^{-3}	6.313×10^{-1}	20.31	7	4.03	4 (383)	10.41	5 (750)
cubicle	3	5750	16869	6.7×10^{-4}	3.586×10^{2}	163.73	9	12.81	4 (312)	6.28	4 (154)
rim	3	10195	29743	3.8×10^{-4}	2.730×10^{3}	716.51	13	31.25	5 (434)	26.81	5 (365)
CSAIL	2	1045	1172	2.3×10^{-4}	1.585×10^{1}	0.21	2	1.20	3 (382)	0.23	3 (26)
manhattan	2	3500	5453	3.2×10^{-4}	3.216×10^{3}	1.88	3	3.73	3 (365)	0.60	3 (30)
city10000	2	10000	20687	2.1×10^{-4}	3.193×10^{2}	23.26	4	21.49	4 (84)	4.95	3 (110)
intel	2	1728	2512	5.3×10^{-4}	2.617×10^{1}	1.03	5	2.47	4 (360)	0.71	3 (66)
ais2klinik	2	15115	16727	1.4×10^{-5}	9.426×10^{1}	33.87	9	131.11	16 (2287)	53.65	10 (1259)
KITTI_00	2	4541	4677	1.3×10^{-5}	6.285×10^{1}	1.98	4	18.03	12 (1800)	1.34	3 (63)
KITTI_02	2	4661	4703	4.0×10^{-6}	5.418×10^{1}	1.99	4	20.77	21 (3150)	1.29	3 (72)
KITTI_05	2	2761	2826	1.7×10^{-5}	1.383×10^{2}	0.83	3	4.09	6 (900)	0.44	3 (29)
KITTI_06	2	1101	1150	8.3×10^{-5}	1.766×10^{1}	0.17	2	1.79	4 (600)	0.14	2 (14)
KITTI_07	2	1101	1106	9.9×10^{-6}	1.197×10^{1}	0.24	3	3.16	7 (1050)	0.26	3 (28)
KITTI_09	2	1591	1592	1.6×10^{-6}	3.065×10^{1}	0.49	4	6.62	15 (2250)	0.43	4 (39)

TABLE I

RESULTS FOR THE PGO (SLAM) BENCHMARK DATASETS

underlying data matrix $\boldsymbol{Q}$ significantly smaller, as shown in Fig. 1(a). This is consistent with the fundamental fact that rotations and translations belong to a compact and non-compact group, respectively, leading to poor conditioning [18]. As a downside, the marginalization step in SE-Sync hinders the simplicity of the optimization objective, introducing the inverse of a matrix and thus the necessity to solve a linear system each time the cost, gradient or Hessian are computed.

In Cartan-Sync we exploit instead the simplicity of the objective to directly head for the ill-conditioning of the problem applying an appropriate preconditioner. The results displayed both in Tab. I and Fig. 1(b) show that the simple heuristic proposed preconditioner (27) corrects much of the ill-conditioning in the problem. As a result, the actual Cartan-Sync approach (with preconditioning) meets the performance of SE-Sync in the 3D datasets resulting in optimization times of similar order and clearly overperforms SE-Sync for the 2D datasets. Note specially how the PGO in the KITTI datasets turns specially hard to deal with for the SE-Sync approach, whereas our preconditioned Cartan-Sync presents a remarkably good behavior. This is relevant as the graph topology stemming from the KITTI datasets is representative of an important problem, that of driving through a city. An important trait characterizing these datasets is the extreme sparseness of the underlying graph, as loop closure occurs only at very particular points (cross-sections). This reflects in the low value of r_{LC} in Tab. I.

Further insight on how convergence differs between SE-Sync and Cartan-Sync is gained from the number of outer and inner iterations in Tab. I. When the conditioning is not good, the number of outer and inner iterations grows remarkably, resulting in slower convergence.

Regarding the more traditional Gauss-Newton approach, our results are consistent with those in [2] for the 3D datasets: Both SE-Sync and Cartan-Sync are notably faster than traditional Gauss-Newton. Interestingly though, for the 2D datasets GN was always faster than SE-Sync, but still slower than Cartan-Sync for most datasets.

Global convergence: In all the evaluated cases the SE-Sync and Cartan-Sync algorithms attained a certified globally optimal solution. The optimal value $f^\star_{\text{ML}}$ obtained for the MLE objective (3), shown in Table I, was the same for both approaches[7]. The maximum suboptimality bound provided by both algorithms was of the same order (10^{-11}), near numerical precision.

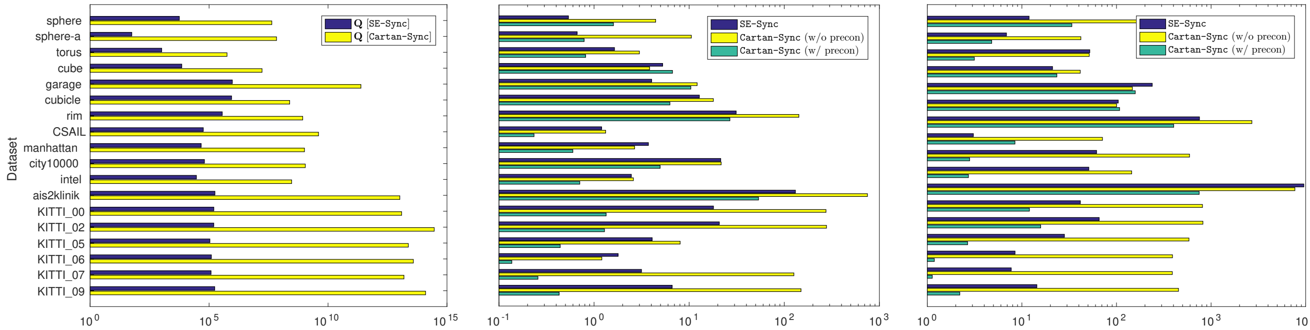
As argued in Section IV-B, we expect Cartan-Sync to offer global convergence regardless of the initialization. To check this feature we solved the datasets under random initialization. In this case both approaches (SE-Sync and Cartan-Sync) still converged to a certified global optimum (logically with the same $f^\star_{\text{ML}}$), although at a much higher computational cost as shown in Fig. 1(c), proving the effectiveness of a good initialization for fast convergence.

VI. CONCLUSION

We have formulated the MLE problem for SE(d)-synchronization in a novel way that provides an explicit relation to the Connection Laplacian of a special graph object. Then, we have shown how to exploit the SDP relaxation of this problem in conjunction with Riemannian optimization to provide a fast solver that certifies global optimality a-posteriori for all the evaluated practical instances. The crux of the approach lies on the partial lifting of the search space characterized as a Cartan motion group which, to the best of our knowledge, has not been explored before in the literature.

Beyond the immediate improvement in performance obtained w.r.t. to the related state-of-the-art approach SE-Sync, we think our approach, Cartan-Sync, has

[7] Note the *half* factor introduced in our formulation when comparing to results in [2].



(a) Condition number (b) Optimization times (chordal init.) [s] (c) Optimization times (random init.) [s]

Fig. 1. Please note the logarithmic scale in the X axis. (a) Condition number of the data matrix $\boldsymbol{Q}$ in `SE-Sync` and `Cartan-Sync`. For `Cartan-Sync` we anchored $\boldsymbol{Q}$ to remove the trivial zero eigenvalue [19, Sec. VII]. (b) Optimization times from chordal initialization [31]. (c) Optimization times from random initialization.

great potential for further enhancement. Namely, the design of better preconditioning approaches is a promising venue for future research.

Acknowledgements

We would like to thank Nicolas Boumal and Bamdev Mishra for the invaluable help with Manopt [29] as well as for pointing at potential issues with the conditioning of the problem and the subsequent extensive discussion on possible preconditioning approaches. We also thank the anonymous reviewers for their valuable comments.

This work has been supported by the Spanish grant program FPU14/06098 and the project PROMOVE (DPI2014-55826-R), funded by the Spanish Government and the "European Regional Development Fund".

References

[1] A. Bandeira, “A note on probably certifiably correct algorithms,” *Comptes Rendus Math.*, 2016.

[2] D. M. Rosen, L. Carlone, A. S. Bandeira, and J. J. Leonard, “SE-Sync: A Certifiably Correct Algorithm for Synchronization over the Special Euclidean Group,” *arXiv Prepr. arXiv1612.07386*, pp. 1–49, 2016.

[3] R. Kummerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, “G2o: A general framework for graph optimization,” *2011 IEEE Int. Conf. Robot. Autom.*, pp. 3607–3613, may 2011.

[4] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, “iSAM2: Incremental smoothing and mapping using the Bayes tree,” *Int. J. Rob. Res.*, 2011.

