Nonlinear Graph Sparsification for SLAM

Mladen Mazuran   Gian Diego Tipaldi   Luciano Spinello   Wolfram Burgard

Department of Computer Science, University of Freiburg, Germany

{mazuran, tipaldi, spinello, burgard}@informatik.uni-freiburg.de

Abstract—In this paper we present a novel framework for nonlinear graph sparsification in the context of simultaneous localization and mapping. Our approach is formulated as a convex minimization problem, where we select the set of nonlinear measurements that best approximate the original distribution. In contrast to previous algorithms, our method does not require a global linearization point and can be used with any nonlinear measurement function. Experiments performed on several publicly available datasets demonstrate that our method outperforms the state of the art with respect to the Kullback-Leibler divergence and the sparsity of the solution.

I. INTRODUCTION

In the past, graph-based optimization techniques have been successfully employed to provide an effective solution to the simultaneous localization and mapping (SLAM) problem. By exploiting the sparse nature of the problem, researchers have developed effective optimization algorithms to solve even large-scale and challenging SLAM problems [11, 15, 17]. In graph-based optimization techniques, the estimation problem is associated with a factor graph, whose nodes represent the variables to be estimated and whose factors represent the measurements between the nodes. In most cases, the observations have nonlinear measurements functions and are affected by Gaussian noise. For such cases it can be shown that performing inference on the factor graph representation is equivalent to nonlinear least squares minimization [6].

Unfortunately, when the number of variables is very large, the computational complexity of the estimation problem is high. In such cases, it is possible to reduce the problem size by eliminating a set of variables and minimizing the approximation loss in a statistical sense. To reduce the approximation error, the information related to the eliminated variables is preserved by marginalization. However, after successive marginalizations, the information matrix of the estimation problem becomes dense and sparsity enforcing methods need to be used [16, 3, 23, 13]. Unfortunately, the marginalization process relies on the linear Gaussian assumption and can potentially introduce errors due to a suboptimal linearization point.

In this paper, we address the nonlinear marginalization and sparsification on factor graphs with Gaussian noise by formulating an appropriate minimization problem. We seek to find the mean and the covariances of a set of nonlinear measurements that minimize the Kullback-Leibler Divergence (KLD) of the approximate distribution with respect to the original one. Our method has several advantages:

- The minimization problem is convex;

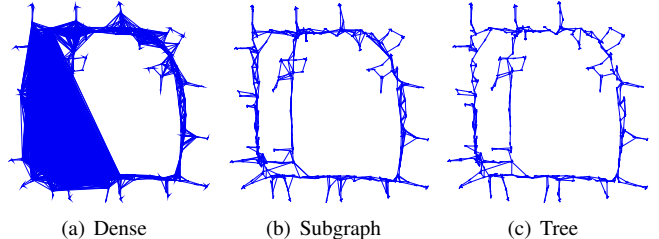(a) Dense   (b) Subgraph   (c) Tree

Fig. 1. Conditional dependence graphs for the Intel dataset with 66.6% node reduction using the three marginalization methods proposed in this paper.

- The approximation is performed locally, i.e., no global linearization point is needed;
- The approach is general, i.e., any nonlinear measurement function can be used;
- The solution preserves the block structure of the matrix;
- A closed form solution exists in some particular cases.

We present an extensive experimental evaluation of the proposed method on several publicly available datasets. We evaluate diverse sparsification and node reduction strategies and compare them with respect to the state of the art. The results demonstrate that our method significantly outperforms the others in terms of the KLD and the sparseness of the solution.

II. RELATED WORK

Many solutions to the SLAM problem rely on graph-based minimization techniques, e.g., iSAM [15], g^{2}o [17], or stochastic gradient descent [19, 10]. With these approaches, the dimension of the graph can grow unboundedly over time thus increasing the computational complexity, which is not desirable especially in the context of long-term operation of a mobile robot.

Over the last decade, numerous efforts have been made towards minimizing the computational requirements by reducing the amount of nodes and edges in the graph. In the context of filtering, Thrun et al. [21] introduced the sparse extended information filter (SEIF). The authors enforced sparsity whenever a node is marginalized by keeping only the edges with the largest entries (in terms of absolute value) in the information matrix. Eustice et al. [8] provided a modification to SEIF minimizing the differences between the SEIF estimate and that of a non-sparsified filter. Vial et al. [23] further extended SEIF by providing a method that ensures that the approximated information is strictly conservative. They also noted that the optimization need only be carried out on the Markov blanket of

the node to marginalize. Our approach is similar in spirit to the one of Vial et al. but we are not restricted to the filtering setting and we are able to recover nonlinear measurement functions.

Recently, Kretzschmar et al. [16] proposed a information-based criterion for determining which nodes to marginalize in a graph-based minimization setting. They further employed the Chow-Liu tree approximation [5] to sparsify the Markov blanket of the marginalized nodes. Carlevaris-Bianco and Eustice [3, 4] extended the previous work by introducing *Generic linear constraint* (GLC) factors. GLCs are n-ary edges of a factor graph, either dense or based on the Chow-Liu tree, which approximate the information matrix of the Markov blanket. With respect to those approaches, we explicitly consider nonlinear measurement functions and provide a sound mathematical framework based on convex minimization.

Huang et al. [13] approximated the dense information matrix solving an ℓ_1-regularized minimization problem. They used the alternating direction of multipliers method (ADMM) [2] to solve the problem and determine a conservative and sparse approximation. The approach, however, requires the information matrix of the full graph. On the contrary, our approach is local, in the sense that we directly operate on the Markov blanket of the marginalized node. Moreover, we explicitly consider nonlinear measurements and the block structure of the state space, while they commit on a linearization point and do not preserve the block structure.

Sparsification problems similar to the one presented in this work have been considered in the machine learning community, under the name of *Sparse Inverse Covariance Selection* (SICS). Banerjee et al. [1] first introduced the problem of estimating the sparsity pattern of an information matrix from a dense covariance by regularizing it with an ℓ_1 penalizer. They showed that the dual problem has a simpler solution and employed a block coordinate descent algorithm. The work has been extended by Friedman et al. [9] with the introduction of the *Graphical Lasso*. They modified the Lasso algorithm to work directly on the primal problem and showed an improved convergence speed.

Duchi et al. [7] extended the approach to deal with block sparsity by introducing an $\ell_{1,\infty}$ regularization term. They solve the resulting minimization problem using a projected gradient descent algorithm. Schmidt et al. [20] introduced the projected quasi newton algorithm (PQN) and showed its application to block inverse covariance selection problems.

Our method shares some grounds with the block inverse covariance selection problem with the difference being that we aim to obtain a set of nonlinear measurements that approximate the target information, instead of a specific information matrix.

III. NONLINEAR INVERSE COVARIANCE RECOVERY

In this section we describe our problem formulation for nonlinear inverse covariance recovery. Suppose that we have a multivariate normal distribution $p(\mathbf{x})$, with mean $\boldsymbol{\mu}$ and information matrix $\boldsymbol{\Omega}$. Moreover, suppose that we have a set of m independent nonlinear measurements $\mathbf{z}_i$ with measurement

functions $\mathbf{f}_i(\mathbf{x})$. These functions, for instance, can be derived from a sensor model or can be defined by an expert user.

Given a linearization point over $\mathbf{x}$ and the nonlinear measurements $\mathbf{z}_i$, we can estimate the first two standardized moments of the distribution over $\mathbf{x}$ by maximum likelihood inference. The two computed moments thus define a distribution $q(\mathbf{x})$, and, since a Gaussian is the maximum entropy distribution defined by a mean and covariance, we will assume $q(\mathbf{x})$ to be normally distributed.

Our goal is to approximate $p(\mathbf{x})$ with $q(\mathbf{x})$ as well as possible. This problem setting entails computing a mean $\boldsymbol{\nu}_i$ and an information matrix $\mathbf{X}_i$ for each measurement $\mathbf{z}_i$. For the mean, we compute $\boldsymbol{\nu}_i = \mathbf{f}_i(\boldsymbol{\mu})$. It is clear that the set of $\boldsymbol{\nu}_i$ will maximize the likelihood of each observation.

Computing the information matrices of the measurements, on the other hand, cannot always be done exactly. Let $\mathbf{f}(\mathbf{x})$ and $\mathbf{z}$ be the vectors obtained by stacking all $\mathbf{f}_i(\mathbf{x})$ and all of the measurement random vectors, respectively. We define the following matrices:

$$
\mathbf{A} = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}=\boldsymbol{\mu}} \qquad \mathbf{X} = \mathrm{cov}(\mathbf{z})^{-1} = \begin{bmatrix} \mathbf{X}_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \mathbf{X}_{m\cdot} \end{bmatrix} \quad (1)
$$

Notice that $\mathbf{X}$ is block diagonal due to the assumption of independence between nonlinear measurements.

We compute the measurement information matrix $\hat{\mathbf{X}}$ that minimizes the Kullback-Leibler divergence between $p(\mathbf{x})$ and $q(\mathbf{x})$

$$
\begin{aligned}
\hat{\mathbf{X}} &= \underset{\mathbf{X} \in \mathcal{X}}{\arg\min}\, D_{KL}(p(\mathbf{x}) \| q(\mathbf{x})) \\
&= \underset{\mathbf{X} \in \mathcal{X}}{\arg\min}\, \left\langle \mathbf{A}^T \mathbf{X} \mathbf{A}, \boldsymbol{\Sigma} \right\rangle - \log \det \left(\mathbf{A}^T \mathbf{X} \mathbf{A} \right), \quad (2)
\end{aligned}
$$

where $\mathbf{A}^T \mathbf{X} \mathbf{A}$ is the information matrix associated to the distribution $q(\mathbf{x})$, $\langle \cdot, \cdot \rangle$ denotes the matrix scalar product, and $\mathcal{X}$ is the set of block diagonal positive semi-definite matrices.

We will assume $\mathbf{A}$ to be full rank with at least as many rows as columns, and $\boldsymbol{\Omega}$ to be invertible, with inverse $\boldsymbol{\Omega}^{-1} = \boldsymbol{\Sigma}$. We will explore the problem of a singular $\boldsymbol{\Omega}$ in Section IV-A.

In general, it is hard to compute a closed-form solution to problem (2). However, the following proposition ensures that it is at least possible to guarantee global convergence to its global optimum.

Proposition 1. *Problem* (2) *is a convex optimization problem.*

Proof: The matrix $\mathbf{A}^T \mathbf{X} \mathbf{A}$ can be factored as sum of symmetric matrices weighted on the variables $\mathbf{X} = [x_{ij}]_{ij}$:

$$
\begin{aligned}
\mathbf{A}^T \mathbf{X} \mathbf{A} &= \sum_i x_{ii} \mathbf{A}^T \mathbf{J}^{ii} \mathbf{A} + \sum_{j>i} x_{ij} \mathbf{A}^T \left(\mathbf{J}^{ij} + \mathbf{J}^{ji} \right) \mathbf{A} \\
&= \sum_i x_{ii} \mathbf{G}_{ii} + \sum_{j>i} x_{ij} \mathbf{G}_{ij} = \mathbf{G}(\mathbf{X}). \quad (3)
\end{aligned}
$$

Here, $\mathbf{J}^{ij}$ denotes the single entry matrix with 1 at position (i, j) and 0 in all other entries, while $\mathbf{G}_{ij} = \mathbf{G}_{ij}^T$ by

construction. Furthermore, this result also implies that

$$\langle \mathbf{A}^T \mathbf{X} \mathbf{A}, \boldsymbol{\Sigma} \rangle = \sum_{j \geq i} x_{ij} \langle \mathbf{G}_{ij}, \boldsymbol{\Sigma} \rangle = \mathbf{c}^T \text{vec}(\mathbf{X}), \qquad (4)$$

where $\text{vec}(\cdot)$ denotes the vectorization operator and $\mathbf{c}$ is the vector of coefficients implicitly defined by the sum.