[5] D. M. Rosen, M. Kaess, and J. J. Leonard, “RISE: An incremental trust-region method for robust online sparse least-squares estimation,” *Robot. IEEE Trans.*, vol. 30, no. 5, pp. 1091–1108, 2014.

[6] L. Carlone, D. M. Rosen, G. Calafiore, J. J. Leonard, and F. Dellaert, “Lagrangian duality in 3D SLAM: Verification techniques and optimal solutions,” in *Intell. Robot. Syst. (IROS), 2015 IEEE/RSJ Int. Conf.*

[7] D. M. Rosen, C. DuHadway, and J. J. Leonard, “A convex relaxation for approximate global optimization in simultaneous localization and mapping,” in *Robot. Autom. (ICRA), 2015 IEEE Int. Conf.*

[8] D. M. Rosen, L. Carlone, A. S. Bandeira, and J. J. Leonard, “A Certifiably Correct Algorithm for Synchronization over the Special Euclidean Group,” *Int. Work. Algorithmic Found. Robot.*, nov 2016.

[9] B. Vlasic and N. E. Boudette, “As U.S. investigates fatal Tesla crash, com- pany defends Autopilot system.”

[10] L. Carlone and F. Dellaert, “Duality-based verification techniques for 2D SLAM,” in *Intl. Conf. Robot. Autom. (ICRA)*, 2015.

[11] J. Briales and J. González-Jiménez, “Fast Global Optimality Verification in 3D SLAM,” in *Int. Conf. Intell. Robot. Syst.*, IEEE/RSJ, 2016.

[12] L. Carlone, G. C. Calafiore, C. Tommolillo, and F. Dellaert, “Planar Pose Graph Optimization: Duality, Optimal Solutions, and Verification,” *IEEE Trans. Robot.*, vol. 32, no. 3, pp. 545–565, 2016.

[13] J. Briales and J. González-Jiménez, “Initialization of 3D Pose Graph Optimization using Lagrangian duality,” in *Int. Conf. Robot. Autom. (ICRA)*, IEEE, 2017.

[14] A. Bandeira, N. Boumal, and A. Singer, “Tightness of the maximum likelihood semidefinite relaxation for angular synchronization,” *Math. Program.*, 2016.

[15] N. Boumal, V. Voroninski, and A. Bandeira, “The non-convex Burer-Monteiro approach works on smooth semidefinite programs,” in *Adv. Neural Inf. Process. Syst.*, pp. 2757–2765, 2016.

[16] N. Boumal, “A Riemannian low-rank method for optimization over semidefinite matrices with block-diagonal constraints,” *arXiv*, 2015.

[17] A. Singer and H.-T. Wu, “Vector diffusion maps and the connection Laplacian,” *Commun. pure Appl. Math.*, vol. 65, no. 8, 2012.

[18] O. Ozyesil, N. Sharon, and A. Singer, “Synchronization over Cartan motion groups via contraction,” *arXiv1612.00059*, nov 2016.

[19] J. Briales and J. Gonzalez-Jimenez, “Cartan-Sync: Fast and Global SE(d)-Synchronization - Supplementary Material.” Online. Available: http://mapir.isa.uma.es/jbriales/RAL17-suppl.pdf.

[20] A. S. Bandeira, A. Singer, and D. A. Spielman, “A Cheeger inequality for the graph connection Laplacian,” *SIAM J. Matrix Anal. Appl.*, vol. 34, no. 4, pp. 1611–1630, 2013.

[21] R. Tron, D. M. Rosen, and L. Carlone, “On the Inclusion of Determinant Constraints in Lagrangian Duality for 3D SLAM,” *Robot. Sci. Syst. Work. “The Probl. Mob. Sensors”*, 2015.

[22] Z.-Q. Luo, W.-K. Ma, A. M.-C. So, Y. Ye, and S. Zhang, “Semidefinite relaxation of quadratic optimization problems,” *IEEE Signal Process. Mag.*, vol. 27, no. 3, pp. 20–34, 2010.

[23] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge University Press, 2004.

[24] S. Burer and R. D. C. Monteiro, “A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization,” *Math. Program.*, vol. 95, no. 2, pp. 329–357, 2003.

[25] S. Burer and R. D. C. Monteiro, “Local minima and convergence in low-rank semidefinite programming,” *Math. Program.*, 2005.

[26] A. Edelman, T. A. Arias, and S. T. Smith, “The geometry of algorithms with orthogonality constraints,” *SIAM J. Matrix Anal. Appl.*, 1998.

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

[28] P.-A. Absil, C. G. Baker, and K. A. Gallivan, “Trust-region methods on Riemannian manifolds,” *Found. Comput. Math.*, 2007.

[29] N. Boumal, B. Mishra, P.-A. Absil, and R. Sepulchre, “Manopt, a Matlab toolbox for optimization on manifolds,” *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1455–1459, 2014.

[30] A. R. Conn, N. I. M. Gould, and P. L. Toint, *Trust region methods*. SIAM, 2000.

[31] L. Carlone, R. Tron, K. Daniilidis, and F. Dellaert, “Initialization techniques for 3D SLAM: a survey on rotation estimation and its use in pose graph optimization,” in *Robot. Autom. (ICRA), 2015 IEEE Int. Conf.*, pp. 4597–4604, IEEE, 2015.

[32] Y. Latif, C. Cadena, and J. Neira, “Robust graph SLAM back-ends: A comparative analysis,” in *2014 IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pp. 2683–2690, IEEE, 2014.

Cartan-Sync: Fast and Global SE(d)-Synchronization Supplementary material

Jesus Briales Javier Gonzalez-Jimenez

Contents

I. Notation

The main matrices involved in the SDP relaxation, namely $\boldsymbol{Q}$, $\boldsymbol{Z}$ and $\boldsymbol{\Lambda}$, are thought of as block-matrices with blocks of size $(d+1)\times(d+1)$. Therefore, the block-vectors (thought of as column vectors with block-elements), such as $\boldsymbol{X}$ or $\hat{\boldsymbol{X}}$ in the paper, have block-sizes of the form $(d+1)\times(*)$. Subscript indexing between brackets, such as $\boldsymbol{M}_{[i,j]}$, refers then to the block in i-th row and j-th column of blocks. If further indexing within the block is required we concatenate it between parentheses as $\boldsymbol{M}_{[i,j](a,b)}$, which refers to the element in position (a,b) in the referenced block. This notation extends to block-vectors as well.

Matlab-like notation is chosen for list of consecutive indeces: $(a:b)\equiv\{a,a+1,\ldots,b-1,b\}, a<b$.

In the context of poses, the convention we choose for representation is $\boldsymbol{T}=\left[\boldsymbol{R},\boldsymbol{t}\right]$, that is, horizontal concatenation of rotation and translation. Becase of this, we will often refer to the rotation block inside a pose as $\boldsymbol{T}_{(\boldsymbol{R})}$. Similarly, for the translation block we use $\boldsymbol{T}_{(\boldsymbol{t})}$. This convention may appear mixed with that of block-indexing, e.g. $\boldsymbol{X}_{[i](\boldsymbol{R})}$ stands for the rotation block in the i-th block of a block-vector $\boldsymbol{X}$ of poses.

The matrix norm $\|\boldsymbol{M}\|_F$ stands for the Frobenius norm, which fulfills $\|\boldsymbol{M}\|_F^2=\operatorname{tr}(\boldsymbol{M}\boldsymbol{M}^\top)$. Because of this we define the more general matrix norm $\|\boldsymbol{M}\|_{\boldsymbol{\Omega}}^2$ as

$$\|\boldsymbol{M}\|_{\boldsymbol{\Omega}}^2=\operatorname{tr}(\boldsymbol{M}\boldsymbol{\Omega}\boldsymbol{M}^\top). \tag{1}$$

The set of $n\times n$ symmetric matrices is $\mathbb{S}^n$, and $\boldsymbol{M}\succcurlyeq\boldsymbol{0}$ means $\boldsymbol{M}$ is positive semidefinite. The operator $\operatorname{sym}(\cdot)$ returns the symmetric part of a matrix:

$$\operatorname{sym}(\boldsymbol{M})=\frac{1}{2}(\boldsymbol{M}+\boldsymbol{M}^\top). \tag{2}$$

At certain points we will use the Kronecker product, denoted by $\otimes$.