If we take into account that $-\log x = \log x^{-1}$, problem (2) can be expressed as

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}^T \text{vec}(\mathbf{X}) + \log \det \mathbf{G}(\mathbf{X})^{-1} \\ \text{subject to} \quad & \mathbf{G}(\mathbf{X}) \succeq \mathbf{0} \qquad\qquad (5) \\ & \mathbf{X} \succeq \mathbf{0}. \end{aligned}$$

This, together with the observation that $\mathbf{G}(\mathbf{X})$ is a sum of symmetric matrices weighted by the optimization variables, ensures that (2) is an instance of the MAXDET problem [22] and therefore is convex. ∎

A. Solving the optimization problem

The convexity of (2) allows us to efficiently and always compute the $\mathbf{X}$ that globally minimizes the KLD. Furthermore, we note that $\log \det \mathbf{G}(\mathbf{X})$ provides a natural barrier function for the positive definiteness of the approximating information matrix so long as $\mathbf{X}$ is positive definite, while the block diagonal structure of $\mathbf{X}$ is maintained by only optimizing with respect to the variables on the block diagonal.

Nevertheless, the definite positiveness of $\mathbf{X}$ needs to be maintained. In order to efficiently optimize the cost function while remaining in the feasible set, we use the *Limited-memory Projected Quasi-Newton* algorithm (PQN) [20]. Being a modification of the L-BFGS algorithm [18], PQN allows to achieve super-linear convergence without the quadratic memory requirements of a Hessian-based method. To be efficient, PQN requires a fast computation of the Euclidean projection onto the constraint set and a fast computation of the gradient.

For the first requirement, the projection function is:

$$\mathcal{P}(\mathbf{X}) = \underset{\mathbf{Y} \succeq \mathbf{0}}{\arg\min} \|\mathbf{X} - \mathbf{Y}\|_F^2, \qquad (6)$$

where $\|\cdot\|_F$ represents the Frobenius norm. For an arbitrary symmetric matrix $\mathbf{X}$ this problem can be solved in closed form [12] by taking the eigendecomposition $\mathbf{X} = \mathbf{V} \text{diag}(\lambda_i) \mathbf{V}^T$ and computing:

$$\mathcal{P}(\mathbf{X}) = \mathbf{V} \text{diag}(\max\{0, \lambda_i\}) \mathbf{V}^T. \qquad (7)$$

Since in our case $\mathbf{X}$ is block diagonal, this process can be carried out independently for each block, resulting in a very efficient linear-time projection.

The second requirement of PQN is the ability to compute the gradient of the cost function. In our case, a closed form expression exists:

$$\frac{\partial D_{KL}}{\partial \mathbf{X}_i} = \left\{ \mathbf{A} \left[\boldsymbol{\Sigma} - \left(\mathbf{A}^T \mathbf{X} \mathbf{A} \right)^{-1} \right] \mathbf{A}^T \right\}_i, \qquad (8)$$

where $\{\cdot\}_i$ denotes the i-th diagonal block of the enclosed matrix. The full gradient is obtained by stacking the vectorization of (8) for all $i \in \{1, \dots, m\}$.

Furthermore, when $\mathbf{A}$ is square, PQN is not required, as a closed form solution to (2) exists:

$$\frac{\partial D_{KL}}{\partial \mathbf{X}_i} = \left\{ \mathbf{A} \boldsymbol{\Sigma} \mathbf{A}^T - \mathbf{X}^{-1} \right\}_i = \mathbf{0} \qquad (9)$$

$$\Rightarrow \mathbf{X}_i = \left(\left\{ \mathbf{A} \boldsymbol{\Sigma} \mathbf{A}^T \right\}_i \right)^{-1}. \qquad (10)$$

Since $\mathbf{A}$ is a full-rank square matrix by assumption, $\mathbf{A} \boldsymbol{\Sigma} \mathbf{A}^T$ is positive definite by construction. Every principal minor of a positive definite matrix is positive, therefore $\left\{ \mathbf{A} \boldsymbol{\Sigma} \mathbf{A}^T \right\}_i$ is also positive definite, implying $\mathbf{X}_i \succeq \mathbf{0}$ and hence $\mathbf{X} \succeq \mathbf{0}$.

The computation of (8) and (10) might be very expensive for large matrices. Fortunately, we are only interested in computing the blocks on the main diagonal. For any $\mathbf{Y}$, we have:

$$\left\{ \mathbf{A} \mathbf{Y} \mathbf{A}^T \right\}_i = \sum_j \sum_k \mathbf{A}_{ij} \mathbf{Y}_{jk} \mathbf{A}_{ik}^T, \qquad (11)$$

which can be computed very efficiently if $\mathbf{A}$ is sparse. To recover the gradient (8) we set $\mathbf{Y} = \boldsymbol{\Sigma} - \left(\mathbf{A}^T \mathbf{X} \mathbf{A} \right)^{-1}$. Similarly, to recover the closed-form solution (10) we set $\mathbf{Y} = \boldsymbol{\Sigma}$.

IV. APPLICATIONS TO SLAM SPARSIFICATION

In this section, we show how the general framework introduced in the previous section can be used for SLAM problems. In particular, we consider the problem of node marginalization and edge sparsification in the context of 2D graph-based SLAM. Without loss of generality, we assume that there exists a node removal method that specifies which node to marginalize. This includes strategies such as the work of Kretschmar et al. [16], or strategies based on the Euclidean distance [14].

The nonlinear functions $\mathbf{f}_i(\mathbf{x})$ that we consider in this work belong to the SE(2) group and represent the rigid transformations between two poses. In order to obtain the SE(2) approximation, we perform the following steps:

1) We extract the Markov blanket of the node to be marginalized in the original graph;
2) We optimize the nodes in the Markov blanket using only measurements within the blanket, thus obtaining a *local* linearization point;
3) We marginalize the node to be removed by computing its Schur complement;
4) We select a set of virtual measurements to be used in the approximation;
5) We solve (2) by using the Jacobian of the measurements evaluated at the local linearization point and the Schur complement;

We use the Markov blanket to approximate the non-linearities of the original problem as closely as possible. This particular choice has the advantage that the procedure is independent of the global linearization point and can be carried out without needing to optimize the full graph beforehand. We select the virtual measurements according to the *sparsity pattern* that we want to maintain on the Markov blanket. This may be a fully dense set of $n(n-1)$ connections or the output

of the method presented in Section IV-B. Either way, once the sparsity pattern is known, we can compute the Jacobian $\mathbf{A}$ evaluated at the local linearization point and solve (2).

Finally, we propagate the solution of (2) into the original graph by replacing the measurements within the Markov blanket with the newly compute ones.

A. Handling Rank Deficient Information Matrices

In Section III, we assumed the information matrix $\boldsymbol{\Omega}$ to be invertible. Unfortunately, when dealing with problems such as variable marginalization, this is often not the case. From a SLAM perspective, this implies that the Markov blanket lacks a rigid transformation edge to the world reference frame.

In such a scenario, $\boldsymbol{\Omega}$ has k null eigenvalues, where k is the dimensionality of a rigid transformation, e.g., 3 for SE(2) and 6 for SE(3). If $\boldsymbol{\Omega}$ has n rows, the distribution of $p(\mathbf{x})$ is actually an $(n - k)$-dimensional multivariate normal embedded in an n-dimensional space. Therefore, we propose to project $p(\mathbf{x})$ and $q(\mathbf{x})$ onto the $(n - k)$-dimensional informative subspace and to compare the resulting $(n-k)$-dimensional distributions.

We compute an $(n-k) \times n$ projection matrix $\boldsymbol{\Pi}$ by stacking the transpose of the eigenvectors of $\boldsymbol{\Omega}$ corresponding to the nonzero eigenvalues. Since $\boldsymbol{\Omega}$ is a symmetric real-valued matrix its eigendecomposition $\boldsymbol{\Omega} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T$ is real-valued and always exists:

$$\boldsymbol{\Omega} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T = \begin{bmatrix} \mathbf{U}_0 & \boldsymbol{\Pi}^T \end{bmatrix} \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Lambda}_+ \end{bmatrix} \begin{bmatrix} \mathbf{U}_0^T \\ \boldsymbol{\Pi}. \end{bmatrix} \qquad (12)$$

The projection matrix $\boldsymbol{\Pi}$ acts as a linear operator to project any arbitrary information matrix $\boldsymbol{\Psi}$ onto the lower dimensional space by computing $\boldsymbol{\Pi}\boldsymbol{\Psi}\boldsymbol{\Pi}^T$. To account for singular information matrices, we apply the following substitutions into (2):

$$\mathbf{A} \mapsto \mathbf{A}\boldsymbol{\Pi}^T \qquad (13)$$

$$\boldsymbol{\Sigma} \mapsto \boldsymbol{\Lambda}_+^{-1}. \qquad (14)$$

This, however, does not guarantee that $q(\mathbf{x})$ is an $(n - k)$-dimensional distribution embedded in an n-dimensional space. We enforce this property by artificially limiting the rank of the Jacobian matrix $\mathbf{A}$, i.e., by avoiding to put any edge relating a pose to the world reference frame.

The same efficient implementation for computing the gradient can also be applied in the low rank case by substituting $\mathbf{Y} = \boldsymbol{\Pi}^T \left[\boldsymbol{\Sigma} - \left(\boldsymbol{\Pi}\,\mathbf{A}^T\mathbf{X}\,\mathbf{A}\,\boldsymbol{\Pi}^T \right)^{-1} \right] \boldsymbol{\Pi}$ into (11).

B. Selecting the Virtual Measurements

To sparsify the Markov blanket, one needs to define the size of $\mathbf{X}$ and the structure of $\mathbf{A}$. In this paper, we propose two strategies to select the virtual measurements: tree-based and subgraph-based.

For the tree-based approximation, we extend the Chow-Liu tree (CLT) [5] approximation to handle rank deficient information matrices. The approximation is computed by finding the maximum spanning tree of the mutual information graph, i.e., a weighted graph whose edges represent the mutual information between the connecting nodes. In such a scenario, the mutual information:

$$I(\mathbf{x}_i, \mathbf{x}_j) = -\frac{1}{2} \log \frac{\det \left(\boldsymbol{\Omega}_{ii} - \boldsymbol{\Omega}_{ij}\boldsymbol{\Omega}_{jj}^{-1}\boldsymbol{\Omega}_{ji} \right)}{\det \boldsymbol{\Omega}_{ii}} \qquad (15)$$

is not well defined, since $\boldsymbol{\Omega}_{ii} - \boldsymbol{\Omega}_{ij}\boldsymbol{\Omega}_{jj}^{-1}\boldsymbol{\Omega}_{ji}$ is a null matrix. This is the case because the mutual information formula requires $\boldsymbol{\Omega}_{ii}$, $\boldsymbol{\Omega}_{jj}$, and $\boldsymbol{\Omega}_{ij}$ to be obtained by marginalizing all $\mathbf{x}_k$ with $k \notin \{i, j\}$. Thus, a further Schur complement will yield a null matrix due to gauge freedom.

To address this problem, Carlevaris-Bianco and Eustice [3] adopted the Tikhonov-regularized matrix $\mathbf{X} + \varepsilon\,\mathbf{I}$, with $\varepsilon = 1$, when computing the determinant of a singular matrix $\mathbf{X}$.

However, computing (15) between each pair of variables requires a quadratic number of Schur complements, hence a quintic computational complexity in the number of nodes. To reduce this complexity, we propose to compute the covariance form of the mutual information

$$I(\mathbf{x}_i, \mathbf{x}_j) = \frac{1}{2} \log \frac{\det \hat{\boldsymbol{\Sigma}}_{ii} \det \hat{\boldsymbol{\Sigma}}_{jj}}{\det \begin{bmatrix} \hat{\boldsymbol{\Sigma}}_{ii} & \hat{\boldsymbol{\Sigma}}_{ij} \\ \hat{\boldsymbol{\Sigma}}_{ji} & \hat{\boldsymbol{\Sigma}}_{jj} \end{bmatrix}}, \qquad (16)$$

where $\hat{\boldsymbol{\Sigma}} = (\boldsymbol{\Omega} + \varepsilon\,\mathbf{I})^{-1}$ is the inverse of the Tikhonov-regularized information matrix $\boldsymbol{\Omega}$. For the efficient computation of the inverse, we employ the Cholesky decomposition.