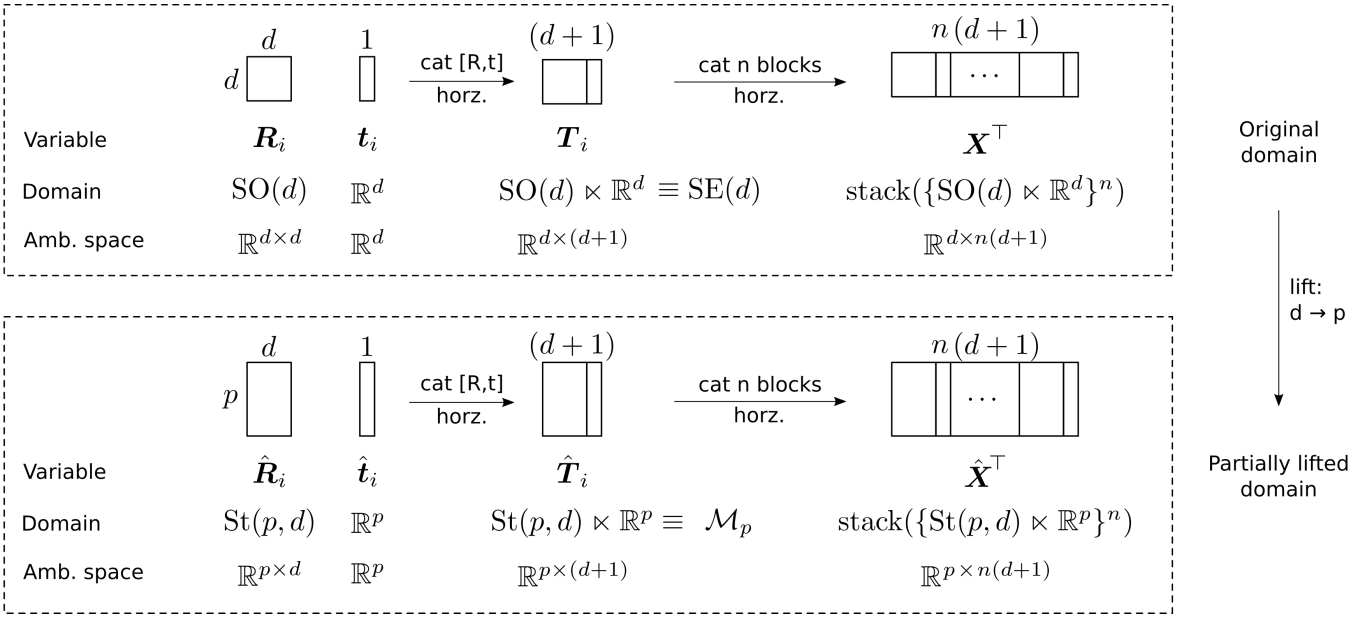


Fig. 1. The geometry of the problem: Original (top, d-dim) and partially lifted (bottom, p-dim).

A. *Group realizations*

Several mathematical groups arise along the paper. We enumerate and define the employed (matrix) characterization in each case.

- The Euclidean (linear) matrix space $\mathbb{R}^{n\times m}$
- Orthogonal group $\mathrm{O}(d) \equiv \{\boldsymbol{R} \in \mathbb{R}^{d\times d} : \boldsymbol{R}^\top \boldsymbol{R} = \boldsymbol{I}_d\}$
- Rotation group $\mathrm{SO}(d) \equiv \{\boldsymbol{R} \in \mathbb{R}^{d\times d} : \boldsymbol{R}^\top \boldsymbol{R} = \boldsymbol{I}_d, \det(\boldsymbol{R}) = +1\}$. Note this is a special case of $\mathrm{O}(d)$ with fixed positive orientation.
- Cartan motion group: $\boldsymbol{K} \ltimes \boldsymbol{V}$, the semidirect product of a compact group $\boldsymbol{K}$ and a linear (non-compact) space $\boldsymbol{V}$.
- Euclidean or isometry group, $\mathrm{E}(d) \equiv \mathrm{O}(d) \ltimes \mathbb{R}^d$.
- Special Euclidean group, $\mathrm{SE}(d) \equiv \mathrm{SO}(d) \ltimes \mathbb{R}^d \subset \mathbb{R}^{d\times(d+1)}$. The group operation is $\boldsymbol{T}_1 \oplus \boldsymbol{T}_2 = \boldsymbol{T}_1 \cdot \tilde{\boldsymbol{T}}_2 = \big[\boldsymbol{R}_1\boldsymbol{R}_2, \boldsymbol{t}_1 + \boldsymbol{R}_1\boldsymbol{t}_2\big]$. The group operation can be expressed in a compact matrix form if the second matrix is augmented with a row of zeros with a final 1 (homogeneous version): $\tilde{\boldsymbol{T}} = \begin{bmatrix} \boldsymbol{T} \\ \boldsymbol{0}_{1\times d} \ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0}_{1\times d} & 1 \end{bmatrix}$.
- Stiefel manifold $St(p,d)$, the set of $p \times d$ matrices which are orthonormal: $St(p,d) \equiv \{\boldsymbol{Y} \in \mathbb{R}^{p\times d} : \boldsymbol{Y}^\top \boldsymbol{Y} = \boldsymbol{I}_d\}$.
- The Cartan motion group of the Stiefel manifold, $St(p,d) \ltimes \mathbb{R}^p \subset \mathbb{R}^{p\times(d+1)}$, with $p \geq d$. This is the essential atom group in the present work. Note that this group is equivalent to $\mathrm{E}(d)$ for $p = d$.

Other useful concepts: product manifold and power manifold, as presented in Manopt tutorial.

II. Connection Laplacian formulation

This section addresses the supplementary material referenced in the homonym section of the main document.

The compact formulation of the MLE objective in sum form is

$$f_{\mathrm{ML}} = \frac{1}{2} \sum_{(i,j)\in E} \|\boldsymbol{T}_j - \boldsymbol{T}_i \tilde{\tilde{\boldsymbol{T}}}_{ij}\|^2_{\boldsymbol{\Omega}_{ij}} = \frac{1}{2} \sum_{(i,j)\in E} \|\boldsymbol{T}_j \boldsymbol{I}_{d+1} - \boldsymbol{T}_i \tilde{\tilde{\boldsymbol{T}}}_{ij}\|^2_{\boldsymbol{\Omega}_{ij}} \tag{3}$$

To write the compact matrix form, we first define a convenient organization of the matrices $\{\boldsymbol{T}_i\}_{i=1}^n$ into the stacked structure $\boldsymbol{X}$:

$$\underset{\substack{[(d+1)\times d]\\(n\times 1)}}{\underline{\boldsymbol{X}}} = \mathrm{stack}(\{\boldsymbol{T}_i\}_{i=1}^n)^\top = \big[\,\boldsymbol{T}_1\big|\ldots\big|\boldsymbol{T}_n\,\big]^\top \in \mathbb{R}^{n(d+1)\times d}.$$

The $d \times (d+1)$ matrices $\boldsymbol{T}_i$ are stacked horizontally first, and then transposed, so $\boldsymbol{X}$ is a *column* block-vector with block size $(d+1)\times d$. Note any particular block $\boldsymbol{T}_i$ can be recovered from $\boldsymbol{X}$ through block indexing, which can be turned to an algebraic expression as

$$\boldsymbol{T}_i = (\boldsymbol{X}_{[i]})^\top = \boldsymbol{X}^\top(\boldsymbol{e}_i \otimes \boldsymbol{I}_{d+1}). \tag{4}$$