For the subgraph-based approximation, we extend the Chow-Liu tree by adding additional chords to the spanning tree. Although finding the best spanning subgraph is an NP-hard problem, we employ a greedy heuristic and leverage on the *"information never hurts"* principle. We consider the subgraphs obtained by adding the chords with the highest mutual information. We choose the number of chords as a fixed proportion of the number of edges in the spanning tree. This allows us to maintain a set of edges which grows linearly with respect to the number of nodes, thus avoiding the quadratic fill-in of full marginalization.

When using a tree approximation, we can solve (2) in closed form. Let k be the dimensionality of a measurement, and n the number of random variables. The Jacobian matrix $\mathbf{A}$ is a $(n - 1)k \times nk$ matrix. Under the projection $\boldsymbol{\Pi}$, $\mathbf{A}\boldsymbol{\Pi}^T$ is square and full rank, hence invertible and (10) can be used. The subgraph-based approximation, on the other hand, cannot be computed in closed form as the matrix $\mathbf{A}$ remains rectangular even when projected. For these problems we adopt the PQN-based iterative solver.

C. Computing the Initial Guess for PQN

For an efficient usage of PQN, we need to provide an initial guess close to the optimal solution. In case of non-redundant edges, the maximum likelihood information matrix $\mathbf{A}^T\mathbf{X}\mathbf{A}$ has the block structure:

$$\{\mathbf{A}^T\mathbf{X}\mathbf{A}\}_{jk} = \begin{cases} \sum_i \mathbf{A}_{ij}^T\mathbf{X}_{ii}\mathbf{A}_{ij} & \text{if } j = k \\ \mathbf{A}_{\star j}^T\mathbf{X}_{\star\star}\mathbf{A}_{\star k} & \text{if } j \neq k, \end{cases} \qquad (17)$$
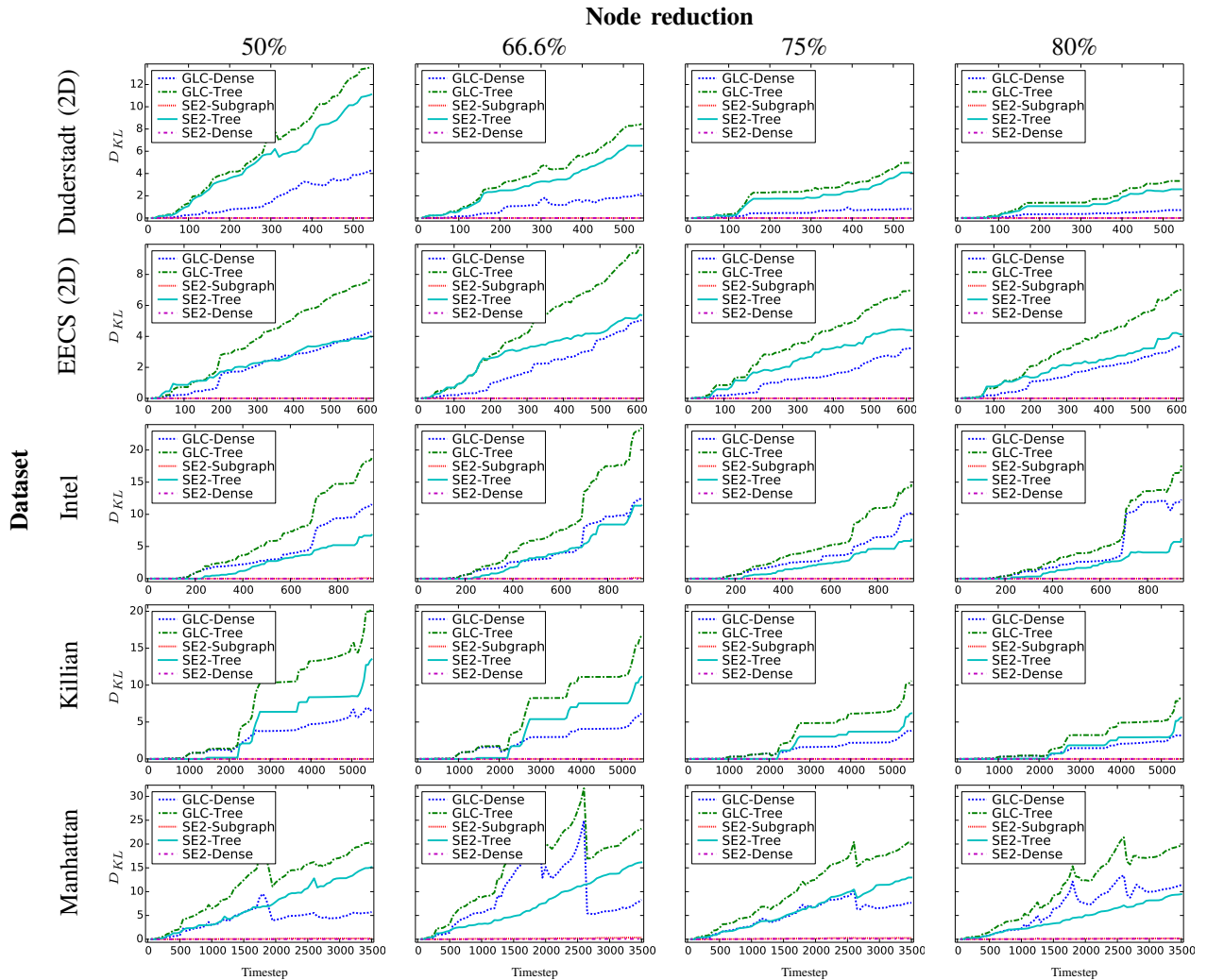
Fig. 2. Online sparsification results. Each graph represents the KLD as a function of the time step for the five methods considered. Notice that "SE2-Dense" and "SE2-Subgraph" overlap most of the time and hence it is difficult to distinguish the differences.

where $\{\cdot\}_{ij}$ denotes the (i,j)-th block of the enclosed matrix and $\star$ denotes the row index in $\mathbf{A}$ of the unique measurement relating random variables j and k.

If equality between $\mathbf{A}^T \mathbf{X} \mathbf{A}$ and $\boldsymbol{\Omega}$ can be achieved and all of the entries $\mathbf{A}_{ij}$ are invertible, the following holds:

$$\mathbf{X}_\star = \mathbf{A}_{\star j}^{-T} \boldsymbol{\Omega}_{jk} \mathbf{A}_{\star k}^{-1}. \tag{18}$$

Even in the case that equality cannot be achieved, we can use (18) to provide an initial guess $\hat{\mathbf{X}}$. If the resulting $\mathbf{X}_\star$ is either indefinite or not symmetric, we apply (7) to the symmetric matrix closest to (18) [12].

V. EXPERIMENTS

We implemented our method by relying on g²o [17] as an optimization back-end and we evaluate its practical effectiveness against GLC [3] on 2D datasets. We devise three different test scenarios: a fully online marginalization method, a periodic batch method and a full batch method.

In the full online setting, we adopt the node removal strategy described by Carlevaris-Bianco and Eustice [4, Alg. 3]. The

graph is created online and we marginalize the recently added node in case it is spatially redundant. In the full batch setting, we consider the full graph and we marginalize all the spatially redundant nodes one after another. In the periodic batch method, we regularly perform a full batch sparsification of the last 100 nodes collected. To simulate the spatial redundancy, we only keep one node every t time steps.

The first two strategies represent typical approaches for online graph sparsification. We chose the last strategy to compare our approach when the linearization point for GLC is the closest to the optimal solution.

For each experiment, we compare the KLD of the sparse solution with the full, non-marginalized graph. For the online and periodic batch case, we need to account for edges that connect a new node with an already marginalized one. We replace, in both the approximate and the baseline graph, the original edge with an edge to the closest node and propagate the measurement accordingly.

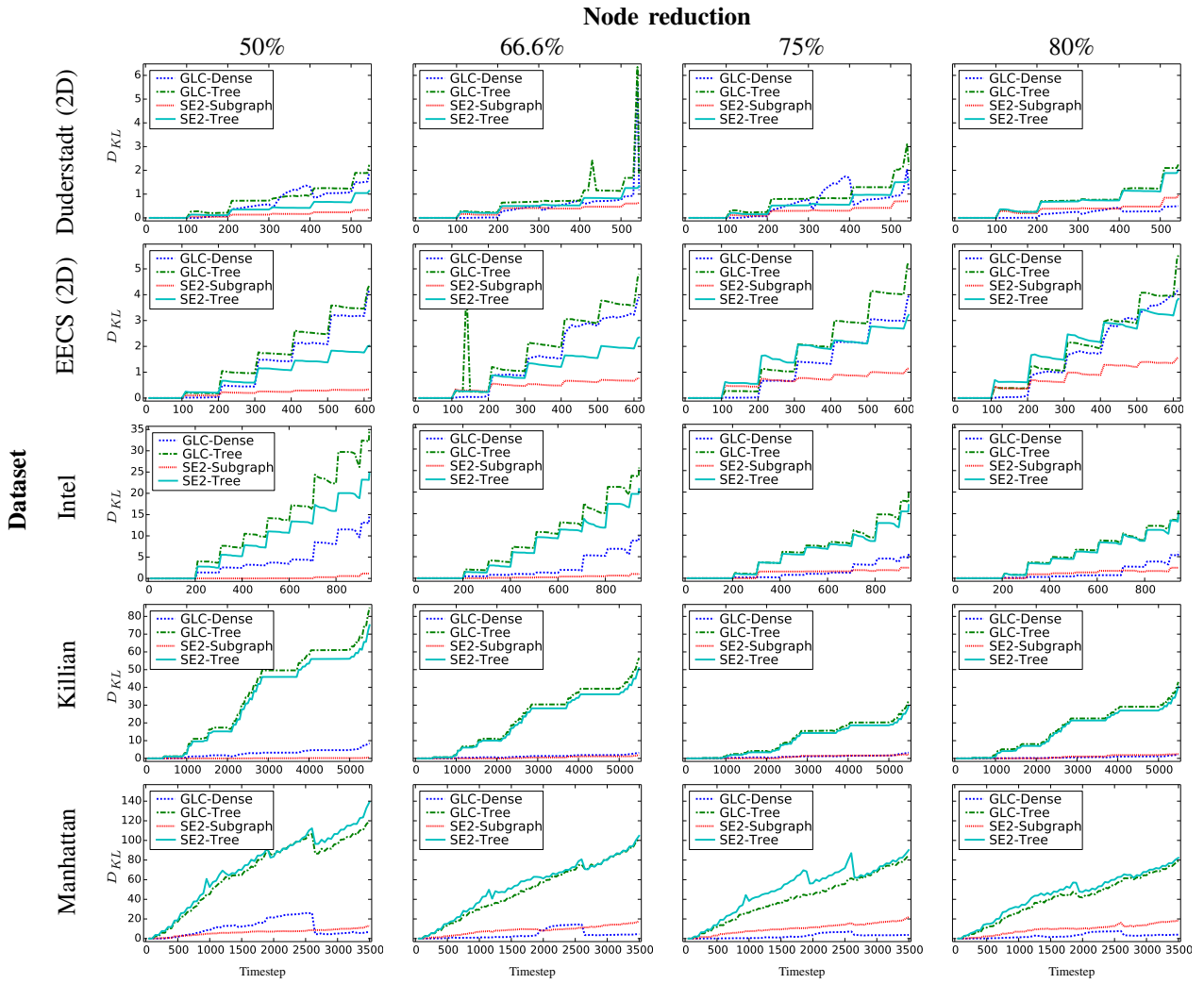We evaluated the proposed approach on five SLAM datasets:

Node reduction

Fig. 3. Periodic batch sparsification results. Each graph represents the KLD as a function of the time step for the five methods considered.