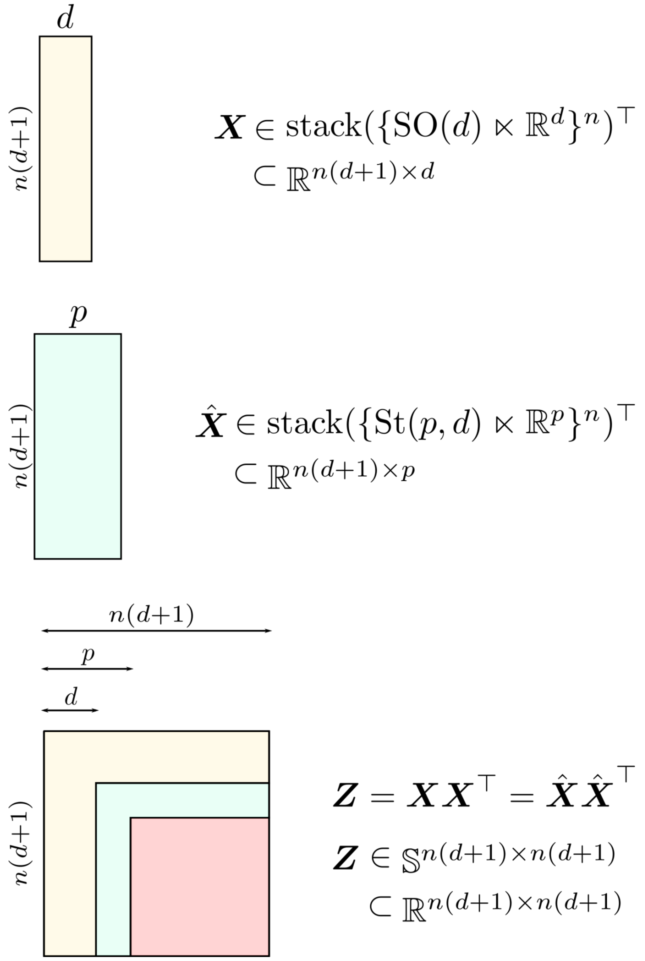

d
$n(d{+}1)$
$\boldsymbol{X} \in \operatorname{stack}(\{\mathrm{SO}(d) \ltimes \mathbb{R}^d\}^n)^\top$
$\subset \mathbb{R}^{n(d+1)\times d}$
p
$n(d{+}1)$
$\hat{\boldsymbol{X}} \in \operatorname{stack}(\{\mathrm{St}(p,d) \ltimes \mathbb{R}^p\}^n)^\top$
$\subset \mathbb{R}^{n(d+1)\times p}$
$n(d{+}1)$
p
d
$n(d{+}1)$
$\boldsymbol{Z} = \boldsymbol{X}\boldsymbol{X}^\top = \hat{\boldsymbol{X}}\hat{\boldsymbol{X}}^\top$
$\boldsymbol{Z} \in \mathbb{S}^{n(d+1)\times n(d+1)}$
$\subset \mathbb{R}^{n(d+1)\times n(d+1)}$

Fig. 2. The hierarchy of lifted relaxations exploited by the presented approach: The original variable has $\operatorname{rank}(\boldsymbol{X}) = d$ (top), whereas a partial relaxation uses $\hat{\boldsymbol{X}}$ with $\operatorname{rank}(\hat{\boldsymbol{X}}) = p \geq d$ (middle). In both cases the variable provides a decomposition of the lifted variable $\boldsymbol{Z}$ in the usual SDP relaxation (bottom). Note the linear ($\hat{\boldsymbol{X}}$) vs. quadratic ($\boldsymbol{Z}$) growth of the problem dimension with n.

Let us take a single term in the sum (3) and rewrite it in a more convenient form:

$$\boldsymbol{T}_j\boldsymbol{I}_{d+1} - \boldsymbol{T}_i\tilde{\tilde{\boldsymbol{T}}}_{ij} = \boldsymbol{X}^\top \underbrace{\Big((\boldsymbol{e}_j \otimes \boldsymbol{I}_{d+1})\boldsymbol{I}_{d+1} - (\boldsymbol{e}_i \otimes \boldsymbol{I}_{d+1})\tilde{\tilde{\boldsymbol{T}}}_{ij}\Big)}_{\boldsymbol{V}_{ij}}. \tag{5}$$

Note that the expression $\boldsymbol{V}_{ij}$ appearing above is a block-vector of dimensions consistent with $\boldsymbol{X}$ populated as

$$\boldsymbol{V}_{[r]} = \begin{cases} -\tilde{\boldsymbol{T}}_{ij} & \text{if } r = i, \\ +\boldsymbol{I}_{d+1} & \text{if } r = j, \\ \boldsymbol{0}_{d+1} & \text{otherwise.} \end{cases} \tag{6}$$

Then, the matrix norm of the sum terms reduces to

$$\begin{aligned} \|\boldsymbol{T}_j\boldsymbol{I}_{d+1} - \boldsymbol{T}_i\tilde{\tilde{\boldsymbol{T}}}_{ij}\|^2_{\boldsymbol{\Omega}_{ij}} &= \|(\boldsymbol{X}^\top\boldsymbol{V}_{ij})\|^2_{\boldsymbol{\Omega}_{ij}} && (7)\\ &= \operatorname{tr}((\boldsymbol{X}^\top\boldsymbol{V}_{ij})\boldsymbol{\Omega}_{ij}(\boldsymbol{X}^\top\boldsymbol{V}_{ij})^\top)) && (8)\\ &= \operatorname{tr}(\boldsymbol{X}^\top(\boldsymbol{V}_{ij}\boldsymbol{\Omega}_{ij}\boldsymbol{V}_{ij}^\top)\boldsymbol{X}). && (9) \end{aligned}$$

The sum along all observations in the sum yields

$$\frac{1}{2}\sum_{(i,j)\in E}\|\boldsymbol{T}_j\boldsymbol{I}_{d+1}-\boldsymbol{T}_i\tilde{\tilde{\boldsymbol{T}}}_{ij}\|^2_{\boldsymbol{\Omega}_{ij}}=\frac{1}{2}\sum_{(i,j)\in E}\operatorname{tr}(\boldsymbol{X}^\top(\boldsymbol{V}_{ij}\boldsymbol{\Omega}_{ij}\boldsymbol{V}_{ij}^\top)\boldsymbol{X}) \tag{10}$$

$$=\frac{1}{2}\operatorname{tr}\left(\boldsymbol{X}^\top\underbrace{\left(\sum_{(i,j)\in E}\boldsymbol{V}_{ij}\boldsymbol{\Omega}_{ij}\boldsymbol{V}_{ij}^\top\right)}_{\boldsymbol{Q}}\boldsymbol{X}\right) \tag{11}$$

The matrix sum above can be written in a more compact form if we concatenate the block-column vectors $\boldsymbol{V}_{ij}$ horizontally as

$$\underbrace{\boldsymbol{A}}_{\substack{[(d+1)\times(d+1)]\\(n\times m)}}=\operatorname{stack}(\{\boldsymbol{V}_{ij}\}_{(i,j)\in E})=\left[\,\boldsymbol{V}_{e_1}\middle|\ldots\middle|\boldsymbol{V}_{e_m}\,\right], \tag{12}$$

and the weight matrices are put in a block-diagonal matrix as

$$\underbrace{\boldsymbol{\Omega}}_{\substack{[(d+1)\times(d+1)]\\(m\times m)}}=\operatorname{blkdiag}(\{\boldsymbol{\Omega}_{ij}\}_{(i,j)\in E})=\operatorname{blkdiag}(\boldsymbol{\Omega}_{e_1},\ldots,\boldsymbol{\Omega}_{e_m}). \tag{13}$$

With these definitions, it is easy to see that

$$\underbrace{\boldsymbol{Q}}_{\substack{[(d+1)\times(d+1)]\\(n\times n)}}=\sum_{(i,j)\in E}\boldsymbol{V}_{ij}\boldsymbol{\Omega}_{ij}\boldsymbol{V}_{ij}^\top=\underbrace{\boldsymbol{A}}_{\substack{[(d+1)\times(d+1)]\\(n\times m)}}\underbrace{\boldsymbol{\Omega}}_{\substack{[(d+1)\times(d+1)]\\(m\times m)}}\underbrace{\boldsymbol{A}^\top}_{\substack{[(d+1)\times(d+1)]\\(m\times n)}}. \tag{14}$$

III. Lagrangian relaxation: dual SDP problem

This section addresses the supplementary material referenced in Section *III.A) Forming the SDP relaxation* in the main document.

The *orthogonal relaxation* of the MLE problem is written with explicit constraints as

$$f^\star_O=\min_{\boldsymbol{X}}\frac{1}{2}\operatorname{tr}(\boldsymbol{X}^\top\boldsymbol{Q}\boldsymbol{X}) \tag{15}$$

$$\text{s.t. }\boldsymbol{R}_i^\top\boldsymbol{R}_i=\boldsymbol{I}_d,\quad\forall i=1,\ldots,n. \tag{16}$$

The Lagrangian function corresponding to (15) is created by adding a penalization term for each (symmetric) orthonormality constraint (16),

$$\text{penalization}(\boldsymbol{R}_i^\top\boldsymbol{R}_i=\boldsymbol{I}_d)\equiv\frac{1}{2}\operatorname{tr}(\boldsymbol{\Lambda}_i(\boldsymbol{I}_d-\boldsymbol{R}_i^\top\boldsymbol{R}_i))=\frac{1}{2}\operatorname{tr}(\boldsymbol{\Lambda}_i)-\frac{1}{2}\operatorname{tr}(\boldsymbol{R}_i\boldsymbol{\Lambda}_i\boldsymbol{R}_i^\top), \tag{17}$$

where $\boldsymbol{\Lambda}_i\in\mathbb{S}^d$ is a symmetric matrix of Lagrange multipliers. We add the $\frac{1}{2}$ factor for conveniency regarding the ultimate expression. This penalization term can be equivalently written in term of the matrices $\boldsymbol{T}_i$ as

$$\operatorname{tr}(\boldsymbol{\Lambda}_i)-\operatorname{tr}(\boldsymbol{R}_i\boldsymbol{\Lambda}_i\boldsymbol{R}_i^\top)=\operatorname{tr}(\tilde{\boldsymbol{\Lambda}}_i)-\operatorname{tr}(\boldsymbol{T}_i\tilde{\boldsymbol{\Lambda}}_i\boldsymbol{T}_i^\top) \tag{18}$$

if we define the *homogeneous version* of $\boldsymbol{\Lambda}$ as

$$\tilde{\boldsymbol{\Lambda}}_i=\begin{bmatrix}\boldsymbol{\Lambda}_i & \boldsymbol{0}_{d\times 1}\\ \boldsymbol{0}_{1\times d} & 0_{1\times 1}\end{bmatrix}. \tag{19}$$

Finally, using again the relation (4) the sum of all the penalization terms for each i yields

$$\frac{1}{2}\sum_{i=1}^{n}\operatorname{tr}(\tilde{\boldsymbol{\Lambda}}_i)-\operatorname{tr}(\boldsymbol{T}_i\tilde{\boldsymbol{\Lambda}}_i\boldsymbol{T}_i^\top)=\frac{1}{2}\operatorname{tr}(\boldsymbol{\Lambda})-\frac{1}{2}\operatorname{tr}(\boldsymbol{X}^\top\boldsymbol{\Lambda}\boldsymbol{X}), \tag{20}$$

where we define the convenient direct sum matrix $\boldsymbol{\Lambda}$ as

$$\boldsymbol{\Lambda}=\operatorname{blkdiag}(\tilde{\boldsymbol{\Lambda}}_1,\ldots,\tilde{\boldsymbol{\Lambda}}_n)=\operatorname{blkdiag}(\boldsymbol{\Lambda}_1,0,\ldots,\boldsymbol{\Lambda}_n,0). \tag{21}$$

With the penalization above, the Lagrangian function corresponding to (15) is written in compact form as

$$\mathrm{L}(\boldsymbol{X},\boldsymbol{\Lambda})=\frac{1}{2}\operatorname{tr}(\boldsymbol{X}^\top\boldsymbol{Q}\boldsymbol{X})+\frac{1}{2}\operatorname{tr}(\boldsymbol{\Lambda})-\frac{1}{2}\operatorname{tr}(\boldsymbol{X}^\top\boldsymbol{\Lambda}\boldsymbol{X}) \tag{22}$$

$$=\frac{1}{2}\operatorname{tr}(\boldsymbol{\Lambda})+\frac{1}{2}\operatorname{tr}(\boldsymbol{X}^\top(\boldsymbol{Q}-\boldsymbol{\Lambda})\boldsymbol{X}), \tag{23}$$

The Lagrangian provides an unconstrained *relaxation* of the original problem

$$d(\boldsymbol{\Lambda}) = \min_{\boldsymbol{X}} \mathrm{L}(\boldsymbol{X}, \boldsymbol{\Lambda}) \tag{24}$$

whose optimal value $d(\boldsymbol{\Lambda}) \leq f^\star$ is called the *dual function*. Since the term $\mathrm{tr}(\boldsymbol{X}^\top(\boldsymbol{Q} - \boldsymbol{\Lambda})\boldsymbol{X})$ is a homogeneous quadratic form (wrt $\boldsymbol{X}$), its minimum value 0 is attained for

$$\begin{aligned}\boldsymbol{X}^\star(\boldsymbol{\Lambda}) &= \{\boldsymbol{X} : \mathrm{tr}(\boldsymbol{X}^\top(\boldsymbol{Q} - \boldsymbol{\Lambda})\boldsymbol{X}) = 0\} \\ &= \{\boldsymbol{X} : (\boldsymbol{Q} - \boldsymbol{\Lambda})\boldsymbol{X} = \boldsymbol{0}_{n(d+1)\times d}\},\end{aligned} \tag{25}$$

if the penalized matrix is semidefinite positive, $\boldsymbol{Q} - \boldsymbol{\Lambda} \succcurlyeq \boldsymbol{0}$. Otherwise this term is unbounded below (its minimum value is $-\infty$). Thus, the optimum value of the Lagrangian (24) is

$$d(\boldsymbol{\Lambda}) = \begin{cases} \frac{1}{2}\mathrm{tr}(\boldsymbol{\Lambda}) & \text{if } \boldsymbol{Q} - \boldsymbol{\Lambda} \succcurlyeq \boldsymbol{0}, \\ -\infty & \text{otherwise.} \end{cases} \tag{26}$$

The *dual problem* seeks the tightest relaxation by maximizing the lower bound $d(\boldsymbol{\Lambda})$ wrt $\boldsymbol{\Lambda}$ as $d^\star = \max_{\boldsymbol{\Lambda}} d(\boldsymbol{\Lambda})$. In view of the expression for the dual objective (26), the search for the maximum can be safely restricted to values of $\boldsymbol{\Lambda}$ for which $\boldsymbol{Q} - \boldsymbol{\Lambda} \succcurlyeq \boldsymbol{0}$, so the dual problem is a *Semidefinite Program* (SDP):

$$d^\star = \max_{\boldsymbol{\Lambda}} \frac{1}{2}\mathrm{tr}(\boldsymbol{\Lambda}), \quad \text{s.t. } \boldsymbol{Q} - \boldsymbol{\Lambda} \succcurlyeq \boldsymbol{0}. \tag{27}$$

In the context of duality theory [1], it is straightforward to obtain the *primal* version of this same SDP problem as

$$p^\star = \min_{\boldsymbol{Z}\in\mathbb{S}^{n(d+1)}} \frac{1}{2}\mathrm{tr}(\boldsymbol{Q}\boldsymbol{Z}) \tag{28}$$

$$\text{s.t. } \boldsymbol{Z} = \begin{bmatrix} \boldsymbol{I}_{d,*} & * & \cdots & * \\ * & \boldsymbol{I}_{d,*} & \cdots & * \\ \vdots & \vdots & \ddots & \vdots \\ * & * & \cdots & \boldsymbol{I}_{d,*} \end{bmatrix} \succcurlyeq \boldsymbol{0}, \tag{29}$$

where $*$ stands for unconstrained values and

$$\boldsymbol{I}_{d,*} \equiv \begin{bmatrix} \boldsymbol{I}_d & *_{d\times 1} \\ *_{1\times d} & *_{1\times 1} \end{bmatrix}. \tag{30}$$

Equivalently, the constraint on the primal SDP variable $\boldsymbol{Z}$ can be seen as n linear constraints of the form

$$\boldsymbol{Z}_{[i,i](1:d,1:d)} = \boldsymbol{I}_d, \quad \forall i = 1, \ldots, n. \tag{31}$$

IV. Global optimality test in SDP relaxation

This section addresses the supplementary material associated to Footnote #13 in the main document.

Using the results in (25) characterizing the optimal solution $\hat{\boldsymbol{X}}^\star$ in terms of a given dual variable $\boldsymbol{\Lambda}$ it is easy to see that there is a linear relation between $\hat{\boldsymbol{X}}^\star$ and $\boldsymbol{\Lambda}^\star$ for the case with tight relaxation:

$$(\boldsymbol{Q} - \boldsymbol{\Lambda}^\star)\hat{\boldsymbol{X}}^\star = \boldsymbol{0}_{n(d+1)\times d}. \tag{32}$$

For a known $\hat{\boldsymbol{X}}^\star$, this provides a quick way to compute the potential dual solution $\boldsymbol{\Lambda}^\star$ and then check the feasibility of this in the dual problem as a way to assert global optimality of the primal-dual pair $(\hat{\boldsymbol{X}}^\star, \boldsymbol{\Lambda}^\star)$.