Dataset	# Nodes	# Edges	Fill-in
Duderstadt Center (2D)	545	1800	1.32%
EECS Building (2D)	615	2134	1.25%
Intel Research	943	1833	0.52%
MIT Killian	5489	7626	0.069%
Manhattan	3500	5596	0.12%

Duderstadt Center, EECS Building, Intel Research, MIT Killian and Manhattan. Given the 2D nature of our approach, we considered the Duderstadt Center and EECS Building dataset projected onto the SE(2) manifold. Table I provides an overview of the characteristics of each dataset.

We considered four levels of node reduction for each scenario, keeping one node every two, three, four or five nodes. In the comparison, we evaluated our approach using a set of dense measurements (*SE2-Dense*), a subgraph with twice the number of edges of the spanning trees (*SE2-Subgraph*) and the Chow-Liu tree (*SE2-Tree*). For GLC, we consider the dense

version (*GLC-Dense*) and the Chow-Liu tree one (*GLC-Tree*).

Fig. 2 depicts the KLD results for the full online strategy. In this scenario, SE2-Dense and SE2-Subgraph obtain near-optimal results with a KLD close to zero in all the cases. Moreover, SE2-Tree is always better than GLC-Tree and even better than GLC-Dense in some cases. The dense and subgraph approaches are a viable choice in this case, since all strategies maintain a sparse information matrix, with about 1% increase in fill-in between the tree and the dense approach.

Fig. 3 shows the KLD results for the periodic batch sparsification scenario. We omitted SE2-Dense in this comparison, given its computational requirements for large Markov blankets. In such cases, the number of variables considered is typically too high for practical computation. This is not the case when using SE2-Subgraph, since the number of variable is linear in the number of nodes. The *saw-tooth* behavior of the graphs is due to the periodic marginalization.

In general, the SE2 approaches obtain better performance than the corresponding GLC equivalents. In Manhattan, GLC is slightly better, due to the characteristic of the data. The

Dataset	Node reduction	KLD			Fill-in		
		GLC-Tree	SE2-Tree	SE2-Subgraph	GLC-Tree	SE2-Tree	SE2-Subgraph
Duderstadt (2D)	50%	1.328	1.307	**0.975**	1.48%	**1.41%**	2.24%
	66.6%	1.556	1.695	**1.179**	2.18%	**2.14%**	3.11%
	75%	1.963	2.233	**1.628**	2.89%	**2.77%**	3.92%
	80%	4.014	2.870	**1.809**	3.40%	**3.38%**	4.84%
EECS (2D)	50%	1.996	2.519	**1.672**	1.99%	**1.87%**	2.91%
	66.6%	**2.907**	3.770	3.006	3.12%	**2.99%**	4.58%
	75%	4.253	6.180	**4.200**	4.35%	**3.94%**	6.28%
	80%	5.064	7.322	**4.677**	5.59%	**5.00%**	8.16%
Intel	50%	46.84	55.79	**17.36**	**0.88%**	0.88%	1.22%
	66.6%	43.70	52.95	**22.72**	**1.27%**	1.27%	1.77%
	75%	39.70	48.78	**19.30**	1.65%	**1.63%**	2.25%
	80%	41.39	50.02	**18.37**	**1.91%**	1.92%	2.66%
Killian	50%	74.36	75.06	**0.53**	**0.13%**	**0.13%**	0.17%
	66.6%	151.41	154.02	**46.16**	**0.18%**	**0.18%**	0.25%
	75%	74.69	76.52	**17.61**	**0.24%**	**0.24%**	0.34%
	80%	128.79	132.90	**40.54**	**0.29%**	**0.29%**	0.40%
Manhattan	50%	163.06	213.37	**32.43**	**0.26%**	**0.26%**	0.38%
	66.6%	155.69	172.46	**46.29**	0.40%	**0.39%**	0.62%
	75%	142.13	159.35	**61.73**	0.54%	**0.52%**	0.78%
	80%	141.49	154.11	**60.51**	0.65%	**0.64%**	0.95%

TABLE II
GLOBAL SPARSIFICATION RESULTS

graph contains many small consecutive loops, creating very rigid Markov blankets. Hence, the linearization points are close to the optimal solution, favoring GLC over SE2. SE2-Subgraph outperforms GLC-Tree, and is generally better or comparable to GLC-Dense. With respect to the fill-in, SE2-Subgraph produces a much sparser graph, with a maximum increase of about 4%. In contrast, GLC-Dense produces a denser graph, with a fill-in increase of about 22%.

Finally, Table II presents the numeric results in terms of KLD and fill-in for the batch sparsification test. As in the periodic batch case, we do not report the results of GLC-Dense and SE2-Dense due to the computation requirements. The batch scenario is the most favorable to GLC, since the linearization point is close to the optimal one. Nevertheless, the results of SE2-Tree and GLC-Tree are comparable. The clear winner in terms of KLD is the SE2-Subgraph method, as it achieves significant reduction in KLD with only an increase of about 3% in fill-in.

Although GLC-Tree and SE2-Tree both rely on the Chow-Liu tree to compute a sparsity pattern, each time a node is marginalized both methods provide a different approximation of the information matrix. In subsequent marginalizations, they may produce different Chow-Liu trees: this explains the difference in fill-in between the two methods, as the connectivity of each node depends on which edges were selected in the tree.