The linear equation on $\boldsymbol{\Lambda}$ following from (32) reads

$$\boldsymbol{\Lambda}\hat{\boldsymbol{X}}^\star = \boldsymbol{Q}\hat{\boldsymbol{X}}^\star. \tag{33}$$

Because of the special block-diagonal structure of $\boldsymbol{\Lambda}$, each $\hat{\boldsymbol{R}}$-block in the i-th block-row of the equation reads

$$\boldsymbol{\Lambda}_i\hat{\boldsymbol{R}}_i^\top = \left(\boldsymbol{Q}\hat{\boldsymbol{X}}^\star\right)_{[i](\hat{\boldsymbol{R}})}^\top. \tag{34}$$

That is, in the right hand side of the equation, following our indexing convention, we grab the i-th $(d+1)\times p$-block in $\boldsymbol{Q}\hat{\boldsymbol{X}}^\star$, then keep only the transpose of the $(p \times d)$-subblock corresponding to the $\hat{\boldsymbol{R}}$ part. Then exploiting the orthonormality of the rows of $\hat{\boldsymbol{R}}_i \in St(p,d)$, we obtain the solution $\boldsymbol{\Lambda}_i$ as

$$\boldsymbol{\Lambda}_i = \left(\boldsymbol{Q}\hat{\boldsymbol{X}}^\star\right)_{[i](\hat{\boldsymbol{R}})}^\top \hat{\boldsymbol{R}}_i. \tag{35}$$

Note that as further developed in previous work [2], the global optimality of the estimated $\boldsymbol{\Lambda}$ requires:

- The computed solution fulfills the complete linear system $\boldsymbol{\Lambda}\hat{\boldsymbol{X}}^{\star} = \boldsymbol{Q}\hat{\boldsymbol{X}}^{\star}$.
- All the blocks $\boldsymbol{\Lambda}_i$ are symmetric.
- The PSD constraint of the SDP relaxation stands: $\boldsymbol{Q} - \boldsymbol{\Lambda}^{\star} \succcurlyeq \boldsymbol{0}$.

V. Riemannian geometry

This section addresses the supplementary material referenced in Section *IV.B.1) Solving the Riemannian problem* in the main document.

Consider the Riemannian optimization problem

$$f^{\star} = \min_{\hat{\boldsymbol{X}} \in \mathrm{stack}(\mathcal{M}_p^n)^{\top}} \frac{1}{2} \operatorname{tr}(\hat{\boldsymbol{X}}^{\top} \boldsymbol{Q} \hat{\boldsymbol{X}}), \quad \mathcal{M}_p \equiv St(p,d) \ltimes \mathbb{R}^p \subset \mathbb{R}^{p \times (d+1)}. \tag{36}$$

Note the manifold considered as the optimization domain is the stacking of the product of n instances of the $\mathcal{M}_p$ manifold (also sometimes referred to as a power manifold).

The objective $f(\hat{\boldsymbol{X}})$ is a simple *positive semidefinite* quadratic function if considered as a function on the ambient Euclidean space $\mathbb{R}^{n(d+1)\times p}$:

$$f : \mathbb{R}^{n(d+1)\times p} \mapsto \mathbb{R}_{+}, \tag{37}$$

$$f(\hat{\boldsymbol{X}}) = \frac{1}{2} \operatorname{tr}(\hat{\boldsymbol{X}}^{\top} \boldsymbol{Q} \hat{\boldsymbol{X}}). \tag{38}$$

The Euclidean gradient and Hessian-vector product of this function, using the concept of Fréchet derivative presented in Appendix I, are identified from a Taylor expansion[1] of the objective:

$$f(\hat{\boldsymbol{X}} + t\boldsymbol{U}) = \frac{1}{2} \operatorname{tr}((\hat{\boldsymbol{X}} + t\boldsymbol{U})^{\top} \boldsymbol{Q} (\hat{\boldsymbol{X}} + t\boldsymbol{U})) \tag{39}$$

$$= \frac{1}{2} \operatorname{tr}(\hat{\boldsymbol{X}}^{\top} \boldsymbol{Q} \hat{\boldsymbol{X}}) + t \operatorname{tr}(\hat{\boldsymbol{X}}^{\top} \boldsymbol{Q} \boldsymbol{U}) + \frac{1}{2} t^2 \operatorname{tr}(\boldsymbol{U}^{\top} \boldsymbol{Q} \boldsymbol{U}) \tag{40}$$

$$= f(\hat{\boldsymbol{X}}) + t \langle \boldsymbol{Q}\hat{\boldsymbol{X}}, \boldsymbol{U} \rangle + \frac{1}{2} t^2 \langle \boldsymbol{Q}\boldsymbol{U}, \boldsymbol{U} \rangle \tag{41}$$

$$= f(\hat{\boldsymbol{X}}) + t \langle \nabla f(\hat{\boldsymbol{X}}), \boldsymbol{U} \rangle + \frac{1}{2} t^2 \langle \nabla f(\hat{\boldsymbol{X}})[\boldsymbol{U}], \boldsymbol{U} \rangle \tag{42}$$

It follows then that the Euclidean operators are

$$\nabla f(\hat{\boldsymbol{X}}) = \boldsymbol{Q}\hat{\boldsymbol{X}}, \tag{43}$$

$$\nabla^2 f(\hat{\boldsymbol{X}})[\boldsymbol{U}] = \boldsymbol{Q}\boldsymbol{U}. \tag{44}$$

A fundamental tool to connect the Euclidean calculus with its Riemannian counterpart for the more restricted function

$$f : \mathrm{stack}(\mathcal{M}_p^n)^{\top} \mapsto \mathbb{R}_{+}, \tag{45}$$

$$f(\hat{\boldsymbol{X}}) = \frac{1}{2} \operatorname{tr}(\hat{\boldsymbol{X}}^{\top} \boldsymbol{Q} \hat{\boldsymbol{X}}). \tag{46}$$

is the availability of an *orthogonal projection operator* from the ambient space $\mathbb{R}^{n(d+1)\times p}$ to the tangent space $T_{\hat{\boldsymbol{X}}}(\mathrm{stack}(\mathcal{M}_p^n)^{\top})$ at a point $\hat{\boldsymbol{X}} \in \mathrm{stack}(\mathcal{M}_p^n)^{\top}$,

$$\mathrm{proj}_{\hat{\boldsymbol{X}}} : \mathbb{R}^{n(d+1)\times p} \mapsto T_{\hat{\boldsymbol{X}}} \mathcal{M}_p^n. \tag{47}$$

Because of the particular product-like structure of the manifold, this projection is readily expressed in terms of the projection operators of its inner components:

$$\mathrm{proj}_{\hat{\boldsymbol{X}}} : \mathbb{R}^{n(d+1)\times p} \mapsto T_{\hat{\boldsymbol{X}}}(\mathrm{stack}(\mathcal{M}_p^n)^{\top}) \tag{48}$$

$$\mathrm{proj}_{\hat{\boldsymbol{X}}}(\boldsymbol{U}) = \Big[\mathrm{proj}_{\hat{\boldsymbol{T}}_1}(\boldsymbol{U}_{[1]}) \Big| \dots \Big| \mathrm{proj}_{\hat{\boldsymbol{T}}_n}(\boldsymbol{U}_{[n]}) \Big]^{\top} \tag{49}$$

$$\mathrm{proj}_{\hat{\boldsymbol{T}}_i} : \mathbb{R}^{p\times(d+1)} \mapsto T_{\hat{\boldsymbol{T}}_i}(\mathcal{M}_p) \tag{50}$$

$$\mathrm{proj}_{\hat{\boldsymbol{T}}_i}(\boldsymbol{U}_{[i]}) = \big[\mathrm{proj}_{\hat{\boldsymbol{R}}_i}(\boldsymbol{U}_{[i](\boldsymbol{R})}), \mathrm{proj}_{\hat{\boldsymbol{t}}_i}(\boldsymbol{U}_{[i](\boldsymbol{t})}) \big] \tag{51}$$

$$\mathrm{proj}_{\hat{\boldsymbol{R}}_i} : \mathbb{R}^{p\times d} \mapsto T_{\hat{\boldsymbol{R}}_i}(St(p,d)) \tag{52}$$

$$\mathrm{proj}_{\hat{\boldsymbol{R}}_i}(*) = \mathrm{Id}(*) - \hat{\boldsymbol{R}}_i \, \mathrm{sym}(\hat{\boldsymbol{R}}_i^{\top} *) \tag{53}$$

$$\mathrm{proj}_{\hat{\boldsymbol{t}}_i} : \mathbb{R}^{p} \mapsto T_{\hat{\boldsymbol{t}}_i}(\mathbb{R}^p) \tag{54}$$

$$\mathrm{proj}_{\hat{\boldsymbol{t}}_i}(*) = \mathrm{Id}(*). \tag{55}$$

[1] See the same trick employed here.

Note that the only non-trivial projection is that for the Stiefel manifold, for which extensive insight is available in [3].