To visually convey the performance of SE2-Subgraph with respect to a dense marginalization, we depict in Fig. 4 the sparsity pattern of the resulting information matrix after marginalization. The figure shows that the proposed subgraph approach is able to produce a highly sparse approximation even in the most challenging batch scenario. Despite being very sparse, SE2-Subgraph produces the closest approximation to the original distribution in terms of KLD.
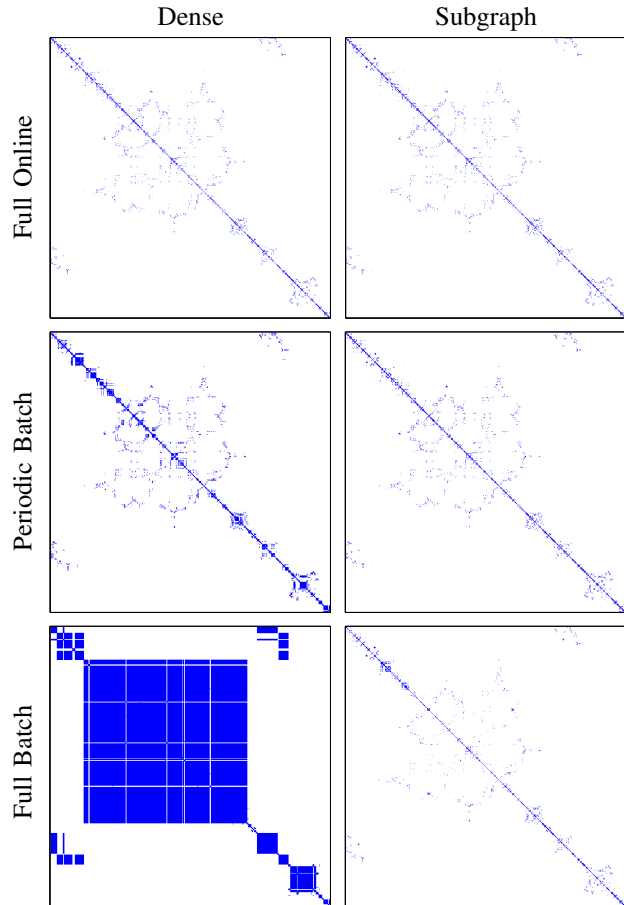

Fig. 4. Sparsity pattern of the Manhattan dataset with 80% reduction. The figure shows the fill-in of the information matrix for the dense and subgraph-based marginalization using the three strategies.

VI. Conclusion

In this paper, we presented a novel approach to non-linear graph sparsification. Our approach is formulated as a convex minimization problem that finds the set of nonlinear measurements that best approximate the original distribution. Our technique has the advantage of not requiring a global linearization point and can be used with any nonlinear measurement function. We presented extensive experiments carried out on publicly available datasets and demonstrated the effectiveness of our approach. We quantified the algorithm performance in the SLAM context by sparsifing maps in an online and in a periodic batch fashion. In both cases, our technique outperforms state-of-the-art methods by closely recovering the original distribution and producing highly sparse graphs.

VII. Acknowledgements

References

[1] O. Banerjee, L. E. Ghaoui, A. d'Aspremont, and G. Natsoulis. Convex optimization techniques for fitting sparse Gaussian graphical models. In *Proc. of the Int. Conf. on Machine Learning*, 2006.

[2] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

[3] N. Carlevaris-Bianco and R. M. Eustice. Generic factor-based node marginalization and edge sparsification for pose-graph SLAM. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 2013.

[4] N. Carlevaris-Bianco and R.M. Eustice. Long-term simultaneous localization and mapping with generic linear constraint node removal. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2013.

[5] C. I. Chow and C. N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Trans. on Information Theory*, 14:462–467, 1968.

[6] F. Dellaert and M. Kaess. Square Root SAM: Simultaneous Localization and Mapping via Square Root Information Smoothing. *Int. Journal of Robotics Research*, 25(12):1181–1204, 2006.

[7] J. C. Duchi, S. Gould, and D. Koller. Projected Subgradient Methods for Learning Sparse Gaussians. In *Proc. of the Conf. in Uncertainty in Artificial Intelligence*, 2008.

[8] R. Eustice, M. Walter, and J. Leonard. Sparse extended information filters: insights into sparsification. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2005.

[9] J. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008.

[10] G. Grisetti, C. Stachniss, S. Grzonka, and W. Burgard. A Tree Parameterization for Efficiently Computing Maximum Likelihood Maps using Gradient Descent. In *Proc. of Robotics: Science and Systems*, 2007.

[11] G. Grisetti, R. Kümmerle, C. Stachniss, and W. Burgard. A Tutorial on Graph-based SLAM. *IEEE Trans. on Intelligent Transportation Systems Magazine*, 2:31–43, 2010.

[12] N. J. Higham. Computing a nearest symmetric positive semidefinite matrix. *Linear Algebra and its Applications*, 103(0):103–118, 1988.

[13] G. Huang, M. Kaess, and J. J. Leonard. Consistent sparsification for graph optimization. In *Proc. of the European Conference on Mobile Robots*, 2013.

[14] H. Johannsson, M. Kaess, M.F. Fallon, and J.J. Leonard. Temporally Scalable Visual SLAM using a Reduced Pose Graph. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 2013.

[15] M. Kaess, A. Ranganathan, and F. Dellaert. iSAM: Fast Incremental Smoothing and Mapping with Efficient Data Association. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 2007.

[16] H. Kretzschmar, C. Stachniss, and G. Grisetti. Efficient information-theoretic graph pruning for graph-based SLAM with laser range finders. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2011.

[17] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. g2o: A General Framework for Graph Optimization. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 2011.

[18] J. Nocedal. Updating quasi-Newton matrices with limited storage. *Mathematics of computation*, 35(151):773–782, 1980.

[19] E. Olson, J. Leonard, and S. Teller. Fast iterative alignment of pose graphs with poor initial estimates. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 2006.

[20] M. Schmidt, E. van den Berg, M. P. Friedlander, and K. Murphy. Optimizing Costly Functions with Simple Constraints: A Limited-Memory Projected Quasi-Newton Algorithm. In *Proc. of the Int. Conf. on Artificial Intelligence and Statistics*, 2009.

[21] S. Thrun, Y. Liu, D. Koller, A. Y. Ng, Z. Ghahramani, and H. Durrant-Whyte. Simultaneous Localization and Mapping with Sparse Extended Information Filters. *Int. Journal of Robotics Research*, 23(7-8):693–716, 2004.

[22] L. Vandenberghe, S. Boyd, and S. P. Wu. Determinant maximization with linear matrix inequality constraints. *SIAM journal on matrix analysis and applications*, 19 (2):499–533, 1998.

[23] J. Vial, H. Durrant-Whyte, and T. Bailey. Conservative Sparsification for efficient and consistent approximate estimation. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2011.