Using the operators above, it is straightforward to obtain the Riemannian gradient and Hessian operators (from Riemannian geometry theory [4]) as

$$\operatorname{grad} f(\cdot) : \operatorname{stack}(\mathcal{M}_p^n)^\top \mapsto T_{\hat{\boldsymbol{X}}}(\operatorname{stack}(\mathcal{M}_p^n)^\top), \tag{56}$$

$$\operatorname{grad} f(\hat{\boldsymbol{X}}) = \operatorname{proj}_{\hat{\boldsymbol{X}}}(\nabla f(\hat{\boldsymbol{X}})), \tag{57}$$

$$\operatorname{Hess} f(\cdot)[\cdot] : \operatorname{stack}(\mathcal{M}_p^n)^\top \times T_{\hat{\boldsymbol{X}}}(\operatorname{stack}(\mathcal{M}_p^n)^\top) \mapsto T_{\hat{\boldsymbol{X}}}(\operatorname{stack}(\mathcal{M}_p^n)^\top), \tag{58}$$

$$\operatorname{Hess} f(\hat{\boldsymbol{X}})[\boldsymbol{U}] = \operatorname{proj}_{\hat{\boldsymbol{X}}}(\mathrm{D}[\operatorname{grad} f(\hat{\boldsymbol{X}})][\boldsymbol{U}]). \tag{59}$$

The value of the Riemannian gradient follows immediately by application of the projection operator, whereas for the Riemannian Hessian we need to compute the directional (Fréchet) derivative of the Riemannian gradient. The (block-wise) expression for this derivative is:

$$\mathrm{D}[\operatorname{grad} f(\hat{\boldsymbol{X}})][\cdot] : T_{\hat{\boldsymbol{X}}}(\operatorname{stack}(\mathcal{M}_p^n)^\top) \mapsto \mathbb{R}^{p\times(d+1)}, \tag{60}$$

$$\left(\mathrm{D}[\operatorname{grad} f(\hat{\boldsymbol{X}})][\boldsymbol{U}]\right)_{[i](\boldsymbol{R})} = \left(\nabla^2 f(\hat{\boldsymbol{X}})[\boldsymbol{U}]\right)_{[i](\boldsymbol{R})} - (\boldsymbol{U})_{[i](\boldsymbol{R})} \operatorname{sym}\left(\hat{\boldsymbol{R}}^\top \left(\nabla f(\hat{\boldsymbol{X}})\right)_{[i](\boldsymbol{R})}\right), \tag{61}$$

$$\left(\mathrm{D}[\operatorname{grad} f(\hat{\boldsymbol{X}})][\boldsymbol{U}]\right)_{[i](\boldsymbol{t})} = \left(\nabla^2 f(\hat{\boldsymbol{X}})[\boldsymbol{U}]\right)_{[i](\boldsymbol{t})}. \tag{62}$$

Again, the only non-trivial results stem from the Stiefel component [3], [5], [6].

VI. Proofs on proposed Riemannian Preconditioner

This section addresses the supplementary material associated to Footnote #10 in the main document.

The Riemannian preconditioner at a specific point $\hat{\boldsymbol{X}}$ should be a linear, symmetric, positive definite operator defined from and onto the tangent space of the manifold,

$$\operatorname{Precon} : T_{\hat{\boldsymbol{X}}}(\operatorname{stack}(\mathcal{M}_p^n)^\top) \mapsto T_{\hat{\boldsymbol{X}}}(\operatorname{stack}(\mathcal{M}_p^n)^\top). \tag{63}$$

We prove next that these conditions stand for the proposed Riemannian Preconditioner

$$\operatorname{Precon}(\hat{\boldsymbol{X}})[\boldsymbol{U}] = \operatorname{proj}_{\hat{\boldsymbol{X}}}(\boldsymbol{Q}^{-1}\boldsymbol{U}). \tag{64}$$

a) Domain and range: It is clear that the domain constraint is fulfilled by definition, as long as the provided $\boldsymbol{U}$ belongs to the tangent space $\boldsymbol{U} \in T_{\hat{\boldsymbol{X}}}(\operatorname{stack}(\mathcal{M}_p^n)^\top)$. In practice this is sometimes assured by simply composing with the projection at the input, $\boldsymbol{U} \leftarrow \operatorname{proj}_{\hat{\boldsymbol{X}}}(\boldsymbol{U})$.

The range condition is also assured by the projection operator, so $\operatorname{Precon}(\hat{\boldsymbol{X}})[\boldsymbol{U}] \in T_{\hat{\boldsymbol{X}}}(\operatorname{stack}(\mathcal{M}_p^n)^\top)$ as expected.

b) Linearity, symmetry and positive semidefiniteness: In order to prove these conditions, it is best to turn the preconditioner into its equivalent matrix formulation. This is given by the matrix form of the projection:

$$\operatorname{proj}_{\hat{\boldsymbol{X}}}(\boldsymbol{U}) = \boldsymbol{P}_{\hat{\boldsymbol{X}}} \operatorname{vec}(\boldsymbol{U}^\top), \tag{65}$$

$$\operatorname{vec}(\operatorname{Precon}(\hat{\boldsymbol{X}})[\boldsymbol{U}]^\top) = \boldsymbol{P}_{\hat{\boldsymbol{X}}}(\boldsymbol{Q}^{-1} \otimes \boldsymbol{I}_p)\boldsymbol{P}_{\hat{\boldsymbol{X}}} \operatorname{vec}(\boldsymbol{U}^\top). \tag{66}$$

The linearity of the operator follows from its matrix formulation. The orthogonal projector is self-adjoint and, as such, it is also symmetric. As a result, $\boldsymbol{P}_{\hat{\boldsymbol{X}}} = \boldsymbol{P}_{\hat{\boldsymbol{X}}}^\top$. Since $\boldsymbol{Q}$ is also symmetric, the matrix operator above is clearly symmetric. It can be shown that the orthogonal projector $\operatorname{proj}_{\hat{\boldsymbol{X}}}(\cdot)$ is positive semidefinite. As a result, so is its matrix representation $\boldsymbol{P}_{\hat{\boldsymbol{X}}}$ and since $\boldsymbol{Q}$ was also positive semidefinite, so must be the product of all the matrices above.

VII. Cholesky decomposition of the Connection Laplacian

This section addresses the supplementary material associated to Footnote #11 in the main document.

As the usual Laplacian matrix, the Connection Laplacian defined in the paper is degenerate. Its nullspace is defined by the vector

$$\boldsymbol{n} = \boldsymbol{1}_{n\times 1} \otimes \begin{bmatrix} \boldsymbol{0}_{d\times 1} \\ 1 \end{bmatrix}. \tag{67}$$

This fact is tightly related to the observability issue stemming from relative observations only, which makes the translation observable only up to a global offset $\boldsymbol{t}_0$. As a result, any solution $\boldsymbol{X}^\star$ to the SE(d)-synchronization problem stands for a family of solutions

$$\boldsymbol{X}^\star + \boldsymbol{n}\boldsymbol{c}^\top, \quad \boldsymbol{c} \in \mathbb{R}^d. \tag{68}$$

In the context of solving our Riemannian problem, this fact becomes an issue only when solving the linear system for the preconditioner:

$$\boldsymbol{Q}\boldsymbol{U} = \boldsymbol{B}. \tag{69}$$

Note that due to the concrete degeneracy of $\boldsymbol{Q}$, the solution to this underdetermined system is

$$\boldsymbol{U} = \boldsymbol{U}_0 + \boldsymbol{n}\boldsymbol{c}^\top, \tag{70}$$

where $\boldsymbol{U}_0$ is *any* solution to the linear system (69) and the second component parameterizes the family of infinite solutions.

Since from the point of view of our preconditioner we are interested in any solution (the RTR solver is agnostic to which one) we can apply the usual *anchoring* operation: We impose $\boldsymbol{U}_{n(d+1),:} = 0$ (which is always possible in view of the family of solutions (70)) and drop the last equation. This is equivalent to dropping the last column and row in the system matrix $\boldsymbol{Q}$, yielding the anchored matrix $\breve{\boldsymbol{Q}}$ which is positive definite. The remaining linear system

$$\breve{\boldsymbol{Q}}\breve{\boldsymbol{U}} = \breve{\boldsymbol{B}} \tag{71}$$

has a unique solution $\breve{\boldsymbol{U}}$. Padding this solution $\breve{\boldsymbol{U}}$ with the fixed last row of zeros yields the desired solution to the original system.

As we point in the paper, in order to save an important amount of computational effort we will exploit a Cholesky decomposition to solve the multiple instances of the linear system (69). Since Cholesky decomposition only exists for *non-degenerate*, positive definite matrices, we will compute the Cholesky decomposition of the anchored matrix, $\breve{\boldsymbol{Q}} = \boldsymbol{L}\boldsymbol{L}^\top$, and use this in the resolution of the anchored linear system (71).

VIII. Testing global optimality

This section reviews the necessary steps to verify global optimality of a given candidate solution for the Riemannian optimization problem.

In order to test the global optimality of the critical point $\hat{\boldsymbol{X}}$ found by RTR in (36), we follow the same steps as in [2]. Namely, we exploit complementary slackness to obtain the potential dual solution $\boldsymbol{\Lambda}$ from the computed $\hat{\boldsymbol{X}}$, which results in a simple closed-form expression

$$\underbrace{\boldsymbol{\Lambda}_i}_{d\times d} = (\boldsymbol{Q}\hat{\boldsymbol{X}})_{[i](1:d,1:p)}\hat{\boldsymbol{R}}_i. \tag{72}$$

Then, the global optimality of the critical point $\hat{\boldsymbol{X}}$ depends on whether the computed $\boldsymbol{\Lambda}$ is a feasible point of the dual SDP problem or not. Since the candidate is a critical point, it must automatically fulfill $\boldsymbol{\Lambda}_i \in \mathbb{S}^d$. To check the Positive Semidefiniteness (PSD) constraint, that is, that $\boldsymbol{Q} - \boldsymbol{\Lambda} \succcurlyeq \boldsymbol{0}$, we compute the most negative eigenvalue and its eigenvector:

$$(\lambda_{\text{min}}, \boldsymbol{v}) \leftarrow \min \lambda(\boldsymbol{Q} - \boldsymbol{\Lambda}). \tag{73}$$

If $\lambda_{\text{min}} \approx 0$, we can certify we found the solution to the convex SDP problem. Otherwise, the Riemannian optimization converged into a suboptimal point and we need to increase the rank p of the partially lifted search space.

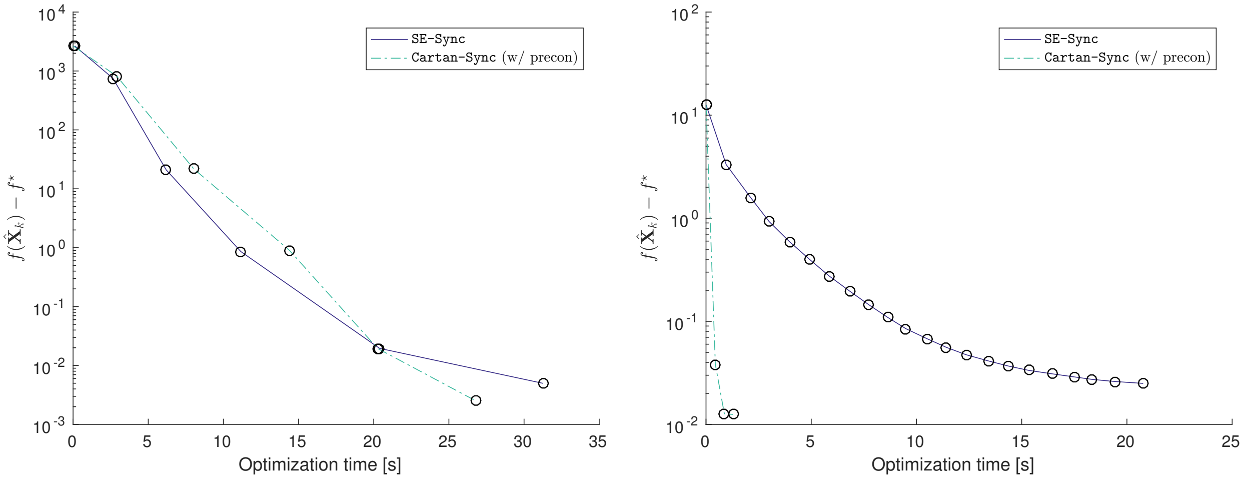
IX. Escaping suboptimal minima

If a step in the Riemannian staircase converged to a suboptimal point, we apply the same trick as Boumal in [5] to escape from it: We lift the current estimate $\hat{\boldsymbol{X}} \in \mathcal{M}_p$ to $\hat{\boldsymbol{X}}_+ \in \mathcal{M}_{p+1}$ by filling the extra terms with zeros, and perform line search along a escape direction $\boldsymbol{U}_{\text{Esc}}$. As in [5], a escape direction is provided by the eigenvector $\boldsymbol{v}$ corresponding to a negative eigenvalue in $\boldsymbol{Q} - \boldsymbol{\Lambda}$, which is already available from the optimality test (73):

$$\boldsymbol{U}_{\text{Esc}} = \begin{bmatrix} \boldsymbol{0}_{n(d+1)\times p} & \boldsymbol{v} \end{bmatrix}. \tag{74}$$

X. Some extra experimental results

Due to lack of space, we moved some experimental results from the main document to here. Concretely, Fig. 3 compares convergence of the absolute objective error w.r.t. time in two exemplary cases (`rim` and `kitti_02`).



(a) `rim` dataset

(b) `kitti_02` dataset

Fig. 3. Convergence of `SE-Sync` and `Cartan-Sync` (with preconditioning) in (a) case of similar performance, (b) case where preconditioning clearly excels.

Appendix I
Some calculus notions and the Fréchet derivative

The most fundamental concept in multivariable calculus for a function $f : U \subset V \mapsto W$ between Banach spaces[2] is the directional derivative or *Fréchet derivative* $\mathrm{D}\, f(\boldsymbol{x})[\boldsymbol{u}]$ of a function $f(\boldsymbol{x})$ at the point $\boldsymbol{x} \in U$ along a direction $\boldsymbol{u} \in V$:

$$\mathrm{D}\, f(\boldsymbol{x})[\cdot] : V \mapsto W. \tag{75}$$

This can be seen as the rate of change of the function $f(\boldsymbol{x})$ when moving from the point $\boldsymbol{x}$ with a velocity specified by $\boldsymbol{u}$.

The directional derivative is tightly related to the *gradient* ∇f of the function $f(\boldsymbol{x})$, since it is defined as the *unique* vector field whose inner product with any vector $\boldsymbol{u}$ at each point $\boldsymbol{x}$ is the directional derivative of $f(\boldsymbol{x})$ at $\boldsymbol{x}$ along $\boldsymbol{u}$:

$$\langle \nabla f(\boldsymbol{x}), \boldsymbol{u} \rangle = \mathrm{D}\, f(\boldsymbol{x})[\boldsymbol{u}]. \tag{76}$$

Thus, the directional derivative at a fixed point $\boldsymbol{x}$ can be seen as a linear operator on the direction $\boldsymbol{u}$:

$$\mathrm{D}\, f(\boldsymbol{x})[\cdot] : \mathbb{R}^n \mapsto \mathbb{R} \tag{77}$$

$$\mathrm{D}\, f(\boldsymbol{x})[\boldsymbol{u}] = \langle \nabla f(\boldsymbol{x}), \boldsymbol{u} \rangle \tag{78}$$

The second directional derivative of $f(\boldsymbol{x})$ measures, at a point $\boldsymbol{x}$, how the rate of change along a direction $\boldsymbol{u}$ is itself changing in direction $\boldsymbol{v}$:

$$\mathrm{D}^2\, f(\boldsymbol{x})[\boldsymbol{u}][\boldsymbol{v}] : V \times V \mapsto W. \tag{79}$$

This is connected to the *curvature* of the function. Similarly to the gradient, the Hessian can be seen as a linear operator that encodes the *gradient* of the directional derivative at a point and direction, so that it becomes the unique operator so that its inner product with any direction $\boldsymbol{v}$ provides the second directional derivative of $f(\boldsymbol{x})$ at $\boldsymbol{x}$ along $\boldsymbol{u}$ and $\boldsymbol{v}$:

$$\langle \boldsymbol{H} f(\boldsymbol{x})[\boldsymbol{u}], \boldsymbol{v} \rangle = \mathrm{D}^2\, f(\boldsymbol{x})[\boldsymbol{u}][\boldsymbol{v}]. \tag{80}$$

Under this perspective, the Hessian must be defined as a function of two parameters $\boldsymbol{x}$ and $\boldsymbol{u}$.

References

[1] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge University Press, 2004.

[2] J. Briales and J. González-Jiménez, “Fast Global Optimality Verification in 3D SLAM,” in *Int. Conf. Intell. Robot. Syst.*, IEEE/RSJ, 2016.

[3] A. Edelman, T. A. Arias, and S. T. Smith, “The geometry of algorithms with orthogonality constraints,” *SIAM J. Matrix Anal. Appl.*, 1998.

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

[5] N. Boumal, “A Riemannian low-rank method for optimization over semidefinite matrices with block-diagonal constraints,” *arXiv*, 2015.

[6] D. M. Rosen, L. Carlone, A. S. Bandeira, and J. J. Leonard, “SE-Sync: A Certifiably Correct Algorithm for Synchronization over the Special Euclidean Group,” *arXiv Prepr. arXiv1612.07386*, pp. 1–49, 2016.

[2] This extends usual calculus to vector-valued functions defined on vector spaces.